

**An NQTHM Mechanization of "An Exercise in the
Verification of Multi-Process Programs"**

by

M. Nagayama, C. Talcott

Department of Computer Science

Stanford University

Stanford, California 94305



REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704-0188	
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.				
1. AGENCY USE ONLY (Leave blank)		2. REPORT DATE		3. REPORT TYPE AND DATES COVERED
4. TITLE AND SUBTITLE An NQTHM Mechanization of "An Exercise in the Verification of Multi-Process Programs"			5. FUNDING NUMBERS	
6. AUTHOR(S) Misao Nagayama + Carolyn Talcott				
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Computer Science Dept Stanford University Stanford CA 94305			8. PERFORMING ORGANIZATION REPORT NUMBER STAN-CS-91-1370	
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) DARPA 1400 Wilson Blvd. Arlington, VA 22209 NSF 1800 G Street Washington, D.C. 20550			10. SPONSORING / MONITORING AGENCY REPORT NUMBER	
11. SUPPLEMENTARY NOTES				
12a. DISTRIBUTION / AVAILABILITY STATEMENT Unlimited			12b. DISTRIBUTION CODE	
13. ABSTRACT (Maximum 200 words) This report presents a formal verification of the local correctness of a mutex algorithm using the Boyer-Moore theorem prover. The formalization follows closely an informal proof of Manna and Pnuelli. The proof method of Manna and Pnueli is to first extract from the program a set of states and induced transition system. One then proves suitable invariants. There are two variants of the proof. In the first (atomic) variant, compound tests involving quantification over a finite set are viewed as atomic operations. In the second (molecular) variant, this assumption is removed, making the details of the transitions and proof somewhat more complicated. The original Manna-Pnueli proof was formulated in terms of finite sets. This led to a concise and elegant informal proof, however one that is not easy to mechanize in the Boyer-Moore logic. In the mechanized version we use a dual isomorphic representation of program states based on finite sequences. Our approach was to outline the formal proof of each invariant, making explicit the case analyses, assumptions and properties of operations used. The outline served as our guide in developing the formal proof. The resulting sequence of events follows the informal plan quite closely. The main difficulties encountered were in discovering the precise form of the lemmas and hints necessary to guide the theorem prover.				
14. SUBJECT TERMS Verification, mechanical proving, mutex algorithm			15. NUMBER OF PAGES 84	
			16. PRICE CODE	
17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION Of THIS PAGE Unclassified	19. SECURITY CLASSIFICATION Of ABSTRACT Unclassified	20. LIMITATION Of ABSTRACT	

An NQTHM Mechanization of
 “An Exercise in the Verification of Multi-Process Programs”

Misao Nagayama Stanford University misao@cs.stanford.edu	Carolyn Talcott Stanford University clt@sail.stanford.edu
--	---

1. Introduction

This report presents a formal verification of the local correctness of a mutex algorithm using the Boyer-Moore theorem prover [1, 2]. The project arose out of a challenge given by Amir Pnueli, in a lecture at Stanford, to obtain a computer checked version of a proof of correctness presented in the lecture (cf. [4]).

The mutex algorithm is the following.

mutex :: *flag* : **array**[1..*n*] of 0..4 where *flag*[1..*n*] = 0
 $P[1] \parallel \dots \parallel P[n]$

where each process $P[i]$, $1 \leq i \leq n$, is given by:

```

local j : [1..n] where j = 0
l0 : loop forever do
  begin
    l1 : Non Critical
    l2 : flag[i] := 1
    l3 : wait until  $\forall j : 1 \leq j \leq n : (flag[j] < 3)$ 
    l4 : flag[i] := 3
    l5 : if  $\exists j : 1 \leq j \leq n : (flag[j] = 1)$  then
      begin
        l6 : flag[i] := 2
        l7 : wait until  $\exists j : 1 \leq j \leq n : (flag[j] = 4)$ 
      end
    l8 : flag[i] := 4
    l9 : wait until  $\forall j : 1 \leq j < i : (flag[j] < 2)$ 
    l10 : Critical
    l11 : wait until  $\forall j : i < j \leq n : (flag[j] < 2 \vee flag[j] > 3)$ 
    l12 : flag[i] := 0
  end
end
  
```

The correctness property to be proved is the *mutual exclusion* property that at any given time in the execution of **mutex** at most one process is executing the statement l_{10} . The proof method of Manna and Pnueli is to first extract from the program a set of states and induced

transition system. One then formulates correctness in terms of invariants-properties that must hold of states reachable from the initial state via any sequence of allowed transitions. Using the INV rule of [4], checking correctness is reduced to checking local invariance, i.e. that the initial state satisfies the invariants and that all allowed transitions preserve the invariant s . As usual with inductive proofs, to prove the mutex property it is necessary to analyse the transition system and discover a stronger invariant that implies the mutex property. There are two variants of the proof. In the first (atomic) variant, compound tests involving quantification over a finite set, for example (l_3) , are viewed as atomic operations. In the second (molecular) variant, this assumption is removed, making the transitions and proof somewhat more complicated.

Our formalization follows closely the proof given in [4]. We proceed as follows. We first define a representation of states, the transition relation, and the invariants to be proved in the Boyer-Moore logic. The original Manna-Pnueli proof was formulated in terms of finite sets. This led to a concise and elegant informal proof, however one that is not easy to mechanize. Thus we use a dual isomorphic representation of program states based on finite sequences. Then we outline the formal proof of each invariant, making explicit the case analyses, assumptions and properties of operations used. The outline served as our guide in developing the formal proof. What was required was to figure out how to state each lemma in the Boyer-Moore logic. Most lemmas are presented in the form $A_1 \wedge \dots \wedge A_n \rightarrow B$ as that is the form required by the theorem prover for rewriting, and is the most natural form for communicating with the theorem prover. There are, however, still a number of technical difficulties in writing lemmas provable by the theorem prover. They are mainly due to difficulty in controlling the rewriting process that is inference engine of the theorem prover. This is done by use of hints, and controlling the set of rewrite lemmas available for consideration.

The remainder of this report is organized as follows. In section 2. we present our formalization of the atomic variant and section 3. contains our formalization of the non-atomic variant. The complete formal proofs (input to the Boyer-Moore prover) appear as appendices. The proof outlines include names of the corresponding events (lemmas) used in the Boyer-Moore proof and are intended to serve as a reading guide for the fully formalized proof. Event names have been chosen systematically to reflect the lemma, case, or property being proved. Some comments on formalization techniques, difficulties, and alternatives are included as comments in the theorem prover input.

The Boyer-Moore logic is a quantifier-free first-order logic of tree structured data and functions defined by recursion on well-founded orderings. Properties are represented by boolean valued functions, and existential statements must be represented using functions that compute the quantity claimed to exist. A proof is a sequence of events. The events of interest here are definition events and prove-lemma events. Lemmas are proved by using propositional reasoning, rewriting, and induction on well-founded orderings. The user may provide guidance in the form of hints for prove-lemma events. The Boyer-Moore prover is implemented in Lisp and uses Lisp notation. We present several definitions and lemmas first in ordinary mathematical notation, then in the Boyer-Moore notation. For more details we refer the reader to [1, 2].

2. Formalization of the Atomic Variant in the Boyer-Moore Logic

2.1. States-Atomic Case

We let $Locs$ be the set of program counters, $Flgs$ be the set of flag values, and for n a positive integer, N_n is the set of positive integers less than or equal to n . An n process state is a pair (l, g) such that l maps N_n to $Locs$ — $Z(i)$ is the location (pc) of process i , and g maps N_n to $Flgs$ — $g(i)$ is the flag value of process i . $Ws[n]$ is the set of n -process states.

$$Locs = \{0, \dots, 12\}$$

$$Flgs = \{0, 1, 2, 3, 4\}$$

$$N_n = \{1, \dots, n\}$$

$$Ws [n] = [N, \rightarrow Locs] \times [N, \rightarrow Flgs]$$

We let n denote the number of processes. We will use i, j, k to range over N_n , and $l, l', \dots$ to range over location maps, and $g, g', \dots$ to range over flag maps.

In the Boyer-Moore logic, finite sets are represented as lists and the membership relation is represented by its characteristic function `member`. Thus N_n , $Locs$, and $Flgs$ are represented as particular lists of numbers. Maps from N_n are represented as lists. Application is represented both as a relation `at` and as a function `nth`. Updating is accomplished by `move`. We also define various forms of bounded quantification. Letting l be the function represented by `l`, s be the set represented by `s` etc. we have `(at l i k)` is true iff `(equal (nth l i) k)` is true iff $Z(i) = k$. `(move l i k)` represents the function l modified to have value k at i . `(union-at-n l i s)` is true just if $Z(i) \in s$ and `(all-union l n s)` is true just if $(\forall i \in N_n)(l(i) \in s)$. `(exists-union l n s)` returns some $i \in N_n$ such that $Z(i) \in s$, if such an i exists, and returns false otherwise. Thus states are recognized by the function `ws`.

```
(defn ws (n l g)
  (and (numberp n)
        (listp l)
        (listp g)
        (equal (length l) n)
        (equal (length g) n)
        (all-union l n '(0 1 2 3 4 5 6 7 8 9 10 11 12))
        (all-union g n '(0 1 2 3 4))))
```

The initial state (l_{init}, g_{init}) has each process at location 0 with flag value 0.

2.2. Transition Relation-Atomic Case

A transition is the execution of a local transition (statement) by one of the processes. The local transition relation ρ and the global transition relation $\mathcal{R}$ corresponding to the mutex program with quantification treated as an atomic operation are defined as follows.¹

¹ Although the number of processes is fixed, we include this as an explicit parameter of the transition relation and invariants to avoid having a free variable in the definition bodies. An alternative would be to introduce a constant, constrained to be a number, but otherwise uninterpreted.

Definition (transitions):

$$\mathcal{R}[n](l, g, l', g') \Leftrightarrow (\exists i \in N_n) \rho[n](i, l, g, l', g')$$

$$\rho[n](i, l, g, l', g') \Leftrightarrow \bigvee_{c \in C} \rho_c[n](i, l, g, l', g')$$

where $C = \{0, 1a, 1b, 2, 3a, 3b, 4, 5a, 5b, 6, 7a, 7b, 8, 9, 10, 11a, 11b, 12\}$ and the component transitions $\rho_c[n](i, l, g, l', g')$ are defined by

$$\rho_0[n](i, l, g, l', g') \Leftrightarrow Z(i) = 0 \wedge g' = g \wedge l' = l\{i := 1\}$$

$$\rho_{1a}[n](i, l, g, l', g') \Leftrightarrow Z(i) = 1 \wedge g' = g \wedge l' = l$$

$$\rho_{1b}[n](i, l, g, l', g') \Leftrightarrow l(i) = 1 \wedge g' = g \wedge l' = l\{i := 2\}$$

$$\rho_2[n](i, l, g, l', g') \Leftrightarrow l(i) = 2 \wedge g' = g\{i := 1\} \wedge l' = l\{i := 3\}$$

$$\rho_{3a}[n](i, l, g, l', g') \Leftrightarrow l(i) = 3 \wedge \phi_3[n](l, g) \wedge g' = g \wedge l' = l\{i := 4\}$$

$$\rho_{3b}[n](i, l, g, l', g') \Leftrightarrow l(i) = 3 \wedge \neg \phi_3[n](l, g) \wedge g' = g \wedge l' = l$$

$$\phi_3[n](l, g) \Leftrightarrow (\forall j \in N_n)(g(j) \neq 3 \wedge g(j) \neq 4) \Leftrightarrow (\forall j \in N_n)(g(j) \in \{0, 1, 2\})$$

$$\rho_4[n](i, l, g, l', g') \Leftrightarrow l(i) = 4 \wedge g' = g\{i := 3\} \wedge l' = l\{i := 5\}$$

$$\rho_{5a}[n](i, l, g, l', g') \Leftrightarrow l(i) = 5 \wedge \phi_5[n](l, g) \wedge g' = g \wedge l' = l\{i := 6\}$$

$$\rho_{5b}[n](i, l, g, l', g') \Leftrightarrow l(i) = 5 \wedge \neg \phi_5[n](l, g) \wedge g' = g \wedge l' = l\{i := 8\}$$

$$\phi_5[n](l, g) \Leftrightarrow (\exists j \in N_n)(g(j) = 1)$$

$$\rho_6[n](i, l, g, l', g') \Leftrightarrow l(i) = 6 \wedge g' = g\{i := 2\} \wedge l' = l\{i := 7\}$$

$$\rho_{7a}[n](i, l, g, l', g') \Leftrightarrow l(i) = 7 \wedge \phi_7[n](l, g) \wedge g' = g \wedge l' = l\{i := 8\}$$

$$\rho_{7b}[n](i, l, g, l', g') \Leftrightarrow l(i) = 7 \wedge \neg \phi_7[n](l, g) \wedge g' = g \wedge l' = l$$

$$\phi_7[n](l, g) \Leftrightarrow (\exists j \in N_n)(g(j) = 4)$$

$$\rho_8[n](i, l, g, l', g') \Leftrightarrow z(i) = 8 \wedge g' = g\{i := 4\} \wedge l' = l\{i := 9\}$$

$$\rho_{9a}[n](i, l, g, l', g') \Leftrightarrow l(i) = 9 \wedge \phi_9[n](l, g) \wedge g' = g \wedge l' = l\{i := 10\}$$

$$\rho_{9b}[n](i, l, g, l', g') \Leftrightarrow l(i) = 9 \wedge \neg \phi_9[n](l, g) \wedge g' = g \wedge l' = l$$

$$\phi_9[n](i, l, g) \Leftrightarrow (\forall j \in N_i)(g(j) \in \{0, 1\})$$

$$\rho_{10}[n](i, l, g, l', g') \Leftrightarrow l(i) = 10 \wedge g' = g \wedge l' = l\{i := 11\}$$

$$\rho_{11a}[n](i, l, g, l', g') \Leftrightarrow l(i) = 11 \wedge \phi_{11}[n](l, g) \wedge g' = g \wedge l' = l\{i := 12\}$$

$$\rho_{11b}[n](i, l, g, l', g') \Leftrightarrow l(i) = 11 \wedge \neg \phi_{11}[n](l, g) \wedge g' = g \wedge l' = l$$

$$\phi_{11}[n](i, l, g) \Leftrightarrow (\forall j \in N_n - N_{i+1})(g(j) \neq 2 \wedge g(j) \neq 3)$$

$$\Leftrightarrow (\forall j \in N_n)(i < j \Rightarrow g(j) \in \{0, 1, 4\})$$

$$\rho_{12}[n](i, l, g, l', g') \Leftrightarrow l(i) = 12 \wedge g' = g\{i := 0\} \wedge l' = l\{i := 1\}$$

Representation of the transition relation in the Boyer-Moore logic is a direct translation of the above definition. For example, letting lp represent l' , etc., the components 2, and 3b are given by:

```
(defn rhoi2 (n i l g lp gp)
  (and (at 1 i 2)
        (equal lp (move 1 i 3))
        (equal gp (move g i 1))))
(defn rhoi3b (n i l g lp gp)
  (and (at 1 i 3)
        (equal gp g)
        (equal lp l)
        (exist-union g n '(3 4))))
```

A key property of the transition model is that each transition involves only one process. Since the flag for each process can only be modified by that process we have the following useful lemma.

Lemma (rho!):

$$Ws[n](l, g) \wedge j, k \in N_n \wedge \rho[n](k, l, g, l', g') \wedge i \neq k \Rightarrow Z(j) = l'(j) \wedge g(j) = g'(j)$$

This lemma is conveyed to the Boyer-Moore theorem prover using the `prove-lemma` command as follows. To facilitate use of the lemma as a rewriting rule the two conjuncts of the conclusions are presented as separate lemmas.

```
(prove-lemma l-rholemma (rewrite)
  (implies (and (ws n l g)
                 (member j (nset n))
                 (member k (nset n))
                 (rhoi n k l g lp gp)
                 (not (equal k j)))
            (equal (nth 1 j) (nth lp j)))))
(prove-lemma g-rholemma (rewrite)
  (implies (and (ws n l g)
                 (member j (nset n))
                 (member k (nset n))
                 (rhoi n k l g lp gp)
                 (not (equal k j)))
            (equal (nth g j) (nth gp j)))))
```

2.3. Invariants-Atomic Case

We say that a state is accessible if it can be reached from the initial state by a finite number (possibly zero) of transitions. The goal is to prove that in any accessible state at most one process has location 10.

$$(\forall i, j \in N_n)(l(i) = 10 \wedge i \neq j \Rightarrow l(j) \neq 10)$$

To do this we prove a much stronger invariant composed of three parts. The first part says that transitions preserve the property $Ws[n](l, g)$ of being an n -state. In a strongly typed world this is a consequence of typing (and can't even be directly expressed), but in most theorem provers it will be necessary to verify something depending on the representation of the sets $Locs$, $Flgs$, N_n and the finite maps from N_n to $Locs$ and $Flgs$.

In the second part the possible flag values at each program point are analyzed. This is expressed by the invariant Lg .

Definition (Lg):

$$\begin{aligned}
Lg[n](l, g) &\Leftrightarrow (\forall i \in N_n) Lgi(i, l, g) \\
Lgi(i, l, g) &\Leftrightarrow l(i) = 0 \wedge g(i) = OV \\
&\quad Z(i) = 1 \wedge g(i) = 0 \vee \\
&\quad Z(i) = 2 \wedge g(i) = 0 \vee \\
&\quad Z(i) = 3 \wedge g(i) = 1 \vee \\
&\quad Z(i) = 4 \wedge g(i) = 1 \vee \\
&\quad l(i) = 5 \wedge g(i) = 3 \vee \\
&\quad Z(i) = 6 \wedge g(i) = 3 \vee \\
&\quad l(i) = 7 \wedge g(i) = 2 \vee \\
&\quad l(i) = 8 \wedge g(i) = 2 \vee \\
&\quad l(i) = 8 \wedge g(i) = 3 \vee \\
&\quad l(i) = 9 \wedge g(i) = 4 \vee \\
&\quad l(i) = 10 \wedge g(i) = 4 \vee \\
&\quad l(i) = 11 \wedge g(i) = 4 \vee \\
&\quad l(i) = 12 \wedge g(i) = 4
\end{aligned}$$

This definition has a direct representation in the Boyer-Moore logic. The third part contains the main invariants which refine the mutex constraints.

Definition (Atomic Invariants):

$$\begin{aligned}
A_0[n](l, g) &\Leftrightarrow (\exists i \in N_n)(l(i) \in \{8, \dots, 12\}) \Rightarrow (\forall i \in N_n)(l(i) \neq 4) \\
A_1[n](l, g) &\Leftrightarrow (\exists i \in N_n)(l(i) \in (8, \dots, 12)) \Rightarrow (\exists i \in N_n)(l(i) \in (8, \dots, 12) \wedge g(i) \in \{3, 4\}) \\
A_2[n](l, g) &\Leftrightarrow (\forall i \in N_n)(\forall k \in N_i)(l(i) \in \{10, 11, 12\} \Rightarrow l(k) \notin \{5, \dots, 12\}) \\
A_3[n](l, g) &\Leftrightarrow (\forall i, k \in N_n)(l(i) = 12 \wedge Z(k) \in (5, \dots, 12) \Rightarrow g(k) = 4)
\end{aligned}$$

From A_2 we conclude the desired mutex property.

$$(\forall i, k \in N_n)(l(i) = 10 \wedge Z(k) = 10 \Rightarrow k = i)$$

The representation of these invariants in the Boyer-Moore logic is given by the following definitions, where the invariants with universal quantifiers are reduced to predicates with free-variables (which are implicitly universally quantified in the Boyer-Moore logic). To represent A_0 we note that $A_0[n](l, g)$ is logically equivalent to

$$(\forall k \in N_n)((\exists i \in N_n)(l(i) \in (8, \dots, 12)) \Rightarrow Z(k) \neq 4).$$

The universal quantifier is represented by the free variable k and the existential quantifier is expressed by the predicate (exist-union 1 n ' (8 9 10 11 12)) which is defined by recursion.

```

(defn a0 (n 1 k)
  (implies (and (member k (nset n))
                (exist-union 1 n '(8 9 10 11 12)))
            (hot (at 1 k 4))))

```

To represent A_1 , the two existential quantifiers are expressed by predicates defined by recursion.

```

(defn a1 (n 1 g)
  (implies (exist-union 1 n '(8 9 10 11 12))
            (exist-intersect-8-12-3-4 n 1 g)))

```

If $(Z(i) \in \{10, 11, 12\} \Rightarrow Z(k) \notin (5, \dots, 12))$ is $\Psi(i, k, l)$, then

$$A_2[n](l, g) \Leftrightarrow (\forall i \in N_n)(\forall k \in N_i)\Psi(i, k).$$

`a2-at-n1-n2 (n1 n2 1)` below expresses $\Psi(n1, n2, l)$. We obtain A_2 from `a2-at-n1-n2` via `a2-at-n2` by two recursions on `n1` and `n2`, each of which corresponds to a universal quantifier.

```

(defn a2-at-n1-n2 (n1 n2 1)
  (if (union-at-n 1 n1 '(10 11 12))
      (not (union-at-n 1 n2 '(5 6 7 8 9 10 11 12))) T))
(defn a2-at-n2 (n1 n2 1)
  (if (zerop n2) T
      (if (not (lessp n2 n1))
          (a2-at-n2 n1 (sub1 n2) 1)
          (and (a2-at-n1-n2 n1 n2 1)
                (a2-at-n2 n1 (sub1 n2) 1))))))
(defn a2 (n1 n2 1)
  (if (zerop n1) T
      (and (a2-at-n2 n1 n2 1)
            (a2 (sub1 n1) n2 1))))

```

Similarly we obtain A_3 .

```

(defn a3-at-n1-n2 (n1 n2 1 g)
  (if (and (at 1 n1 12)
            (union-at-n 1 n2 '(5 6 7 8 9 10 11 12)))
      (at g n2 4) T))
(defn a3-at-n2 (n1 n2 1 g)
  (if (zerop n2) T
      (and (a3-at-n1-n2 n1 n2 1 g)
            (a3-at-n2 n1 (sub1 n2) 1 g))))
(defn a3 (n1 n2 1 g)
  (if (zerop n1) T
      (and (a3-at-n2 n1 n2 1 g)
            (a3 (sub1 n1) n2 1 g))))

```

2.4. Proving the Invariants-Atomic Case

According to the Manna-Pnueli proof rules, to prove invariance of some property $I[n](l, g)$ it suffices to prove

- (i) $I[n](l_{\text{init}}, g_{\text{init}})$, and
- (ii) $I[n](l, g) \wedge \mathcal{R}[n](l, g, l', g') \Rightarrow I[n](l', g')$

From the definition of $\mathcal{R}$ it is clear that it suffices to prove

(ii') $I[n](l, g) \wedge k \in N_n \wedge \rho[n](k, l, g, l', g') \Rightarrow I[n](l', g')$

For our case I is the conjunction of the three parts.

$$I[n](l, g) = Ws[n](l, g) \wedge Lg[n](l, g) \wedge A_0[n](l, g) \wedge A_1[n](l, g) \wedge A_2[n](l, g) \wedge A_3[n](l, g)$$

We focus on the proof of (ii'). We assume

(ws) $Ws[n](l, g)$

(lg) $Lg[n](l, g)$

(a0) $A_0[n](l, g)$

(a1) $A_1[n](l, g)$

(a2) $A_2[n](l, g)$

(a3) $A_3[n](l, g)$

(kh) $k \in N_n$

(rho) $\rho[n](k, l, g, l', g')$

and prove each of the conjuncts of $I[n](l', g')$. Since the Boyer-Moore logic is quantifier-free, we cannot directly express the quantified invariants in the hypothesis. We note that when proving a formula of the form $Q(l, l') \wedge (\forall j)P(j, l) \Rightarrow (\forall j)P(j, l')$ it suffices to prove $Q(l, l') \wedge P(j_1, l) \wedge \dots \wedge P(j_n, l) \Rightarrow P(j, l')$. Thus in for each quantified invariant we submit to the Boyer-Moore prover a suitable lemma of the latter form. In the case analyses for the main invariants we include a list of names of the main Boyer-Moore events for each case. The indentation indicates dependence in outline form—each lemma depends on those of lesser indentation listed just above it. Note that these events follow the informal outline quite closely.

Proving $Ws[n](l', g')$ The proof for Ws is straightforward. The corresponding Boyer-Moore event is

```
(prove-lemma rho-preserves-ws (rewrite)
  (implies (and (ws n l g)
    (member k (nset n))
    (rhoi n k l g lp gp))
    (ws n lp gp))
  ((use (lm-rho-preserves-ws))))
```

Proving $Lg[n](l', g')$ The proof for Lg is also relatively simple straightforward. The corresponding Boyer-Moore event is

```
(prove-lemma rho-preserves-lg (rewrite)
  (implies (and (ws n l g)
    (member k (nset n))
    (rhoi n k l g lp gp)
    (lg n l g))
    (lg n lp gp))
  ((disable rhoi0 rhoi1a rhoi1b rhoi2 rhoi3a rhoi3b rhoi4 rhoi5a rhoi5b
    rhoi6 rhoi7a rhoi7b rhoi8 rhoi9a rhoi9b rhoi10 rhoi11a rhoi11b
    rhoi12)
  (enable rhoi))))
```

Proving $A_0[n](l', g')$ We further assume

(a0h) $(\exists i \in N_n)(l'(i) \in (8, \dots, 12))$

and show $(\forall j \in N_n)(l'(j) \neq 4)$. Thus we assume

(jh) $j \in N_n$

and show $Z'(j) \neq 4$. Let i' be a witness for (a0h). Thus

(iph) $i' \in N_n \wedge l'(ip) \in (8, \dots, 12)$.

We consider two cases.

Case (i): $(\exists i \in N_n)(l(i) \in (8, \dots, 12))$. Then $j \neq k \Rightarrow Z'(j) \neq 4$ by (i), (rho!), and (a0). If $Z'(k) = 4$ then by definition of p $Z(k) = 3$ and $\neg\phi_3(g)$, i.e. $(\forall j \in N_n)(g(j) \in \{0, 1, 2\})$, which contradicts (al) and (i).

```
int-8-12-3-4-then-un34
day-lckd
j-eq-k-18-112-nonemp
j-neq-k-18-112-nonemp
18-112-nonemp
```

Case (ii): $(\forall i \in N_n l(i) \notin \{8, \dots, 12\})$. Then $i' = k$ by (rho!), (ih). By definition of p $Z(k) = 5$ and $\neg\phi_5$ or $Z(k) = 7$ and ϕ_7 . But $\phi_7 = (\exists i \in N_n)(g(i) = 4)$ contradicts (lg) and (ii) and $\neg\phi_5 \Rightarrow g(j) \neq 1 \Rightarrow l(j) \neq 4$ by (lg).

```
exist-18-12
ex-lp8-12-in-lp8-12
k-in-lp8-12
k-not-in-18-12-then-157
j-ex-18-12
k-in-157
ex-k-in-157
j-ex-18-12
ex-lp8-12-in-lp8-12
exist-18-12
cond-rhoi5
k-in-lp8-12
ex-cond-rhoi5
cond-rhoi7
k-in-lp8-12
ex-cond-rhoi7
ex-if4
15-only-lp8
lp4-then-un34
j-neq-k-j-not-in-lp4
j-eq-k-j-not-in-lp4
lp4-empty
l34-empty
18-112-empty
```

□_{a0}

The corresponding Boyer-Moore event is

```
(prove-lemma rho-preserves-a0 ())
  (implies (and (ws n 1 g)
    (member j (nset n))
    (member k (nset n))
    (rhoi n k 1 g lp gp)
```

```

      (lg n 1 g)
      (a0 n 1 j)
      (a1 n 1 g))
    (a0 n lp j))
  ((use (18-112-nonemp) (18-112-empty))))

```

Proving $A_1[n](l', g')$ We further assume

(alh) $(\exists i \in N_n)(l'(i) \in (8, \dots, 12))$

and show $(\exists j \in N_n)(l'(j) \in (8, \dots, 12) \wedge g'(j) \in \{3, 4\})$. Thus we want to find some j' such that $(l'(j') \in (8, \dots, 12) \wedge g'(j') \in \{3, 4\})$. Let i' be a witness for (alh). Thus

(iph) $i' \in N_n \wedge l'(i') \in (8, \dots, 12)$

Again we consider two cases.

Case (i): $(\exists i \in N_n)(l(i) \in (8, \dots, 12))$. Then by (al) we can choose $j \in N_n$ such that $l(j) \in \{8, \dots, 12\} \wedge g(j) \in \{3, 4\}$. If $j \neq k$ let $j' = j$ and we are done by (rho!).

```

      int-wtn
      intersect-8-12-3-4-then-8-12
      intersect-8-12-3-4-then-3-4
      un8-12-and-un34-then-int
      int-k-not-ex-int
      intersect-8-12-3-4-then-8-12
      al-k-not-in-18-12-nep-18-12

```

If $j = k$ and $Z(k) \in (8, \dots, 11)$ let $j' = j$ and we are done by definition of p .

```

      if4
      lp4-then-un34
      lm-al-k-in-18-ii-nep-18-12
      un8-12-and-un34-then-int
      int-wtn
      al-k-in-18-11-nep-18-12

```

If $j = k$ and $Z(k) = 12$ then $Z'(k) = 1$ and $k \neq i'$ and $g(i') = g'(i') = 4$ by (a3) and (rho!).

```

      k-in-lp9-12-or-lp8
      un8-11-then-un5-12
      lp9-12-k-in-18-11
      k-in-lp9-12-then-15-12
      un57-then-un5-12
      lp8-k-in-157
      k-in-lp8-then-15-12
      k-in-15-12
      un8-12-then-un5-12
      k-neq-ex-lp8-12-in-15-12
      ex-lp8-12-then-15-12
      ex-lp8-12-in-gp4
      ex-lp8-12-not-in-lp0
      k-in-lp0
      k-not-ex-lp8-12
      lp4-then-un34
      ex-lp8-12-in-lp8-12
      lm-al-k-in-112-nep-18-12
      un8-12-and-un34-then-int
      int-wtn

```

al-k-in-112-nep-18-12
lm1-a1-nep-18-12
a3-ex-a3-at-n1-n2
lm-al-nep-18-12
al-nep-18-12

Case (ii): $(\forall i \in N_n)(l(i) \notin (8, \dots, 12))$. Then $i' = k$ by (rho!) and (ih). By definition of p either $Z(k) = 5$ and $\neg\phi_5$ or $Z(k) = 7$ and ϕ_7 . $Z(k) \neq 7$ since ϕ_7 contradicts (lg) and (ii). Thus we take $j' = i'$ and note that $g(k) = g'(k) = 3$ by (lg) and definition of p .

gp-rhoi5
k-in-lp8-12
lg-15-g3
ex-gp-rhoi
ex-cond-rhoi7
ex-if4
k-in-157
gp3-then-un34
exist-18-12
k-in-gp34
ex-lp8-12-in-lp8-12
un8-12-and-un34-then-int
exist-18-12
int-wtn
al-ep-18-12
rho-preserves-al

□_{a1}

The corresponding Boyer-Moore event is

```
(prove-lemma rho-preserves-al 0
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rhoi n k 1 g lp gp)
    (lg n 1 g)
    (a1 n 1 g)
    (a3 n n 1 g))
    (a1 n lp gp))
  ((use (al-nep-18-12))
   (use (al-ep-18-12))))
```

Proving $A_2[n](l', g')$ We further assume

(a2h) $i, j \in N_n \wedge j < i \wedge Z'(i) \in \{10, 11, 12\}$

and show $Z'(j) \notin (5, \dots, 12)$. We consider cases according to the relation of i, j, k .

Case (i): $i, j \neq k$ follows by (a2) and (rho!).

lm-i-j-neq-k
a2-i-j-a2-at-n1-n2
i-j-neq-k

Case (ii): $i \neq k, j = k$. Then $Z(i) = Z'(i) \in \{10, 11, 12\}$ by (rho!), $Z(k) \notin (5, \dots, 12)$ by (a2), $l(k) \neq 4$ by (a0), and by definition of p we are done.

k-in-lp5-7-not-14-then-15-7
un5-7-then-un5-11
k-in-lp5-7-or-lp8-or-lp9-12
un57-then-un5-11

```

      lp8-k-in-157
      k-in-lp8-then-15-11
      lp9-12-k-in-18-11
      un8-11-then-un5-11
      k-in-lp9-12-then-15-11
      k-in-lp5-7-then-15-11
      k-in-15-11
      un5-11-then-un5-12
      k-not-in-14
      un10-12-then-un8-12
      k-not-in-lp5-12
      lml-i-neq-k-j-eq-k
      lm-i-neq-k-j-eq-k
      a2-i-j-a2-at-n1-n2
      i-neq-k-j-eq-k
      i-neq-k

```

Case (iii) : $j \neq k, i = k$. Then $Z(k) \in \{9, 10, 11\}$ by definition of p . If $Z(k) \in (10, 11\}$ we are done by (a2) and (rho!). If $Z(k) = 9$ then $\phi_9[n](i, l, g)$, i.e. $(\forall j \in N_i)(g(j) \in \{0, 1\})$ and we are done by (lg).

```

      phi9-j-in-g01
      if1
      case-k-in-phi9
      un10-11-then-un10-12
      case-k-in-110-11
      k-in-110-11-or-phi9
      lml-i-eq-k-j-neq-k
      lm-i-eq-k-j-neq-k
      a2-i-j-a2-at-n1-n2
      i-eq-k-j-neq-k
      i-eq-k
      rho-preserves-a2

```

□_{a2}

The corresponding Boyer-Moore event is

```

(prove-lemma rho-preserves-a2 ())
  (implies (and (ws n 1 g)
    (member k (nset n))
    (member i (nset n))
    (member j (nset n))
    (rhoi n k 1 g lp gp)
    (lessp j i)
    (lg n 1 g)
    (a0 n 1 k)
    (a2 n n 1))
    (a2-at-n1-n2 i j lp))
    ((use (i-neq-k) (i-eq-k))))

```

Proving $A_3[n](l', g')$ We further assume

(a3h) $i, j \in N_n \wedge Z'(i) = 12 \wedge Z'(j) \in (5, \dots, 12)$

and show $g'(j) = 4$. We consider cases according to the relation of i, j, k .

Case (i): $i, j \neq k$ follows by (a3), (rho!).

lm-a3-i-j-neq-k

a3-i-j-a3-at-n1-n2
a3-i-j-neq-k

Case (ii): $i = j$ follows by (lg).

if4
l12-then-un9-12
lm-a3-i-j-eq-k
a3-i-j-a3-at-n1-n2
a3-i-j-eq-k

Case (iii): $i \neq k, j = k$. Then $Z(k) \in \{4, 5, \dots, 11\}$ by definition of p and (a3h), and $Z(i) = Z'(i) = 12$ by (rho!). $Z(k) \neq 4$ by (a0) and $g(k) = 4$ by (a3). Thus $Z(k) \in \{9, 10, 11\}$ by (lg). $Z'(k) \in \{9, 10, 11, 12\}$ and $g'(k) = 4$ by definition of p .

un5-11-then-un5-12
k-in-15-ii-g4-then-19-11
lm-k-in-19-11
l12-then-un8-12
k-in-15-11
k-in-19-11
k-in-lp9-12
if4
lml-a3-i-neq-k-j-eq-k
lm-a3-i-neq-k-j-eq-k
a3-i-j-a3-at-n1-n2
a3-i-neq-k-j-eq-k
a3-i-neq-k

Case (iv): $j \neq k, i = k$. We need only show $g(j) = 4$. By (a3h) and definition of ρ , $l(i) = 11$ and $\phi_{11}[n](i, l, g)$, i.e. $(\forall j \in N_n - N_{i+1})(g(j) \neq 2 \wedge g(j) \neq 3)$. $l(j) = l'(j) \in \{2, \dots, 12\}$ by (rho!). If $Z(j) \in \{9, \dots, 12\}$ then $g(j) = 4$ by (lg). If $Z(j) \in \{5, \dots, 8\}$ then $g(j) \in \{2, 3\}$ so $j < i$, which contradicts (a2).

k-lt-j
phill-j-not-in-g23
lml-j-not-in-g23
l11-then-un10-12
cond-rhoi11
lm2-j-not-in-g23
a2-n-a2-at-n2
j-not-in-g23
if4
l5-12-eq-l5-8-or-l9-12
if3
j-in-g4
a3-j-in-l5-12
k-in-l11
lm1-a3-i-eq-k-j-neq-k
lm-a3-i-eq-k-j-neq-k
a3-i-j-a3-at-n1-n2
a3-i-eq-k-j-neq-k
a3-i-eq-k
rho-preserves-a3

□_{a3}

The corresponding Boyer-Moore event is
(prove-lemma rho-preserves-a3 ())

```

(implies (and (ws n 1 g)
              (member k (nset n))
              (member i (nset n))
              (member j (nset n))
              (rhoi n k 1 g lp gp)
              (lg n 1 g)
              (a0 n 1 k)
              (a2 n n 1)
              (a3 n n 1 g))
          (a3-at-n1-n2 i j lp gp))
((use (a3-i-neq-k) (a3-i-eq-k))))

```

3. Formalization of the Molecular Variant in the Boyer-Moore Logic

In the molecular case, we introduce an additional map h , called a counter map. If a condition at location $Z(i)$ given by a universal or an existential sentence, which is involved the states of all processes, then it is no longer evaluated in one transition. In each transition the truth of such a sentence is examined for only $h(i)$'s process. Thus an n -process state is a triple (l, g, h) where (l, g) is as for the atomic case, h maps N_n to N_{n+1} — $h(i)$ is the counter value of process i , and $Ws^m[n]$ is the set of n -process states.

$$Ws^m[n] = [N, \rightarrow Locs] \times [N, \rightarrow Flgs] \times [N_n \rightarrow N_{n+1}]$$

We fix the number of processes n , and we will use i, j, k to range over N_n , and $l, l', \dots$ to range over location maps, and $g, g', \dots$ to range over flag maps, and $h, h', \dots$ to range over counter maps.

3.1. Transition Relation-Molecular Case

The transition relation for molecular case corresponding to the mutex program is defined as follows.

Definition (transitions for molecular case):

$$\begin{aligned} \mathcal{R}^m[n](l, g, l', g') &\Leftrightarrow (\exists i \in N_n) \rho^m[n](i, l, g, h, l', g', h') \\ \rho^m[n](i, l, g, h, l', g', h') &\Leftrightarrow \bigvee_{c \in C} \rho_c^m[n](i, l, g, h, l', g', h') \end{aligned}$$

where $C = \{0, 1a, 1b, 2, 3a, 3b, 4, 5a, 5b, 5c, 6, 7a, 7b, 8, 9a, 9b, 10, 11a, 11b, 12\}$ and the component transitions $\rho_c^m[n](i, l, g, h, l', g', h')$ are defined by

$$\begin{aligned} \rho_0^m[n](i, l, g, h, l', g', h') &\Leftrightarrow l(i) = 0 \wedge g' = g \wedge h' = h \wedge l' = l \{i := 1\} \\ \rho_{1a}^m[n](i, l, g, h, l', g', h') &\Leftrightarrow Z(i) = 1 \wedge g' = g \wedge h' = h \wedge l' = l \\ \rho_{1b}^m[n](i, l, g, h, l', g', h') &\Leftrightarrow Z(i) = 1 \wedge g' = g \wedge h' = h \wedge l' = l \{i := 2\} \\ \rho_2^m[n](i, l, g, h, l', g', h') &\Leftrightarrow Z(i) = 2 \wedge g' = g \{i := 1\} \wedge h' = h \{i := 1\} \wedge l' = l \{i := 3\} \\ \rho_{3a}^m[n](i, l, g, h, l', g', h') &\Leftrightarrow Z(i) = 3 \wedge h(i) = n + 1 \wedge g' = g \wedge h' = h \wedge l' = l \{i := 4\} \\ \rho_{3b}^m[n](i, l, g, h, l', g', h') &\Leftrightarrow Z(i) = 3 \wedge h(i) < n + 1 \wedge g(h(i)) \in \{0, 1, 2\} \wedge g' = g \end{aligned}$$

$$\begin{aligned}
& \text{A } h' = h\{i := h(i) + 1\} \text{ A } l' = l \\
\rho_4^m[n](i, l, g, h, l', g', h') & \Leftrightarrow l(i) = 4 \text{ A } g' = g\{i := 3\} \text{ A } h' = h\{i := 1\} \text{ A } l' = l\{i := 5\} \\
\rho_{5a}^m[n](i, l, g, h, l', g', h') & \Leftrightarrow l(i) = 5 \text{ A } h(i) < n + 1 \text{ A } g(h(i)) = l \text{ A } g' = g \text{ A } h' = h \\
& \text{A } l' = l\{i := 6\} \\
\rho_{5b}^m[n](i, l, g, h, l', g', h') & \Leftrightarrow l(i) = 5 \text{ A } h(i) = n + 1 \text{ A } g' = g \text{ A } h' = h \text{ A } l' = l\{i := 8\} \\
\rho_{5c}^m[n](i, l, g, h, l', g', h') & \Leftrightarrow l(i) = 5 \text{ A } h(i) < n + 1 \text{ A } g(h(i)) \neq l \text{ A } g' = g \\
& \text{A } h' = h\{i := h(i) + 1\} \text{ A } l' = l \\
\rho_6^m[n](i, l, g, h, l', g', h') & \Leftrightarrow l(i) = 6 \text{ A } g' = g\{i := 2\} \text{ A } h' = h\{i := 1\} \text{ A } l' = l\{i := 7\} \\
\rho_{7a}^m[n](i, l, g, h, l', g', h') & \Leftrightarrow l(i) = 7 \text{ A } g(h(i)) = 4 \text{ A } g' = g \text{ A } h' = h \text{ A } l' = l\{i := 8\} \\
\rho_{7b}^m[n](i, l, g, h, l', g', h') & \Leftrightarrow l(i) = 7 \text{ A } g(h(i)) \neq 4 \text{ A } g' = g \\
& \text{A } h' = h\{i := (h(i) - 1 \bmod n) + 1\} \text{ A } l' = l \\
\rho_8^m[n](i, l, g, h, l', g', h') & \Leftrightarrow l(i) = 8 \text{ A } g' = g\{i := 4\} \text{ A } h' = h\{i := 1\} \text{ A } l' = l\{i := 9\} \\
\rho_{9a}^m[n](i, l, g, h, l', g', h') & \Leftrightarrow l(i) = 9 \text{ A } h(i) = i \text{ A } g' = g \text{ A } h' = h \text{ A } l' = l\{i := 10\} \\
\rho_{9b}^m[n](i, l, g, h, l', g', h') & \Leftrightarrow l(i) = 9 \text{ A } h(i) < i \text{ A } g(h(i)) \in \{0, 1\} \text{ A } g' = g \\
& \text{A } h' = h\{i := h(i) + 1\} \text{ A } l' = l \\
\rho_{10}^m[n](i, l, g, h, l', g', h') & \Leftrightarrow l(i) = 10 \text{ A } g' = g \text{ A } h' = h\{i := i + 1\} \text{ A } l' = l\{i := 11\} \\
\rho_{11a}^m[n](i, l, g, h, l', g', h') & \Leftrightarrow l(i) = 11 \text{ A } h(i) = n + 1 \text{ A } g' = g \text{ A } h' = h \text{ A } l' = l\{i := 12\} \\
\rho_{11b}^m[n](i, l, g, h, l', g', h') & \Leftrightarrow l(i) = 11 \text{ A } g(h(i)) \notin \{2, 3\} \text{ A } h(i) < n + 1 \text{ A } g' = g \\
& \text{A } h' = h\{i := h(i) + 1\} \text{ A } l' = l \\
\rho_{12}^m[n](i, l, g, h, l', g', h') & \Leftrightarrow l(i) = 12 \text{ A } g' = g\{i := 0\} \text{ A } h' = h \text{ A } l' = l\{i := 0\}
\end{aligned}$$

The lemma for molecular case corresponding to (rho!) is the following.

Lemma (mrho!):

$$\begin{aligned}
& Ws^m[n](l, g, h) \text{ A } i, \quad k \in N_n \text{ A } \rho^m[n](k, l, g, h, l', g', h') \text{ A } i \neq k \\
& \Rightarrow l(i) = l'(i) \text{ A } g(i) = g'(i) \text{ A } h(i) = h'(i)
\end{aligned}$$

3.2. Invariants-Molecular Case

As in the atomic case, the invariants decompose into three parts. The first part says that statehood is preserved, i.e. $Ws^m[n](l, g, h)$ is invariant. The second part is the analysis of possible flag values at each location and is identical to the atomic case. The main invariants refine those of the atomic case as follows.

$$\begin{aligned}
B_0^a[n](l, h) & \Leftrightarrow (\forall i, j \in N_n)(l(i) = 5 \text{ A } j < h(i) \Rightarrow l(j) \neq 4) \\
B_0^b[n](l, h) & \Leftrightarrow (\forall i, j \in N_n)(l(i) = 5 \text{ A } j < h(i) \text{ A } l(j) = 3 \Rightarrow h(j) \leq i) \\
B_1^a[n](l) & \Leftrightarrow (\forall i, j \in N_n)(l(i) \in (8, \dots, 12] \Rightarrow l(j) \neq 4)
\end{aligned}$$

$$\begin{aligned}
B_1^b[n](l, g, h) &\Leftrightarrow (\forall i, j \in N_n)(l(i) \in (8, \dots, 12) \wedge l(j) = 3 \Rightarrow \\
&\quad (\exists r \in N_n)(l(r) \in (8, \dots, 12) \wedge g(r) \in \{3, 4\} \wedge h(r) \leq i)) \\
B_1^c[n](l, g, h) &\Leftrightarrow \\
&\quad (\forall i \in N_n)(l(i) \in (8, \dots, 12) \wedge g(i) \notin \{3, 4\} \Rightarrow (h(i) \in N_n \wedge g(h(i)) = 4)) \\
B_1^d[n](l, h) &\Leftrightarrow (\forall i \in N_n)(l(i) = 7 \Rightarrow h(i) \in N_n) \\
B_2^a[n](l) &\Leftrightarrow (\forall i, j \in N_n)(j < i \wedge l(i) \in \{10, 11, 12\} \Rightarrow l(j) \notin (5, \dots, la)) \\
B_2^b[n](l, h) &\Leftrightarrow (\forall i, j \in N_n)(j < i \wedge l(i) = 9 \wedge j < h(i) \Rightarrow l(j) \notin (5, \dots, 12)) \\
B_3^a[n](l, g) &\Leftrightarrow (\forall i, j \in N_n)(l(i) = 12 \wedge l(j) \in (5, \dots, 12) \Rightarrow g(j) = 4) \\
B_3^b[n](l, g, h) &\Leftrightarrow (\forall i, j \in N_n)(l(i) = 11 \wedge j < h(i) \wedge l(j) \in \{5, \dots, 12\} \Rightarrow g(j) = 4)
\end{aligned}$$

As before, from B_2^a we conclude

$$(\forall i, k \in N_n)(l(i) = 10 \wedge l(k) = 10 \Rightarrow k = i).$$

We note that the invariants B_1^c and B_1^d turned out to be necessary, though they do not appear in [4]. B_1^c and B_1^d are used in the proofs of B_1^b and B_1^c respectively. The representation of these invariants in the Boyer-Moore logic is given by the following definitions.

```

(defn b0a (n l h i j)
  (implies (and (at l i 5)
                (lessp j (nth h i)))
            (not (at l j 4))))
(defn b0b (n l h i j)
  (implies (and (at l i 5)
                (lessp j (nth h i))
                (at l j 3))
            (not (lessp i (nth h j)))))
(defn b1a (l i j)
  (implies (union-at-n l i '(8 9 10 11 12))
            (not (at l j 4))))
(defn hint-8-12-3-4-at-n (n l g h j)
  (and (intersect-8-12-3-4-at-n n l g)
        (not (lessp n (nth h j)))))
(defn exist-hint-8-12-3-4 (n l g h j)
  (if (zerop n) F
      (if (hint-8-12-3-4-at-n n l g h j) n
          (exist-hint-8-12-3-4 (sub1 n) l g h j))))
(defn bib (n l g h i j)
  (implies (and (union-at-n l i '(8 9 10 11 12))
                (at l j 3))
            (exist-hint-8-12-3-4 n l g h j)))
(defn b1c (n l g h i)
  (implies (and (union-at-n l i '(8 9 10 11 12))
                (not (union-at-n g i '(3 4))))
            (and (member (nth h i) (nset n))
                  (at g (nth h i) 4))))
(defn bid (n l h i)
  (implies (at l i 7)
            (member (nth h i) (nset n))))

```

```

(defn b2a (l i j)
  (implies (and (lessp j i)
                (union-at-n 1 i '(10 11 12)))
            (not (union-at-n 1 j '(5 6 7 8 9 10 11 12)))))
(defn b2b (l h i j)
  (implies (and (lessp j i)
                (at 1 i 9)
                (lessp j (nth h i)))
            (not (union-at-n 1 j '(5 6 7 8 9 10 11 12)))))
(defn b3a (l g i j)
  (implies (and (at 1 i 12)
                (union-at-n 1 j '(5 6 7 8 9 10 11 12)))
            (at g j 4)))
(defn b3b (l g h i j)
  (implies (and (at 1 i 11)
                (lessp j (nth h i))
                (union-at-n 1 j '(5 6 7 8 9 10 11 12)))
            (at g j 4)))

```

3.3. Proving the invariants

As in the atomic case, to prove invariance of some property $I[n](l, g, h)$ it suffices to prove:

- (i) $I[n](l_{\text{init}}, g_{\text{init}}, h_{\text{init}})$, and
- (ii) $I[n](l, g) \wedge k \in N_n \wedge \rho^m[n](k, l, g, h, l', g', h') \Rightarrow I[n](l', g', h')$

For our case $I[n](l, g, h)$ is the following conjunction.

$$\begin{aligned}
 &Ws^m[n](l, g) \wedge Lg[n](l, g) \wedge B_0^a[n](l, h) \wedge B_0^b[n](l, h) \wedge \\
 &B_1^a[n](l) \wedge B_1^b[n](l, g, h) \wedge B_1^c[n](l, g, h) \wedge B_1^d[n](l, h) \wedge \\
 &B_2^a[n](l) \wedge B_2^b[n](l, h) \wedge B_3^a[n](l, g) \wedge B_3^b[n](l, g, h)
 \end{aligned}$$

We focus on the proof of (ii). Thus we assume

(molws) $Ws^m[n](l, g, h)$

(lg) $Lg[n](l, g)$

(b0a) $B_0^a[n](l, h)$

(b0b) $B_0^b[n](l, h)$

(b1a) $B_1^a[n](l)$

(b1b) $B_1^b[n](l, g, h)$

(b1c) $B_1^c[n](l, g, h)$

(b1d) $B_1^d[n](l, h)$

(b2a) $B_2^a[n](l)$

(b2b) $B_2^b[n](l, h)$

(b3a) $B_3^a[n](l, g)$

(b3b) $B_3^b[n](l, g, h)$

(kh) $k \in N_n$

(mrho) $\rho^m[n](k, l, g, h, l', g', h')$

and prove each of the conjuncts. The proofs for Ws^m and Lg are just as for the atomic case. As in the atomic case we replace the quantified invariants in the hypothesis by a suitable finite collection of instances. B_1^b is the hardest invariant to prove, since it involves an existential statement in a fundamental way.

Proving $B_0^a[n](l', h', i, j)$ We further assume

(b0ah) $l'(i) = 5 \wedge h'(i) > j$

and show $l'(j) \neq 4$. Thus we assume

(ih) $i \in N_n \wedge l'(i) = 5 \wedge h'(i) > j$.

(jh) $j \in N_n$.

We consider four cases according to the relation of i, j, k .

Case (i): $i = k, j = k$. Obvious.

b0a-i-j-eq-k

Case (ii): $i, j \neq k$. The case follows by (mrho!) and (b0a).

b0a-i-j-neq-k

Case (iii): $i \neq k, j = k$. Then $h'(i) = h(i) > k$ by definition of (b0ah) and (mrho!) and $l(i) = l'(i) = 5$ by (mrho!). $l(k) \neq 3$ or $h(k) \leq i$ by (b0b). Hence by definition of ρ^m , $l'(k) \neq 4$.

cond-lp4

not-l3-then-lp4

i-in-l5

lm-b0a-i-neq-k-j-eq-k

b0a-i-neq-k-j-eq-k

b0a-i-neq-k

Case (iv): $i = k, j \neq k$. We need only show $l(j) \neq 4$. Then $l'(k) = 5 \wedge h'(k) > j$ by (boah) and (mrho!). $l(k) \neq 4$ since otherwise $h'(k) \leq j$ by definition of ρ^m . Hence $l(k) = 5 \wedge h'(k) = h(k) + 1 \wedge g(h(k)) \neq 1$.

If $h(k) = j$, then $g(j) \neq 1$, hence $l(j) \neq 4$ by (lg).

If $h(k) > j$, then $l(j) \neq 4$ by (b0a).

b0a-if1

if1-nth-h-k

l5-not-g1

l5-nth-h-k-eq-j

l5-j-lt-nth-k

nth-k-lt-j-or-eq-j

lm-j-not-in-l4

cond-l5

j-not-in-l4

k-in-l5

lm-b0a-i-eq-k-j-neq-k

b0a-i-eq-k-j-neq-k

b0a-i-eq-k

□_{b0a}

The corresponding Boyer-Moore event is

```
(prove-lemma rho-preserves-boa 0
  (implies (and (molws n 1 g h)
    (member i (nset n))
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (lg n 1 g)
    (b0a n 1 h i j)
    (b0b n 1 h i j))
    (b0a n lp hp i j))
    ((use (b0a-i-neq-k) (b0a-i-eq-k)))))
```

Proving $B_0^b[n](l', h', i, j)$ We further assume

(b0bh) $Z'(i) = 5 \wedge h'(i) > j \wedge Z'(j) = 3$

and show $h'(j) \leq i$. Thus we assume

(ih) $i \in N_n \wedge Z'(i) = 5$.

(jh) $j \in N_n \wedge Z'(j) = 3$.

We consider four cases according to the relation of i, j, k .

Case (i): $i = k, j = k$. Obvious.

b0b-i-j-eq-k

Case (ii): $i, j \neq k$. The case follows by (mrho!) and (b0b).

b0b-i-j-neq-k

Case (iii): $i \neq k, j = k$. Then $Z(i) = Z'(i) = 5$ and $h'(i) = h(i) > k$ by definition of (b0ah) and (mrho!). Since $Z'(k) = 3$ by (b0bh), definition of ρ^m implies that either $Z(k) = 2$ or $Z(k) = 3$ holds.

If $Z(k) = 2$, then $h'(k) = 1 \leq i$, hence the claim holds.

k-in-12-imp

If $Z(k) = 3$, then $g(h(k)) < 3 \wedge h(k) \leq i \wedge h'(k) = h(k) + 1$. By (lg) $h(k) \neq i$ because $Z(i) = 5$. Now $h'(k) \leq i$ follows from $h(k) \leq i \wedge h(k) \neq i \wedge h'(k) = h(k) + 1$.

b0b-if3

lm-i-neq-h-k

h-k-g02

i-neq-h-k

n-k-leq-subl-i

lml-k-in-13-imp

lm-k-in-13-imp

cond-13

k-in-13-imp

bob-i-in-15

lm-b0b-i-neq-k-j-eq-k

b0b-i-neq-k-j-eq-k

b0b-i-neq-k

Case (iv): $i = k, j \neq k$. We need only $h(j) \leq k$. Then $Z(j) = Z'(j) = 3$ by (mrho!) and (b0bh). Since $Z(k) \neq 4$ by definition of ρ^m and $h'(k) > j$, $Z'(k) = 5$ implies $Z(k) = 5 \wedge g(h(k)) \neq 1 \wedge h(k) \leq n \wedge h'(k) = h(k) + 1 \wedge h'(k) > j$. By (lg), $g(h(k)) \neq 1$ and

$Z(j) = 3$ imply $h(k) \neq j$, and so $h(k) > j$. Finally $l(k) = 5 \wedge h(k) > j \wedge Z(j) = 3$ implies $h(j) \leq k$ by (b0b).

```

      b0b-if1
      lm-j-neq-h-k
      h-k-not-g1
      j-neq-h-k
      lml-j-in-13
      cond-15
      lm-j-in-13
      k-in-15
      j-in-13
      lm-b0b-i-eq-k-j-neq-k
      b0b-i-eq-k-j-neq-k
      b0b-i-j-eq-k
      bob-i-eq-k
      rho-preserves-bob
□b0b

```

The corresponding Boyer-Moore event is

```

(prove-lemma rho-preserves-bob ()
  (implies (and (molws n 1 g h)
    (member i (nset n))
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (lg n 1 g)
    (b0b n 1 h i j))
    (b0b n lp hp i j)))
  ((use (b0b-i-neq-k) (b0b-i-eq-k))))

```

Proving $B_1^a[n](l', i, j)$ We further assume

(blah) $Z'(i) \in (8, \dots, 12]$

and show $Z'(j) \neq 4$.

(ih) $i \in N_n \wedge Z'(i) \in (8, \dots, 12]$

We consider four cases according to the relation of i, j, k .

Case (i): $i = k, j = k$. Obvious.

bla-i-j-eq-k

Case (ii): $i, j \neq k$. The case follows by (mrho!) and (bla).

bla-i-j-neq-k

Case (iii): $i \neq k, j = k$. $Z(i) \in (8, \dots, 12]$ by (mrho!).

If $h(k) \neq n + 1$, then $Z'(k) \neq 4$ because of definition of ρ^m .

h-k-neq-addl-n

If $h(k) = n + 1$, then there is no r such that $h(k) = n + 1 \leq r$, hence by (blb), $Z(k) \neq 3$.
By definition of $\rho^m, l'(k) \neq 4$.

```

      lm-h-k-eq-addl-n-nex-hint
      h-k-eq-addl-n-nex-hint
      h-k-eq-addl-n-k-not-in-13
      not-13-then-not-lp4
      h-k-eq-addl-n

```

lm-bla-i-neq-k-j-eq-k
 bla-i-neq-k-j-eq-k
 bla-i-neq-k

Case (iv): $i = k, j \neq k$. We only need show $Z(j) \neq 4$. $Z'(k) \in (8, \dots, 12)$ by (blbh), hence there are three cases: Either $Z(k) \in (8, \dots, 12\}$, $Z(k) = 5$ or $Z(k) = 7$.

If $Z(k) \in (8, \dots, 12\}$, then $Z(j) \neq 4$ by (bla).

lp9-12-k-in-18-12
 k-in-lp9-12-then-j-not-14

If $Z(k) = 5$, then $h(k) = n + 1$. Hence $Z(k) = 5 \wedge h(k) > j$ implies $Z(j) \neq 4$ by (b0a).

15-j-1t-h-k
 k-in-15-then-j-not-14
 k-not-in-17-then-lp9-12-or-15
 k-in-not-17-imp
 lm-bla-i-eq-k-j-neq-k
 bla-i-eq-k-j-neq-k
 bla-i-eq-k
 mrho-preserves-bla

If $Z(k) = 7$, then $h(k) < n \wedge g(h(k)) = 4$ by definition of ρ^m . By (lg), $l(h(k)) \in \{9, 10, 11, 12\}$, hence by (bla), $Z(j) \neq 4$.

cond-17
 bla-if4
 k-in-17-imp
 lm-bla-i-eq-k-j-neq-k
 bla-i-eq-k-j-neq-k

□_{bla}

The corresponding Boyer-Moore event is

```
(prove-lemma mrho-preserves-bla ()
  (implies (and (molws n 1 g h)
    (member i (nset n))
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (b1d n 1 h i)
    (lg n 1 g)
    (b0a n 1 h i j)
    (b1a 1 i j)
    (b1a 1 (nth h i) j)
    (b1b n 1 g h i j))
    (b1a lp i j))
    ((use (bla-i-neq-k) (b1a-i-eq-k))))
```

Proving $B_1^b[n](l', g', h', i, j)$ We further assume

(blbh) $Z'(i) \in (8, \dots, 12\}$ $Z'(j) = 3$

and show $(\exists r \in N_n)(l'(r) \in (8, \dots, 12) \wedge g'(r) \in \{3, 4\} \wedge h'(j) \leq r)$.

(ih) $i \in N_n \wedge l'(i) \in \{8, \dots, 12\}$.

(jh) $j \in N_n \wedge Z'(j) = 3$.

We consider four cases according to the relation of i, j, k .

Case (i): $i = k, j = k$. Obvious.

bib-i-j-eq-k

Case (ii): $i, j \neq k$. Then $Z(i) \in (8, \dots, 12) \wedge Z(j) = 3$ by (blbh) and (mrho!). Hence by (blb), there exists r such that $l(r) \in (8, \dots, 12) \wedge g(r) \in \{3, 4\} \wedge h(j) \leq r$.

If $l(r) \neq 12$, then the same r satisfies $Z'(T) \in (8, \dots, 12) \wedge g'(r) \in \{3, 4\} \wedge h'(j) \leq r$ by (mrho!) and definition of ρ^m .

hint-wtn
ex-hint-not-in-112
ex-hint-l-g-h
i-neq-k-ex-hint-not-in-112
lml-bib-i-j-neq-k
j-neq-k-then-hp-eq-h
lm-blb-i-j-neq-k
blb-i-j-neq-k

If $l(r) = 12$, then by (b3a) $g(i) = 4$. Hence by (mrho!) $Z'(i) \in (8, \dots, 12) \wedge g'(i) \in \{3, 4\}$. Moreover $r \leq i$ by (b2a) since $l(r) \in \{10, 11, 12\} \wedge Z(i) \in (5, \dots, 12)$. Hence $h(j) \leq r \leq i$ by definition of r , which implies $h'(j) = h(j) \leq i$ by (mrho!). It shows that i satisfies $Z'(i) \in \{8, \dots, 12\} \wedge g'(i) \in \{3, 4\} \wedge h'(j) \leq i$.

k-neq-u-in-lp8-12
lm-i-neq-k-in-int-8-12-3-4
un8-12-then-un5-12
i-neq-k-in-int-8-12-3-4
l12-then-un10-12
un8-12-then-un5-12
ex-hint-l-g-h
h-j-leq-i
hint-wtn
i-neq-k-ex-hint-in-112

Case (iii): $i \neq k, j = k$. $Z(i) \in (8, \dots, 12)$ by (mrho!). Since (blb) implies $Z'(k) = 3$, either $Z(k) = 2$ or $Z(k) = 3$.

If $Z(k) = 2$, then $h'(k) = 1$. There are two cases:

If $Z(i) \in (8, \dots, 12) \wedge g(i) \in \{3, 4\}$, then by (mrho!), $Z'(i) \in (8, \dots, 12) \wedge g'(i) \in \{3, 4\}$.
Let $r = 2$.

hint-wtn
i-in-int-8-12-3-4
hp-k-leq-i
i-in-g34

If $Z(i) \in (8, \dots, 12) \wedge g(i) \notin \{3, 4\}$, then by (blc) $g(h(i)) = 4$. By (lg), $l(h(i)) \in \{8, \dots, 12\} \wedge g(h(i)) \in \{3, 4\}$. Since $h(i) \neq k$, $l'(h(i)) \in (8, \dots, 12) \wedge g'(h(i)) \in \{3, 4\}$. Hence let $r = h(i)$.

u-if4
bib-u-neq-k
lm-u-in-int-8-12-3-4
bla-if4
k-neq-u-in-lp8-12
lml-u-in-int-8-12-3-4
u-in-int-8-12-3-4
hint-wtn
hp-k-leq-i

h-i-in-g34-imp
 i-not-in-g34
 k-in-12
lp3-then-13-or-12
 lm-k-not-in-13
 k-not-in-13

If $Z(k) = 3$, then $g(h(k)) < 3 \wedge h'(k) = h(k) + 1$ by definition of ρ^m . On the other hand, by (blb) there exists r such that $l(r) \in (8, \dots, 12] \wedge g(r) \in \{3, 4\} \wedge h(k) \leq r$. Since $k \neq r$, by (mrho!) the same r satisfies $Z'(r) \in (8, \dots, 12] \wedge g'(r) \in \{3, 4\}$. Moreover $h(k) \neq r$ since $g(r) \in \{3, 4\} \wedge g(h(k)) < 3$. Combined $h'(k) = h(k) + 1 \wedge h(k) \leq r$, it implies that $h'(k) \leq r$. Hence the same r works.

ex-hint-not-in-g02
h-k-g02
 ex-hint-neq-h-k
 ex-hint-leq-h-k
 h-k-leq-subl-ex-hint
 lm-hp-k-leq-ex-l-g-h
 cond-13
 ex-cond-13
 hint-member
 hp-k-leq-ex-l-g-h
 ex-hint-in-18-12
 ex-hint-neq-k-in-13
 ex-hint-neq-k-imp
 ex-hint-l-g-h-in-int-8-12-3-4
 hint-wtn
 ex-l-g-h-k-in-13
 lm-k-in-13
 k-in-13
 bib-i-neq-k-j-eq-k

Case (iv): $i = k, j \neq k$. First of all, we have $Z(j) = 3$ by (mrho!). Since $Z'(k) \in \{8, \dots, 12\}$, there are three cases: Either $Z(k) \in \{8, 9, 10, 11\}$, $Z(k) = 5$ or $Z(k) = 7$.

If $Z(k) \in \{8, 9, 10, 11\}$, then by (blb) there exists r such that $l(r) \in (8, \dots, 12) \wedge g(r) \in \{3, 4\} \wedge h(j) \leq r$. This generates the following two cases:

If there exists r such that $l(r) \in \{8, 9, 10, 11\} \wedge g(r) \in \{3, 4\} \wedge h(j) \leq r$, then by definition of ρ^m , $Z'(r) \in (8, \dots, 12] \wedge g'(r) \in \{3, 4\} \wedge h(j) \leq r$. Since $j \neq k$, $h'(j) \leq r$ follows from (mrho!).

r-eq-k-18-II-k-in-lp8-12
un8-11-then-un8-12
 r-eq-k-18-II-k-in-lp8-12
18-11-k-in-lp8-12
18-11-k-in-gp34
 hint-in-18-11
 ex-hint-in-18-12
 case-k
 ex-hint-not-in-112

If there exists r such that $l(r) = 12 \wedge g(r) \in \{3, 4\} \wedge h(j) \leq r$, then $l'(r) = (8, \dots, 12] \wedge g'(r) \in \{3, 4\} \wedge h'(j) \leq r$ by (mrho!) since $r \neq k$ and $j \neq k$.

r-neq-k
 ex-hint-neq-k-imp

ex-hint-in-112
 ex-hint-in-int-8-12-3-4-18-11
 ex-hint-l-g-h
 hint-wtn
 ex-hint-wtn-18-11
un8-11-then-un8-12
 k-in-18-II-imp
m-lp9-12-k-in-18-11
 k-in-lp9-12-imp

If $Z(k) = 5$, then by definition of ρ^m , $h(k) = n + 1 > j$. By (b0b) $h(j) \leq k$. Since $j \neq k$, by (mrho!) $h'(j) \leq k$. Moreover $g'(k) = 3$ by (lg) and definition of ρ^m . Hence let $r = k$.

lm-lp8-then-k-in-g34
lp8-then-k-in-g34
lp9-12-then-k-in-g34
un8-12-then-18-or-19-12
 lm-k-in-g34
 mrho-preserves-lg
 k-in-g34
 k-in-int
15-j-lt-h-k
 h-j-leq-k
 hint-wtn
 k-in-15-imp
 k-not-in-17-then-lp9-12-or-15
 k-not-in-17-imp

If $Z(k) = 7$, then by definition of ρ^m , $g(h(k)) = 4$. By (lg) $l(h(k)) \in \{9, 10, 11, 12\}$. Now by (b1b) there exists r such that $Z(r) \in (8, \dots, 12) \wedge g(r) \in \{3, 4\} \wedge h(j) \leq r$. Because $r \neq k$ and $j \neq k$, the same r satisfies $Z'(r) \in \{8, \dots, 12\} \wedge g'(r) \in \{3, 4\} \wedge h, '(j) \leq r$.

ex-hint-in-g34
 ex-hint-in-18-12
 ex-hint-neq-k-imp
 ex-hint-in-18-12
 ex-hint-neq-k-in-17
 ex-hint-in-int-8-12-3-4-17
 ex-hint-l-g-h
 hint-wtn
 ex-hint-wtn-17
 cond-17
 bla-if4
 lm-k-in-17-imp
 bib-k-in-17-imp
 lml-bib-i-eq-k-j-neq-k
 ex-hint-lp-gp-h-leq-h-j
 ex-hint-lp-gp-h-leq-hp-j
 ex-hint-lp-gp-h-in-int-8-12-3-4
 hint-wtn
 j-neq-k-then-hp-eq-h
 lm-bib-i-eq-k-j-neq-k
 bib-i-eq-k-j-neq-k
 blb-i-eq-k
 mrho-preserves-blb

□_{b1b}

The corresponding Boyer-Moore event is

```
(prove-lemma mrho-preserves-bib ())
  (implies (and (molws n l g h)
    (member i (nset n))
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k l g h lp gp hp)
    (b1d n l h k)
    (lg n l g)
    (b0b n l h i j)
    (b1b n l g h (nth h k) j)
    (b1b n l g h i j)
    (b1c n l g h i)
    (b2a 1 (exist-hint-8-12-3-4 n l g h j) i)
    (b3a 1 g
      (exist-hint-8-12-3-4 n l g h j) i))
    (b1b n lp gp hp i j))
  ((use (b1b-i-neq-k) (b1b-i-eq-k))))
```

Proving $B_1^c[n](l', g', h', i)$ We further assume

(b1ch) $Z'(i) \in (8, \dots, 12] \wedge g'(i) \notin \{3, 4\}$

and show $h'(i) \in N_n \wedge g'(h'(i)) = 4$.

(ih) $i \in N_n \wedge l'(i) \in (8, \dots, 12) \wedge g'(i) \notin \{3, 4\}$.

We consider four cases according to the relation of $i, h(i), k$.

Case (i): $i = k, h(i) = k$. By definition of ρ^m , $Z(k) = 7 \wedge g(h(i)) = g(k) = 4$. This contradicts (lg) since $g(k) = 4$ is equivalent to $Z(k) \in (9, \dots, 12]$.

```
contra-if4
lml-i-eq-k-then-h-k-neq-k
h-k-g4
lm-i-eq-k-then-h-k-neq-k
k-in-17
h-k-g4
i-eq-k-then-h-k-neq-k
```

Case (ii): $i \neq k, h(i) \neq k$. By (b1ch) and (mrho!), $Z(i) \in (8, \dots, 12] \wedge g(i) \notin \{3, 4\}$. (blc) implies $h(i) \in N$, hence $h'(i) \in N_n$ by (mrho!) since $i \neq k$. Similarly (blc) implies $g(h(i)) = 4$, hence $g'(h'(i)) = 4$ by (mrho!) since $i \neq k, h(i) \neq k$.

```
lm-blc-i-h-i-neq-k
blc-i-h-i-neq-k
```

Case (iii): $i \neq k, h(i) = k$. By (b1ch) and (mrho!), $Z(i) \in (8, \dots, 12) \wedge g(i) \notin \{3, 4\}$. (blc) implies $k \in N$. Since $i \neq k$ and (mrho!) imply $h'(i) = h(i) = k, h'(i) \in N$. Moreover (blc) implies $g(k) = 4$. On the other hand $Z(k) \neq 12$ by (b3a), hence $Z(k) \in \{9, 10, 11\}$ by (lg). By definition of ρ^m $g'(k) \in (9, \dots, 12]$, which, by (lg), is equivalent to $g'(k) = g'(h'(i)) = 4$.

```
contra-if4
19-11-then-in-lp9-12
k-in-lp9-12
if4
lm-k-not-in-112-imp
mrho-preserves-lg
k-not-in-112-imp
```

un8-12-then-un5-12
 not-g34-then-not-g4
 k-not-in-112
 lml-blc-i-neq-k-h-i-eq-k
 lm-blc-i-neq-k-h-i-eq-k
b3a-h-rhlemma
 blc-i-neq-k-h-i-eq-k
 blc-i-neq-k

Case (iv): $i = k, h(i) \neq k$. By (blch), $Z'(k) \in (8, \dots, 12) \wedge g'(k) \notin \{3, 4\}$. BY definition of ρ^m and (mrho!), $Z(k) = 7 \wedge g'(h(k)) = g(h(k)) = 4 \wedge h'(k) = h(k)$. Clearly $g'(h'(k)) = g'(h(k)) = 4$ follows. (bld) implies $h(k) \in N_n$, hence $h'(k) \in N_n$.

lp8-not-15-then-17
lp8-then-k-in-g34
lp8-not-g34-then-k-in-17
un8-12-then-18-or-19-12
lp9-12-then-k-in-g34
 lm-k-in-17
 mrho-preserves-lg
 k-in-17
 cond-17
 h-k-cond-17
 lm-h-k-g4
 k-in-17
 cond-17
 h-k-g4
 blc-i-eq-k-hp-k-neq-k
 blc-i-eq-k
 mrho-preserves-blc

□**b1c**

The corresponding Boyer-Moore event is

```

(prove-lemma mrho-preserves-blc ()
  (implies (and (molws n 1 g h)
    (member i (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (bld n 1 h k)
    (lg n 1 g)
    (b1c n 1 g h i)
    (b3a 1 g (nth h i) i))
    (b1c n lp gp hp i))
    ((use (b1c-i-neq-k))
     (use (b1c-i-eq-k)))))

```

Proving $B_1^d[n](l', h', i)$ We further assume

(bldh) $Z'(i) = 7$

and show $h'(i) \in N_n$.

(ih) $i \in N_n \wedge Z'(i) = 7$.

We consider cases according to the relation of i, k .

Case (i): $i \neq k$. Obvious.

17-th-i-neq-k

Case (ii): $i = k$. By (bldh) and definition of ρ^m , either $Z(k) = 6$ or $Z(k) = 7$.

If $Z(k) = 6$, then $h'(k) = 1 \in N$.

If $Z(k) = 7$, then by (bld) $h(k) \in N_n$ and by definition of ρ^m $h'(k) = (h(k) - 1 \bmod n) +$

1. By elementary number theory, $h'(k) \in N$.
 remainder-quotient
 lml-member-remainder
 lm-member-remainder
 member-remainder
 one-nset
 lm-bid-i-eq-k
 bld-i-eq-k
 bid-i-neq-k
 mrho-preserves-bld

□_{bld}

The corresponding Boyer-Moore event is

```
(prove-lemma mrhoi-preserves-bid (rewrite)
  (implies (and (molws n l g h)
    (member i (nset n))
    (member k (nset n))
    (mrhoi n k l g h lp gp hp)
    (bld n l h i))
    (bld n lp hp i))
  ((disable bld)
   (use (bid-i-neq-k))
   (use (bld-i-eq-k)))))
```

Proving $B_2^a[n](l', i, j)$ We further assume

(b2ah) $j < i \wedge Z'(i) \in \{10, 11, 12\}$

and show $Z'(j) \notin \{5, \dots, 12\}$. We consider cases according to the relation of i, j, k .

Case (i): $i, j \neq k$ follows by (b2a), (mrho!).

b2a-i-j-neq-k

Case (ii): $i \neq k, j = k$. Then $k < i \wedge Z(i) \in \{10, 11, 12\}$ by (mrho!). By (b2a) $l(k) \notin \{5, \dots, 12\}$. Since by (bla) $Z(k) \neq 4$, by definition of ρ^m $Z'(k) \notin \{5, \dots, 12\}$.

```
un57-then-un5-11
m-lp8-k-in-157
m-k-in-lp8-then-15-11
m-k-in-lp5-7-not-14-then-15-7
un5-7-then-un5-11
m-k-in-lp5-7-then-15-11
lp9-12-k-in-18-11
un8-11-then-un5-11
m-k-in-lp9-12-then-15-11
k-in-lp5-7-or-lp8-or-lp9-12
m-k-in-15-11
un10-12-then-un8-12
m-k-not-in-14
un5-11-then-un5-12
m-k-not-in-lp5-12
lm-b2a-i-neq-k-j-eq-k
b2a-i-neq-k-j-eq-k
```

b2a-i-neq-k

Case (iii): $j \neq k, i = k$. We need only show $Z(j) \notin \{5, \dots, 12\}$. By (b2ah) $j < k \wedge Z'(k) \in \{10, 11, 12\}$, hence by definition of ρ^m , either $Z(k) = 9$ or $Z(k) \in \{10, 11\}$.

If $Z(k) = 9$, then by definition of ρ^m $h(k) = k$. Since $j < k, j < h(k)$. By (b2b) $l(j) \notin \{5, \dots, 12\}$.

If $Z(k) \in \{10, 11\}$, then by (b2a) $Z(j) \notin \{5, \dots, 12\}$.

j-lt-h-k
lm-case-k-in-19
case-k-in-19
k-in-110-II-or-19
case-k-in-110-11
lm-b2a-i-eq-k-j-neq-k
b2a-i-eq-k-j-neq-k
b2a-i-eq-k
mrho-preserves-b2a

□_{b2a}

The corresponding Boyer-Moore event is

```
(prove-lemma mrho-preserves-b2a ()
  (implies (and (molws n 1 g h)
    (member i (nset n))
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (lessp j i)
    (lg n 1 g)
    (b1a 1 i j)
    (b2a 1 i j)
    (b2b 1 h i j))
    (b2a lp i j))
    ((use (b2a-i-neq-k) (b2a-i-eq-k)))))
```

Proving $B_2^b[n](l', h', i, j)$ We further assume

(b2bh) $j < i \wedge Z'(i) = 9 \wedge h'(i) > j$

and show $Z'(j) \notin \{5, \dots, 12\}$. We consider cases according to the relation of i, j, k .

Case (i): $i, j \neq k$ follows by (b2b), (mrho!).

b2b-i-j-neq-k

Case (ii): $i \neq k, j = k$. Then by (mrho!) $k < i \wedge Z(i) = 9 \wedge h(i) > k$. (b2b) implies $k \notin \{5, \dots, 12\}$. On the other hand (b1a) implies $Z(k) \neq 4$. by definition of ρ^m , $Z'(k) \notin \{5, \dots, 12\}$.

19-then-un8-12
m-k-not-in-14
not-k-in-15-12-imp
lm-b2b-i-neq-k-j-eq-k
b2b-i-j-neq-k
b2b-i-neq-k-j-eq-k
b2b-i-neq-k

Case (iii): $j \neq k, i = k$. We need only show $Z(j) \notin \{5, \dots, 12\}$. Since $Z'(k) = 9$ by definition of ρ^m , either $Z(k) = 8$ or $Z(k) = 9$.

If $Z(k) = 8$, then $h'(k) = 1$ by definition of ρ^m , and so it contradicts $h'(k) > j$.

If $Z(k) = 9$, then by definition of ρ^m , $g(h(k)) < 2 \wedge h'(k) = h(k) + 1$. Since $h'(k) > j$ by (b2bh), there are two cases:

If $h(k) > j$, then (b2b) implies $Z(j) \notin (5, \dots, 12\}$ because $j < k$ follows from (b2bh).

If $h(k) = j$, then $g(j) < 2$. By (lg) $Z(j) \notin (5, \dots, 12)$.

```

      if1
      lg-nth-h-k
      19-g01
      19-nth-h-k-eq-j
      nth-k-lt-j-or-eq-j
      lm-j-not-in-15-12
      cond-19
      j-not-in-15-12
      k-in-19
      1m-b2b-i-eq-k-j-neq-k
      b2b-i-eq-k-j-neq-k
      b2b-i-eq-k
      mrho-preserves-b2b

```

□_{b2b}

The corresponding Boyer-Moore event is

```

(prove-lemma mrho-preserves-b2b ()
  (implies (and (molws n 1 g h)
    (member i (nset n))
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (lessp j i)
    (lg n 1 g)
    (b1a 1 i j)
    (b2b 1 h i j))
    (b2b lp hp i j))
    ((use (b2b-i-neq-k) (b2b-i-eq-k)))))

```

Proving $B_3^a[n](l', g', i, j)$ We further assume

(b3ah) $i, j \in N_n \wedge l'(i) = 12 \wedge l'(j) \in (5, \dots, 12)$

and show $g'(j) = 4$. We consider cases according to the relation of i, j, k .

Case (i): $i, j \neq k$ follows by (b3a), (mrho!).

b3a-i-j-neq-k

Case (ii): $i = j$ follows from (lg).

```

      if4
      112-then-un9-12
      b3a-i-j-neq-k

```

Case (iii): $i \neq k, j = k$.

Then $Z(k) \in \{4, 5, \dots, 11\}$ by definition of ρ^m and (b3ah), and $Z(i) = Z'(i) = 12$ by (mrho!). Since $Z(k) \neq 4$ by (b1a) and $g(k) = 4$ by (b3a), $Z(k) \in \{9, 10, 11\}$ by (lg). By definition of ρ^m , $l'(k) \in \{9, 10, 11, 12\}$, and so $g'(k) = 4$ by (lg).

```

      un5-11-then-un5-12
      k-in-15-II-g4-then-19-11

```

```

lm-b3a-k-in-l9-11
l12-then-un8-12
m-k-in-15-11
b3a-k-in-l9-11
m-k-in-lp9-12
if4
lm-b3a-i-neq-k-j-eq-k
mrho-preserves-lg
b3a-i-neq-k-j-eq-k
b3a-i-neq-k-j-eq-k
b3a-i-neq-k

```

Case (iv): $j \neq k, i = k$. We need only show $g(j) = 4$. By (b3ah) and definition of ρ^m , $Z(k) = 11$ and $h(k) = n + 1 > j$. $Z(j) = Z'(j) \in (5, \dots, 12\}$ by (mrho!). By (b3b) $g(j) = 4$.

```

cond-lp12
b3a-j-in-l5-12
m-k-in-l11
lm-b3a-i-eq-k-j-neq-k
b3a-i-eq-k-j-neq-k
b3a-i-j-eq-k
b3a-i-eq-k
mrho-preserves-b3a

```

□_{b3a}

The corresponding Boyer-Moore event is

```

(prove-lemma mrho-preserves-b3a ()
  (implies (and (molws n l g h)
    (member i (nset n))
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k l g h lp gp hp)
    (lg n l g)
    (b1a l i j)
    (b3a l g i j)
    (b3b l g h i j))
    (b3a lp gp i j))
    ((use (b3a-i-neq-k) (b3a-i-eq-k)))))

```

Proving $B_3^b[n](l', g', h', i, j)$ We further assume

(b3bh) $i, j \in N_n \wedge Z'(i) = 11 \wedge h'(i) > j \wedge Z'(j) \in (5, \dots, 12\}$

and show $g'(j) = 4$. We consider cases according to the relation of i, j, k .

Case (i): $i, j \neq k$ follows by (b3b), (mrho!).

```
b3b-i-j-neq-k
```

Case (ii): $i = j$ follows from (lg).

```

if4
l11-then-un9-12
b3b-i-j-eq-k

```

Case (iii): $i \neq k, j = k$. Then $Z(i) = 11 \wedge h(i) > k$ by (mrho!). Since (bla) implies $Z(k) \neq 4$, by definition of ρ^m $Z(k) \in \{5, \dots, 11\}$. Hence by (b3b) $g(k) = 4$. It follows from (lg) that $Z(k) \in \{9, 10, 11\}$. Again by definition of ρ^m , $Z'(k) \in \{9, 10, 11, 12\}$, which is, by (lg), equivalent to $g'(k) = 4$.

```

un5-11-then-un5-12
k-in-15-II-g4-then-19-11
lm-b3b-k-in-19-11
l11-then-un8-12
m-k-in-15-11
b3b-k-in-19-11
m-k-in-lp9-12
if4
lm-b3b-i-neq-k-j-eq-k
mrho-preserves-lg
b3b-i-neq-k-j-eq-k
b3b-i-neq-k

```

Case (iv): $j \neq k$, $i = k$. We need only show $g(j) = 4$. By (b3bh) and definition of ρ^m , $l(j) \in \{5, \dots, 12\}$. Since $Z'(k) = 11$, by definition of ρ^m , either $Z(k) = 10$ or $Z(k) = 11$.

If $Z(k) = 10$, then by definition of ρ^m , $h'(k) = k + 1 > j$. Since $k \neq j$, $k > j$. This contradicts (b2a).

```

l10-then-un10-12
j-leq-addlk-then-k-not-in-110
not-j-leq-addlk-then-k-not-in-110
k-not-in-110

```

If $Z(k) = 11$, then by definition of ρ^m , $h'(k) = h(k) + 1 \wedge g(h(k)) \in \{0, 1, 4\}$. Since (b3bh) implies $h'(k) > j$, there are two cases:

If $h(k) = j$, then $g(j) \in \{0, 1, 4\} \wedge Z(j) \in \{5, \dots, 12\}$. It follows from (lg) that $g(j) = 4$.

```

if4
l5-12-eq-15-8-or-19-12
if3
j-in-g4
l11-g14
l11-nth-h-k-eq-j

```

If $h(k) > j$, then by (b3b) $g(j) = 4$.

```

nth-k-lt-j-or-eq-j
lm-j-in-15-12
cond-111
j-in-15-12
k-not-in-110
lp11-then-l11-or-l10
b3b-k-in-l11
lm-b3b-i-eq-k-j-neq-k
b3b-i-eq-k-j-neq-k
b3b-i-eq-k
mrho-preserves-b3b

```

□_{b3b}

The corresponding Boyer-Moore event is

```

(prove-lemma mrho-preserves-b3b ()
  (implies (and (molws n 1 g h)
    (member i (nset n))
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (lg n 1 g)

```

```

      (b1a 1 i j)
      (b3b 1 g h i j)
      (b2a 1 i j))
    (b3b 1p gp hp ij))
  ((use (b3b-i-neq-k) (b3b-i-eq-k))))

```

4. Remarks

The formalization of the Manna-Pnueli proof in the Boyer-Moore logic is fairly direct modulo small details of quantifier manipulation. The informal outline was developed prior to attempting to present the proof to the theorem prover. The resulting sequence of events follows the informal plan quite closely. The main difficulties encountered were in discovering the precise form of the lemmas and hints necessary for the theorem prover.

This experiment was carried out entirely in the basic quantifier free logic. Since many of the quantifiers involved are bounded (for example $(\forall i \in N,)$) it is possible that a more compact formalization could be obtained by using the bounded quantifier facilities of the latest version of the theorem prover. A further experiment would be to use the full quantifier extension of the prover developed by Kaufmann [3] to formalize and prove *Ziveness* properties of the algorithm, such as “a process waiting at l_9 to enter the critical section l_{10} will eventually do so”.

Acknowledgments

The authors would like to thank Amir Pnueli for the challenge that initiated this project, and Alexandre Bronstein for many helpful hints on using the Boyer-Moore theorem prover. This research was funded in part by DARPA contract N00039-84-C-0211 and by NSF grants CCR-8718605 and CCR-8917606.

5. References

- [1] Robert S. Boyer and J. Strother Moore. *A Computational Logic*. Academic Press, 1979.
- [2] Robert S. Boyer and J. Strother Moore. *A Computational Logic Handbook*. Academic Press, 1988.
- [3] Matt Kaufmann. An extension of the boyer-moore theorem prover to support first-order quantification. *Journal of Automated Reasoning*, to appear, 1991.
- [4] Zohar Manna and Amir Pnueli. An exercise in the verification of multi-process programs. In W. H. J. Feijen, A. J. M. van Gasteren, D. Gries, and J. Misra, editors, *Beauty is Our Business*, pages 289-301. Springer Verlag, 1990.

6. Appendix-Listing of Events

This appendix contains a listing of the input to the Boyer-Moore prover. The page-header is the event set name. Below is a list of the event-sets and a brief description of the contents. Comments in the input are signaled by one or more ;s.

Common Events

com.ev Definitions and lemmas common to atomic and molecular cases-manipulation of finite sets and arrays, flag invariants.

Atomic Case Events

defn.ev Definitions of transition relation and invariants.

basic.ev Properties of well-formed states are turned into rewrite rules. Several formulations of the rho! lemma are proved for use in different circumstances. Basic properties of the A-invariants are proved.

ws.ev Proof that transitions preserve the well-formedness invariant Ws .

lg.ev Proof that transitions preserve the flag invariant Lg .

a0.ev Proof that transitions preserve A_0 .

a1.ev Proof that transitions preserve A_1 .

a2.ev Proof that transitions preserve A_2 .

a3.ev Proof that transitions preserve A_3 .

Molecular Case Events

moldefn.ev Definitions of molecular transition relation and invariants.

molbasic.ev More properties of finite sets. Properties of well-formed states are turned into rewrite rules. Several formulations of the molecular rho! lemma are proved for use in different circumstances.

mollg.ev Proof that molecular transitions preserve the flag invariant Lg .

b0.ev Proof that molecular transitions preserve B_0^a, B_0^b .

b1.ev Proof that molecular transitions preserve $B_1^a, B_1^b, B_1^c, B_1^d$.

b2.ev Proof that molecular transitions preserve B_2^a, B_2^b .

b3.ev Proof that molecular transitions preserve B_3^a, B_3^b .

```

**sequence and finite set utilities

;;The ith entry in l.
(defn nth (l i)
  (if (listp l)
      (if (equal i 1) (car l) (nth (cdr l) (sub1 i)))
      (if (numberp l)
          (if (equal i 1) 1 F) F)))
(disable nth)

;;update ith entry of l to be k
(defn move (l i k)
  (if (equal i 0) l
      (if (nlistp l)
          (if (equal i 1) k l)
          (if (equal i 1)
              (cons k (cdr l))
              (cons (car l)
                    (move (cdr l) (sub1 i) k))))))
(disable move)

(defn at (l i k)
  (equal (nth l i) k))
(disable at)

(defn length (l)
  (if (listp l) (add1 (length (cdr l))) (zero)))
(disable length)

;;The nth entry in l is in the list i.
(defn union-at-n (l n i)
  (member (nth l n) i))
(disable union-at-n)

;;Any entry in l is in the list i.
(defn all-union (l n i)
  (if (zerop n) T
      (and (union-at-n l n i)
            (all-union l (sub1 n) i))))
(disable all-union)

;;There exists an entry in l which belongs to
;;the list i, moreover when exists, some such
;;j is returned.
(defn exist-union (l n i)
  (if (zerop n) F
      (if (union-at-n l n i) n
          (exist-union l (sub1 n) i))))
(disable exist-union)

;;n is in the intersection of l8-12 and g34.
(defn intersect-8-12-3-4-at-n (n l g)
  (and (union-at-n l n '(8 9 10 11 12))
        (union-at-n g n '(3 4))))
(disable intersect-8-12-3-4-at-n)

;;There exists n in the intersection of l8-12 and g34
(defn exist-intersect-8-12-3-4 (n l g)
  (if (zerop n) F
      (if (intersect-8-12-3-4-at-n n l g) n
          (exist-intersect-E-12-3-4 (sub1 n) l g))))
(disable exist-intersect-8-12-3-4)

**Flag invariant.
(defn lg-1-at-n (n l g)
  (or (and (at l n 0) (at g n 0))
      (and (at l n 1) (at g n 0))
      (and (at l n 2) (at g n 0))
      (and (at l n 3) (at g n 1))
      (and (at l n 4) (at g n 1))))
(disable lg-1-at-n)

(defn lg-2-at-n (n l g)
  (or (and (at l n 5) (at g n 3))
      (and (at l n 6) (at g n 3))
      (and (at l n 7) (at g n 2))
      (and (at l n 8) (at g n 3))
      (and (at l n 8) (at g n 2))))
(disable lg-2-at-n)

(defn lg-3-at-n (n l g)
  (or (and (at l n 9) (at g n 4))
      (and (at l n 10) (at g n 4))
      (and (at l n 11) (at g n 4))
      (and (at l n 12) (at g n 4))))
(disable lg-3-at-n)

(defn lg-at-n (n l g)
  (and (lg-1-at-n n l g)
        (lg-2-at-n n l g)
        (lg-3-at-n n l g)))
(disable lg-at-n)

(defn lg (n l g)
  (if (zerop n) T
      (and (lg-at-n n l g)
            (lg (sub1 n) l g))))

    (and (lg-at-n n l g)
          (lg (sub1 n) l g)))
(disable lg)

**The set {1...n}.
(defn nset (n)
  (if (zerop n) NIL
      (cons n (nset (sub1 n)))))
(disable nset)

;;n belongs to nset.
(prove-lemma n-in-nset (rewrite)
  (implies (not (zerop n))
            (member n (nset n))))
(enable nset)

;;Any element in nset is a number.
(prove-lemma nset-number (rewrite)
  (implies (member k (nset n))
            (numberp k)))
(enable nset)

;;If a nonzero number plus one belongs to nset,
;;then so does the nonzero number itself.
(prove-lemma add1-nset (rewrite)
  (implies (and (not (zerop k))
                 (member (add1 k) (nset n)))
            (member k (nset n))))
(enable nset)

;;Any list has its length at least nonzero.
(prove-lemma list-ln (rewrite)
  (implies (listp l)
            (not (equal (length l) 0))))
(enable length)

;;(move l k i) is again a list if l is a list.
(prove-lemma move-is-list (rewrite)
  (implies (listp l)
            (listp (move l k i))))
(enable move)

(enable length)

;;(move l k i) has i as its kth entry.
;;(enable length) is critical to prove this lemma.
(prove-lemma move-nth (rewrite)
  (implies (and (listp l)
                 (member k (nset (length l))))
            (equal (nth (move l k i) k) i)))
(enable nth move nset)

(prove-lemma zero-not-member-nset (rewrite)
  (not (member 0 (nset n))))
(enable nset)

;;Lists l and (move l k i) have the same length.
(prove-lemma move-unchange-length (rewrite)
  (implies (and (listp l)
                 (member k (nset (length l))))
            (equal (length (move l k i)) (length l))))
(enable move nset)

;;Lists l and (move l k i) have the same entries
;;except kth one.
(prove-lemma move-unchange-other-than-nth (rewrite)
  (implies (and (listp l)
                 (member k (nset (length l)))
                 (not (equal j k)))
            (equal (nth (move l k i) j) (nth l j))))
(enable move nth nset)

(prove-lemma member-ex-union (rewrite)
  (implies (exist-union l n i)
            (member (exist-union l n i) (nset n))))
(enable nset exist-union)

;;(exist-union l n i) is a number.
(prove-lemma number-ex-union (rewrite)
  (implies (exist-union l n i)
            (numberp (exist-union l n i))))
(enable exist-union)

;;(exist-intersect-E-12-3-4 n l g) belongs to nset.
(prove-lemma member-intersect (rewrite)
  (implies (exist-intersect-E-12-3-4 n l g)
            (member (exist-intersect-E-12-3-4 n l g) (nset n))))
(enable nset exist-intersect-8-12-3-4 intersect-8-12-3-4-at-n)

;;(exist-intersect-E-12-3-4 n l g) is a number.
(prove-lemma number-intersect (rewrite)
  (implies (exist-intersect-E-12-3-4 n l g)
            (numberp (exist-intersect-E-12-3-4 n l g))))
(enable exist-intersect-8-12-3-4)

;;any member of nset is nonzero.

```

```

(prove-lemma k-not-0 (rewrite)
  (implies (member k (nset n))
    (not (zerop k))))
(enable nset)))

;*lemmas for a0

;;;If j's entry in 1 is between 8..12 then
;;;(exist-union 1 n '(8 9 10 11 12)) holds.
(prove-lemma j-ex-18-12 (rewrite)
  (implies (and (member j (nset n))
    (union-at-n 1 j '(8 9 10 11 12)))
    (exist-union 1 n '(8 9 10 11 12)))
  ((enable nset exist-union union-at-n at))))

;;;Witness of (exist-union lp n '(8 9 10 11 12))
;;;has in lp its entry between 8..12.
(prove-lemma ex-lp8-12-in-lp8-12 (rewrite)
  (implies (exist-union lp n '(8 9 10 11 12))
    (union-at-n lp (exist-union lp n
      '(8 9 10 11 12)) '(8 9 10 11 12)))
  ((enable exist-union union-at-n at))))

;;;If (not (exist-union 1 n '(8 9 10 11 12)))
;;;holds, then (not (exist-union g n '(4))) by lg.
(prove-lemma ex-if4 (rewrite)
  (implies (and (not (exist-union 1 n '(8 9 10 11 12)))
    (lg n 1 g))
    (not (exist-union g n '(4))))
  ((enable exist-union union-at-n lg
    lg-at-n lg-2-at-n at))))

;;;If (not (exist-union g n '(1))) holds,
;;;then there is no entry either 3 or 4.
(prove-lemma l34-empty (rewrite)
  (implies (and (member j (nset n))
    (lg n 1 g))
    (not (exist-union g n '(1))))
  (not (union-at-n 1 j '(3 4))))
  ((enable at nset exist-union union-at-n lg
    lg-at-n lg-1-at-n at))))

;;;If j's entry in lp is 4, then (certainly)
;;;it is either 3 or 4.
(prove-lemma lp4-then-un34 (rewrite)
  (implies (at lp j 4)
    (union-at-n lp j '(3 4)))
  ((enable union-at-n at))))

;;;If (exist-intersect-E-12-3-4 n 1 g) holds,
;;;then so does (exist-union g n '(3 4)).
(prove-lemma int-8-12-3-4-then-un34 (rewrite)
  (implies (exist-intersect-E-12-3-4 n 1 g)
    (exist-union g n '(3 4)))
  ((enable exist-intersect-E-12-3-4
    intersect-8-12-3-4-at-n
    union-at-n exist-union at))))

;*lemmas for a1

;;;i is the witness of
;;;(exist-intersect-E-12-3-4 n lp gp).
(prove-lemma int-wtn (rewrite)
  (implies (and (member j (nset n))
    (intersect-8-12-3-4-at-n j lp gp))
    (exist-intersect-E-12-3-4 n lp gp))
  ((enable nset exist-intersect-E-12-3-4)))

;;;If there exists j such that j's entry in lp
;;;is between 8..12 and entry in gp is either 3 or 4
;;;then (intersect-8-12-3-4-at-n j lp gp) holds.
(prove-lemma un8-12-and-un34-then-int (rewrite)
  (implies (and (union-at-n lp j '(8 9 10 11 12))
    (union-at-n gp j '(3 4)))
    (intersect-8-12-3-4-at-n j lp gp))
  ((enable intersect-8-12-3-4-at-n)))

;;;By the two lemmas above,
;;;(exist-intersect-E-12-3-4 n lp gp) holds provided
;;;that there exists j such that j's entry in lp is
;;;between 8..12 and entry in gp is either 3 or 4.

:* ep-18-12

;;;If the k's entry in 1 is 5, then the k's entry
;;;in g is 3 by lg.
(prove-lemma lg-15-g3 (rewrite)
  (implies (and (member k (nset n))
    (lg n 1 g))
    (at k 5))
  (at g k 3))
  ((enable lg lg-at-n lg-2-at-n nset at)))

;;;If the k's entry in gp is 3 then certainly
;;;it is either 3 or 4.
(prove-lemma gp3-then-un34 (rewrite)
  (implies (at gp k 3)
    (union-at-n gp k '(3 4))))

```

```

((enable union-at-n at)))

;;;nep-18-12

;;;If the k's entry in 1 is between 8..12 then
;;;it is either between 8..11 or equal to 12.
(prove-lemma case-k (rewrite)
  (implies (and (union-at-n 1 k '(8 9 10 11 12))
    (not (union-at-n 1 k '(8 9 10 11))))
    (at 1 k 12))
  ((enable union-at-n at)))

;;;k-not-18-12

;;;If (exist-intersect-E-12-3-4 n 1 g) holds
;;;then the witness has its entry in g either equal
;;;to 3 or 4.
(prove-lemma intersect-8-12-3-4-then-3-4 (rewrite)
  (implies (exist-intersect-E-12-3-4 n 1 g)
    (union-at-n g
      (exist-intersect-E-12-3-4 n 1 g) '(3 4)))
  ((enable exist-intersect-E-12-3-4
    intersect-8-12-3-4-at-n
    union-at-n at)))

;;;If (exist-intersect-E-12-3-4 n 1 g) holds,
;;;then the witness has its entry in g between 8 and 12.
(prove-lemma intersect-8-12-3-4-then-8-12 (rewrite)
  (implies (exist-intersect-E-12-3-4 n 1 g)
    (union-at-n 1 (exist-intersect-E-12-3-4 n 1 g)
      '(8 9 10 11 12)))
  ((enable exist-intersect-E-12-3-4
    intersect-8-12-3-4-at-n
    union-at-n at)))

;;;k-in-18-11

;;;If k's entry in lp is between 9 and 12,
;;;then it is certainly between 8 and 12.
(prove-lemma un9-12-then-un8-12 (rewrite)
  (implies (union-at-n lp k '(9 10 11 12))
    (union-at-n lp k '(8 9 10 11 12)))
  ((enable union-at-n at)))

;;;If the i's entry in 1 is between 9 and 12,
;;;then the k's entry in g is 4.
(prove-lemma if4 (rewrite)
  (implies (and (member j (nset n))
    (lg n 1 g))
    (union-at-n 1 j '(9 10 11 12)))
    (at g j 4))
  ((enable nset union-at-n at lg lg-at-n lg-3-at-n)))

;;;k-in-112

;;;If (exist-union lp n '(8 9 10 11 12)) holds then
;;;its witness does not have its entry in lp equal to 1.
(prove-lemma ex-lp8-12-not-in-lp0 (rewrite)
  (implies (exist-union lp n '(8 9 10 11 12))
    (not (at lp (exist-union lp n '(8 9 10 11 12)) 0)))
  ((enable exist-union union-at-n at)))

;;;If k's entry in lp is between 8 and 12,
;;;then it is either between 8 and 11 or 12.
(prove-lemma k-in-lp9-12-or-lp8 (rewrite)
  (implies (and (union-at-n lp k '(8 9 10 11 12))
    (not (union-at-n lp k '(9 10 11 12))))
    (at lp k 8))
  ((enable union-at-n at)))

;;;If the k's entry is either 5 or 7.
;;;then it is between 5 and 7.
(prove-lemma un57-then-un5-12 (rewrite)
  (implies (union-at-n 1 k '(5 7))
    (union-at-n 1 k '(5 6 7 8 9 10 11 12)))
  ((enable union-at-n at)))

;;;If the k's entry in 1 is between 8 and 11,
;;;then it is between 5 and 12.
(prove-lemma un8-11-then-un5-12 (rewrite)
  (implies (union-at-n 1 k '(8 9 10 11))
    (union-at-n 1 k '(5 6 7 8 9 10 11 12)))
  ((enable union-at-n at)))

;;;If the k's entry in 1 is between 8 and 12,
;;;then it is between 5 and 12.
(prove-lemma un8-12-then-un5-12 (rewrite)
  (implies (union-at-n 1 k '(8 9 10 11 12))
    (union-at-n 1 k '(5 6 7 8 9 10 11 12)))
  ((enable union-at-n at)))

;*lemmas for a2

;;;i-eq-k-j-neq-k

;;;If the k's entry in 1 is either 10 or 11,
;;;then the k's entry in 1 is between 10 and 12.

```

```

(prove-lemma un10-11-then-un10-12 (rewrite)
  (implies (union-at-n 1 k '(10 11))
    (union-at-n 1 k '(10 11 12)))
  ((enable union-at-n at)))

;;;If the j's entry in g is either 0 or 1 then
;;;the j's entry in 1 is not between 5 and 12.
(prove-lemma if1 (rewrite)
  (implies (and (member j (nset n))
    (lg n 1 g)
    (union-at-n g j '(0 1)))
    (not (union-at-n 1 j
      '(5 6 7 8 9 10 11 12))))
  ((enable nset union-at-n at lg lg-at-n
    lg-1-at-n)))

;;;j-eq-k-i-neq-k

;;;If the k's entry in 1 is between 5 and 7.
;;;then it is certainly between 5 and 12.
(prove-lemma un5-7-then-un5-11 (rewrite)
  (implies (union-at-n 1 k '(5 6 7))
    (union-at-n 1 k '(5 6 7 8 9 10 11)))
  ((enable union-at-n at nset)))

;;;If the k's entry in lp is between 5 and 7 then
;;;it is certain between 5 and 11.
(prove-lemma un57-then-un5-11 (rewrite)
  (implies (union-at-n 1 k '(5 7))
    (union-at-n 1 k '(5 6 7 8 9 10 11)))
  ((enable union-at-n at)))

;;;If the k's entry in 1 is between 8 and 11,
;;;then it is certainly between 5 and 11.
(prove-lemma un8-11-then-un5-11 (rewrite)
  (implies (union-at-n 1 k '(8 9 10 11))
    (union-at-n 1 k '(5 6 7 8 9 10 11)))
  ((enable union-at-n at)))

;;;If the k's entry in lp is between 5 and 12 and
;;;the k's entry in lp is between 5 and 7. then
;;;the k's entry in lp in fact is between 9 and 12.
(prove-lemma k-in-lp5-7-or-lp8-or-lp9-12 (rewrite)
  (implies (and (union-at-n lp k '(5 6 7 8 9 10 11 12))
    (not (union-at-n lp k '(5 6 7)))
    (not (at lp k 8)))
    (union-at-n lp k '(9 10 11 12)))
  ((enable union-at-n at)))

;;;If the k's entry in 1 is between 5 and 11,
;;; then it is certainly between 5 and 12.
(prove-lemma un5-11-then-un5-12 (rewrite)
  (implies (union-at-n 1 k '(5 6 7 8 9 10 11))
    (union-at-n 1 k '(5 6 7 8 9 10 11 12)))
  ((enable union-at-n at)))

;;;If the k's entry in 1 is between 10 and 12,
;;; then it is certainly between 8 and 12.
(prove-lemma un10-12-then-un8-12 (rewrite)
  (implies (union-at-n 1 i '(10 11 12))
    (union-at-n 1 i '(8 9 10 11 12)))
  ((enable union-at-n at)))

;;;j-eq-k-i-neq-k

;;;If (exist-union 1 n '(8 9 10 11 12)) does not hold,
;;;then the i's entry in 1 is not between 10 and 12.
(prove-lemma i-not-110-12 (rewrite)
  (implies (and (member i (nset n))
    (not (exist-union 1 n '(8 9 10 11 12))))
    (not (union-at-n 1 i '(10 11 12))))
  ((enable exist-union union-at-n at nset)))

;*lemmas for a3

;;;j-eq-k-i-neq-k

;;;If the k's entry in 1 is between 5 and 11,
;;;then the k's entry in 1 is between 9 and 11.
(prove-lemma un5-11-eq-un58-or-un8-11 (rewrite)
  (implies (and (union-at-n 1 k '(5 6 7 8 9 10 11))
    (not (union-at-n 1 k '(5 6 7 8))))
    (union-at-n 1 k '(9 10 11)))
  ((enable union-at-n at)))

;;;If the k's entry in g is 4,
;;;then the k's entry in 1 is between 5 and 8.
(prove-lemma a3-if4 (rewrite)
  (implies (and (member k (nset n))
    (lg n 1 g)
    (at g k 4))
    (not (union-at-n 1 k '(5 6 7 8))))
  ((enable nset union-at-n at lg lg-at-n lg-3-at-n)))

;;;If the k's entry in 1 is between 5 and 11,
;;;and the k's entry in 1 is between 5 and 12,
;;;then the k's entry in 1 is 9 and 11.

```

```

(prove-lemma k-in-15-11-g4-then-19-11 (rewrite)
  (implies (and (member k (nset n))
    (lg n 1 g)
    (union-at-n 1 k '(5 6 7 8 9 10 11))
    (at g k 4))
    (union-at-n 1 k '(9 10 11)))
  ((use (a3-if4))
    (use (un5-11-eq-un58-or-un8-11))))

;;;If the i's entry in 1 is 12,
;;;then the i's entry in 1 is between 8 and 12.
(prove-lemma 112-then-un8-12 (rewrite)
  (implies (at 1 i 12)
    (union-at-n 1 i '(8 9 10 11 12)))
  ((enable at union-at-n)))

;;;If (exist-union 1 n '(8 9 10 11 12)) does not hold,
;;;then the i's entry in 1 is 12.
(prove-lemma i-not-in-112 (rewrite)
  (implies (and (member i (nset n))
    (not (exist-union 1 n '(8 9 10 11 12))))
    (not (at 1 i 12)))
  ((enable exist-union nset at union-at-n)))

;;;j-neq-k-i-eq-k

;;;If the k's entry in 1 is 11,
;;; then the k's entry in 1 is between 10 and 12.
(prove-lemma 111-then-un10-12 (rewrite)
  (implies (at 1 k 11)
    (union-at-n 1 k '(10 11 12)))
  ((enable union-at-n at)))

;;;If the j's entry in g is either 2 or 3,
;;;then the j's entry in 1 is between 5 and 8 by lg.
(prove-lemma if3 (rewrite)
  (implies (and (member j (nset n))
    (lg n 1 g)
    (not (union-at-n g j '(2 3)))
    (not (union-at-n 1 j '(5 6 7 8))))
    (not (union-at-n 1 j '(9 10 11 12)))
  ((enable union-at-n at nset lg lg-at-n lg-2-at-n)))

;;;If the j's entry in 1 is between 5 and 12 and
;;;the j's entry in 1 is between 5 and 8, then
;;;the j's entry in 1 is 9 and 12.
(prove-lemma 15-12-eq-15-8-or-19-12 (rewrite)
  (implies (and (union-at-n 1 j '(5 6 7 8 9 10 11 12))
    (not (union-at-n 1 j '(5 6 7 8))))
    (union-at-n 1 j '(9 10 11 12)))
  ((enable union-at-n at)))

;;;i-j-eq-k

;;;If the k's entry in lp is 12,
;;;then it is certainly between 5 and 12.
(prove-lemma 112-then-un9-12 (rewrite)
  (implies (at lp k 12)
    (union-at-n lp k '(9 10 11 12)))
  ((enable union-at-n at)))

;*lemmas for bla

;;;If the u's entry in g is 4,
;;;then the u's entry in 1 is between 8 and 12 by lg.
(prove-lemma bla-if4 (rewrite)
  (implies (and (member u (nset n))
    (lg n 1 g)
    (at g u 4))
    (union-at-n 1 u '(8 9 10 11 12)))
  ((enable nset union-at-n at lg lg-at-n
    lg-3-at-n)))

;*lemmas for blb

;;;If the k's entry in lp is between 9 and 12,
;;;then the k's entry in gp is iether 3 or 4 by lg.
(prove-lemma lp9-12-then-k-in-g34 (rewrite)
  (implies (and (member k (nset n))
    (union-at-n lp k '(9 10 11 12))
    (lg n lp gp))
    (union-at-n gp k '(3 4)))
  ((enable nset at union-at-n lg lg-at-n
    lg-3-at-n)))

;;;If the k's entry in lp is between 8 and 12, and
;;;it is not 8. then it is certainly between 9 and 12.
(prove-lemma un8-12-then-l8-or-l9-12 (rewrite)
  (implies (and (union-at-n lp k '(8 9 10 11 12))
    (not (at lp k 8)))
    (union-at-n lp k '(9 10 11 12)))
  ((enable at union-at-n)))

```

```

;;;Well-formed-state.
(defn ws (n l g)
  (and (numberp n)
        (listp l)
        (listp g)
        (equal (length l) n)
        (equal (length g) n)
        (all-union 1 n
                    '(0 1 2 3 4 5 6 7 8 9 10 11 12))
        (all-union g n '(0 1 2 3 4))))
(disable ws)

;;;Transitions.

(defn rhoi0 (n i l g lp gp)
  (and (at 1 i 0)
        (equal gp g) (equal lp (move 1 i 1))))

(defn rhoi1a (n i l g lp gp)
  (and (at 1 i 1)
        (equal gp g)
        (equal lp (move 1 i 2))))

(defn rhoi1b (n i l g lp gp)
  (and (at 1 i 1)
        (equal 4 gp)
        (equal lp 1)))

(defn rhoi2 (n i l g lp gp)
  (and (at 1 i 2)
        (equal lp (move 1 i 3))
        (equal gp (move g i 1))))

(defn rhoi3a (n i l g lp gp)
  (and (at 1 i 3)
        (equal gp g)
        (equal lp (move 1 i 4))
        (not (exist-union g n '(3 4)))))

(defn rhoi3b (n i l g lp gp)
  (and (at 1 i 3)
        (equal gp g)
        (equal lp 1)
        (exist-union g n '(3 4))))

(defn rhoi4 (n i l g lp gp)
  (and (at 1 i 4)
        (equal gp (move g i 3))
        (equal lp (move 1 i 5))))

(defn rhoi5a (n i l g lp gp)
  (and (at 1 i 5)
        (equal gp g)
        (exist-union g n '(1))
        (equal lp (move 1 i 6))))

(defn rhoi5b (n i l g lp gp)
  (and (at 1 i 5)
        (equal gp g)
        (not (exist-union g n '(1)))
        (equal lp (move 1 i 8))))

(defn rhoi6 (n i l g lp gp)
  (and (at 1 i 6)
        (equal gp (move g i 2))
        (equal lp (move 1 i 7)))))

(defn rhoi7a (n i l g lp gp)
  (and (at 1 i 7)
        (exist-union g n '(4))
        (equal lp (move 1 i 8))
        (equal gp g)))

(defn rhoi7b (n i l g lp gp)
  (and (at 1 i 7)
        (not (exist-union g n '(4)))
        (equal lp 1)
        (equal gp g)))

(defn rhoi8 (n i l g lp gp)
  (and (at 1 i 8)
        (equal gp (move g i 4))
        (equal lp (move 1 i 9))))

(defn phi9 (i n g)
  (if (or (nlistp g)
          (not (numberp i))
          (not (numberp n))) F
      (if (equal n 0) T
          (or (and (not (lessp n i))
                    (phi9 i (sub1 n) g))
              (and (union-at-n g n '(0 1))
                    (phi9 i (sub1 n) g)))))))
(disable phi9)

(defn rhoi9a (n i l g lp gp)
  (and (at 1 i 9)
        (equal gp g)
        (phi9 i n g)
        (equal lp 1)))

(defn rhoi9b (n i l g lp gp)
  (and (at 1 i 9)
        (equal gp g)
        (not (phi9 i n g))
        (equal lp 1)))

(defn rhoi10 (n i l g lp gp)
  (and (at 1 i 10)
        (equal lp (move 1 i 11))
        (equal gp g)))

(defn phi11 (i n g)
  (if (or (nlistp g)
          (not (numberp i))
          (not (numberp n))) F
      (if (equal n 0) T
          (or (and (not (lessp i n))
                    (phi11 i (sub1 n) g))
              (and (not (union-at-n g n '(2 3))
                    (phi11 i (sub1 n) g)))))))
(disable phi11)

(defn rhoi11a (n i l g lp gp)
  (and (at 1 i 11)
        (equal gp g)
        (phi.11 i n g)
        (equal lp (move 1 i 12))))

(defn rhoi11b (n i l g lp gp)
  (and (at 1 i 11)
        (equal gp g)
        (not (phi11 i n g))
        (equal lp 1)))

(defn rhoi12 (n i l g lp gp)
  (and (at 1 i 12)
        (equal gp (move g i 0))
        (equal lp (move 1 i 0))))

;;;The transition operates on i'th.
(defn rhoi (n i l g lp gp)
  (or (rhoi0 n i l g lp gp)
      (rhoi1a n i l g lp gp)
      (rhoi1b n i l g lp gp)
      (rhoi2 n i l g lp gp)
      (rhoi3a n i l g lp gp)
      (rhoi3b n i l g lp gp)
      (rhoi4 n i l g lp gp)
      (rhoi5a n i l g lp gp)
      (rhoi5b n i l g lp gp)
      (rhoi6 n i l g lp gp)
      (rhoi7a n i l g lp gp)
      (rhoi7b n i l g lp gp)
      (rhoi8 n i l g lp gp)
      (rhoi9a n i l g lp gp)
      (rhoi9b n i l g lp gp)
      (rhoi10 n i l g lp gp)
      (rhoi11a n i l g lp gp)
      (rhoi11b n i l g lp gp)
      (rhoi12 n i l g lp gp)))
(disable rhoi)

;;; Propositions
;;;a0
(defn a0 (n 1 k)
  (implies (and (member k (nset n))
                (exist-union 1 n '(8 9 10 11 12)))
            (not (at 1 k 4))))
(disable a0)

;;;a1
(defn a1 (n 1 g)
  (implies (exist-union 1 n '(8 9 10 11 12))
            (exist-intersect-E-12-3-4 n 1 g)))
(disable a1)

;;;: a2
(defn a2-at-n1-n2 (n1 n2 1)
  (if (union-at-n 1 n1 '(10 11 12))
      (not (union-at-n 1 n2
                      '(5 6 7 8 9 10 11 12))) T))
(disable a2-at-n1-n2)

(defn a2-at-n2 (n1 n2 1)
  (if (zerop n2) T
      (if (not (lessp n2 n1))
          (a2-at-n2 n1 (sub1 n2) 1)
          (and (a2-at-n1-n2 n1 n2 1)
                (a2-at-n2 n1 (sub1 n2) 1)))))

```

```
(disable a2-at-n2)

(defn a2 (n1 n2 1)
  (if (zerop n1) T
      (and (a2-at-n2 n1 n2 1)
            1a2 (sub1 n1) n2 1))))
(disable a2)

:::a3

(defn a3-at-n1-n2 (n1 n2 1 g)
  (if (and (at 1 n1 12)
            (union-at-n 1 n2
                        '(5 6 7 8 9 10 11 12))))
      (at g n2 4) T))

(disable a3-at-n1-n2)

(defn a3-at-n2 (n1 n2 1 g)
  (if (zerop n2) T
      (and (a3-at-n1-n2 n1 n2 1 g)
            (a3-at-n2 n1 (sub1 n2) 1 g))))

(disable a3-at-n2)

(defn a3 (n1 n2 1 g)
  (if (zerop n1) T
      (and (a3-at-n2 n1 n2 1 g)
            (a3 (sub1 n1) n2 1 g)))))
(disable a3)
```

```

;;ws implies that n is a number.
(prove-lemma ws-num-n (rewrite)
  (implies (ws n 1 g)
    (numberp n))
  ((enable us)))

;;ws implies that l is a list.
(prove-lemma ws-list-l (rewrite)
  (implies (ws n 1 g)
    (listp l))
  ((enable us)))

;;ws implies that g is a list.
(prove-lemma ws-list-g (rewrite)
  (implies (ws n 1 g)
    (listp g))
  ((enable us)))

;;ws implies that length of l is n.
(prove-lemma ws-ln-l (rewrite)
  (implies (ws n 1 g)
    (equal (length l) n))
  ((enable ws)))

;;ws implies that length of g is n.
(prove-lemma ws-ln-g (rewrite)
  (implies (ws n 1 g)
    (equal (length g) n))
  ((enable us)))

;;ws and rho imply that lp is a list.
(prove-lemma ws-ln-lp (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rhoi n k 1 g lp gp))
    (listp lp))
  ((enable ws rhoi)))

;;ws and rho imply that gp is a list.
(prove-lemma ws-ln-gp (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rhoi n k 1 g lp gp))
    (listp gp))
  ((enable ws rhoi)))

;;ws implies that n is nonzero.
(prove-lemma us-n-not-0 (rewrite)
  (implies (ws n 1 g)
    (not (zerop n)))
  ((enable ws)))

(prove-lemma n-not-0 (rewrite)
  (implies (ws n 1 g)
    (member n (nset n)))
  ((use (n-in-nset))
  (use (us-n-not-0))))

;*the rho! lemmas

;;Auxiliary lemma.
(prove-lemma lm-1-rholemma (rewrite)
  (implies (and (listp l)
    (member j (nset (length l)))
    (member k (nset (length l)))
    (rhoi n k 1 g lp gp)
    (not (equal k j)))
    (equal (nth 1 j) (nth lp j)))
  ((enable rhoi)))

(disable lm-1-rholemma)

;;Rholemma for list l.
(prove-lemma l-rholemma (rewrite)
  (implies (and (ws n 1 g)
    (member j (nset n))
    (member k (nset n))
    (rhoi n k 1 g lp gp)
    (not (equal k j)))
    (equal (nth 1 j) (nth lp j)))
  ((enable lm-1-rholemma)
  (use (lm-1-rholemma))))

;;Auxiliary lemma.
(prove-lemma lm-g-rholemma (rewrite)
  (implies (and (listp g)
    (member j (nset (length g)))
    (member k (nset (length g)))
    (rhoi n k 1 g lp gp)
    (not (equal k j)))
    (equal (nth g j) (nth gp j)))
  ((enable rhoi)))

(disable lm-g-rholemma)

;;Rholemma for list g.
(prove-lemma g-rholemma (rewrite)
  (implies (and (ws n 1 g)
    (member j (nset n))
    (member k (nset n))
    (numberp m)
    (rhoi n k 1 g lp gp)
    (not (equal j k))
    (at lp j m))
    (at 1 j m))
  ((enable at)
  (use (l-rholemma))))

(prove-lemma gp-same-g-at (rewrite)
  (implies (and (ws n 1 g)
    (member j (nset n))
    (member k (nset n))
    (numberp m)
    (rhoi n k 1 g lp gp)
    (not (equal j k))
    (at lp j m))
    (at 1 j m))
  ((enable at)
  (use (l-rholemma))))

(member j (nset n))
(member k (nset n))
(rhoi n k 1 g lp gp)
(not (equal k j))
(equal (nth g j) (nth gp j)))
((enable lm-g-rholemma)
  (use (lm-g-rholemma))))

;; lp-gp-same-l-g

;;Another version of Rholemma for l.
;;It applies to (union-at-n 1 j m) in stead of
;;(nth 1 j).
(prove-lemma lp-same-l (rewrite)
  (implies (and (ws n 1 g)
    (listp m)
    (member j (nset n))
    (member k (nset n))
    (rhoi n k 1 g lp gp)
    (not (equal j k))
    (union-at-n 1 j m))
    (union-at-n lp j m))
  ((enable union-at-n at)
  (use (l-rholemma))))

;;Contrast to the one above,
;;the order of l and lp is reversed.
(prove-lemma l-same-lp (rewrite)
  (implies (and (ws n 1 g)
    (listp m)
    (member j (nset n))
    (member k (nset n))
    (rhoi n k 1 g lp gp)
    (not (equal j k))
    (union-at-n lp j m))
    (union-at-n 1 j m))
  ((enable union-at-n at)
  (use (l-rholemma))))

(prove-lemma lp-same-l-not (rewrite)
  (implies (and (ws n 1 g)
    (listp m)
    (member j (nset n))
    (member k (nset n))
    (rhoi n k 1 g lp gp)
    (not (equal j k))
    (not (union-at-n 1 j m)))
    (not (union-at-n lp j m)))
  ((use (l-same-lp))))

;;Another version of Rholemma for g.
(prove-lemma gp-same-g (rewrite)
  (implies (and (ws n 1 g)
    (listp m)
    (member j (nset n))
    (member k (nset n))
    (rhoi n k 1 g lp gp)
    (not (equal j k))
    (union-at-n g j m))
    (union-at-n gp j m))
  ((enable union-at-n at)
  (use (g-rholemma))))

;;Contrast to the one above,
;;the order of g and gp is reversed.
(prove-lemma g-same-gp (rewrite)
  (implies (and (ws n 1 g)
    (listp m)
    (member j (nset n))
    (member k (nset n))
    (rhoi n k 1 g lp gp)
    (not (equal j k))
    (union-at-n gp j m))
    (union-at-n g j m))
  ((enable union-at-n at)
  (use (g-rholemma))))

;;It applies to (at 1 j m) in stead of
;;(nth 1 j).
(prove-lemma l-same-lp-at (rewrite)
  (implies (and (ws n 1 g)
    (member j (nset n))
    (member k (nset n))
    (numberp m)
    (rhoi n k 1 g lp gp)
    (not (equal j k))
    (at lp j m))
    (at 1 j m))
  ((enable at)
  (use (l-rholemma))))

(prove-lemma gp-same-g-at (rewrite)
  (implies (and (ws n 1 g)
    (member j (nset n))
    (member k (nset n))
    (numberp m)
    (rhoi n k 1 g lp gp)
    (not (equal j k))
    (at lp j m))
    (at 1 j m))
  ((enable at)
  (use (l-rholemma))))

```

```

(not (equal j k))
(at g j m))
(at gp j m))
((enable at)
 (use (g-rholemma))))

(prove-lemma 1-same-lp-at-not (rewrite)
 (implies (and (ws n 1 g)
 (numberp m)
 (member j (nset n))
 (member k (nset n))
 (rhoi n k 1 g lp gp)
 (not (equal j k))
 (not (at 1 j m)))
 (not (at lp j m)))
 ((use (1-same-lp-at))))

; *basic properties of a2

;; Auxiliary lemma.
(prove-lemma lm-a2-at-n2-a2-at-n1-n2 (rewrite)
 (implies (and (numberp n)
 (numberp k)
 (member j (nset n))
 (lessp j k)
 (a2-at-n2 k n 1))
 (a2-at-n1-n2 k j 1))
 ((enable nset a2-at-n2 a2-at-n1-n2))))

(disable lm-a2-at-n2-a2-at-n1-n2)

(prove-lemma a2-at-n2-a2-at-n1-n2 (rewrite)
 (implies (and (we n 1 g)
 (member k (nset n))
 (member j (nset n))
 (lessp j k)
 (a2-at-n2 k n 1))
 (a2-at-n1-n2 k j 1))
 ((enable lm-a2-at-n2-a2-at-n1-n2)
 (use (lm-a2-at-n2-a2-at-n1-n2))))

(prove-lemma lm-a2-a2-at-n2 (rewrite)
 (implies (and (numberp n)
 (numberp i)
 (member k (nset n))
 (a2 n i 1))
 (a2-at-n2 k i 1))
 ((enable nset a2))))

(prove-lemma a2-a2-at-n2 (rewrite)
 (implies (and (ws n 1 g)
 (member i (nset n))
 (member k (nset n))
 (a2 n i 1))
 (a2-at-n2 k i 1))
 ((use (lm-a2-a2-at-n2))))

; *basic properties of a3

(prove-lemma lm-a3-at-n2-a3-at-n1-n2 (rewrite)
 (implies (and (numberp n)
 (numberp u)
 (member j (nset n))
 (a3-at-n2 u n 1 g))
 (a3-at-n1-n2 u j 1 g))
 ((enable nset a3-at-n2 a3-at-n1-n2))))

(disable lm-a3-at-n2-a3-at-n1-n2)

(prove-lemma a3-at-n2-a3-at-n1-n2 (rewrite)
 (implies (and (ws n 1 g)
 (member u (nset n))
 (member j (nset n))
 (a3-at-n2 u n 1 g))
 (a3-at-n1-n2 u j 1 g))
 ((enable lm-a3-at-n2-a3-at-n1-n2)
 (use (lm-a3-at-n2-a3-at-n1-n2))))

(prove-lemma lm-a3-a3-at-n2 (rewrite)
 (implies (and (numberp n)
 (numberp i)
 (member u (nset n))
 (a3 n i 1 g))
 (a3-at-n2 u i 1 g))
 ((enable nset a3))))

(disable lm-a3-a3-at-n2)

(prove-lemma a3-a3-at-n2 (rewrite)
 (implies (and (ws n 1 g)
 (member i (nset n))
 (member u (nset n))
 (a3 n i 1 g))
 (a3-at-n2 u i 1 g))
 ((enable lm-a3-a3-at-n2)
 (use (lm-a3-a3-at-n2))))

; ::::: Instances used in the proofs of

```

```

; ::::: a1 a2 and a3.

; :: (a2 n n 1) and (a3 n n 1 g) are involved
; :: in double bounded quantifiers
; :: \forall i \leq n \forall j \leq n,
; :: with their quantifier-free formulas
; :: (a3-at-n1-n2 i j 1 g) and (a2-at-n1-n2 i j 1)
; :: respectively. What follows are all instances of
; :: the following type: If (a3 n n 1 g) holds, then
; :: in particular, so its instance
; :: (a3-at-n1-n2 i j 1 g) does.

; :: The instances are i and j.
(prove-lemma a3-i-j-a3-at-n1-n2 (rewrite)
 (implies (and (ws n 1 g)
 (member i (nset n))
 (member j (nset n))
 (a3 n n 1 g))
 (a3-at-n1-n2 i j 1 g))
 ((use (a3-a3-at-n2 (u i) (i n)))
 (use (a3-at-n2-a3-at-n1-n2 (u i))))))

; :: The instances are k and
; :: (exist-union lp n ' (8 9 10 11 12)).
(prove-lemma a3-ex-a3-at-n1-n2 (rewrite)
 (implies (and (ws n 1 g)
 (member k (nset n))
 (a3 n n 1 g)
 (exist-union lp n ' (8 9 10 11 12)))
 (a3-at-n1-n2 k
 (exist-union lp n ' (8 9 10 11 12)) 1 g))
 ((use (a3-a3-at-n2 (u k) (i n)))
 (use (a3-at-n2-a3-at-n1-n2 (u k)
 (j (exist-union lp n ' (8 9 10 11 12)))))))

(prove-lemma a2-n-a2-at-n2 (rewrite)
 (implies (and (ws n 1 g)
 (member k (nset n))
 (a2 n n 1))
 (a2-at-n2 k n 1))
 ((use (a2-a2-at-n2 (i n))))))

; :: The instances are i and j.
(prove-lemma a2-i-j-a2-at-n1-n2 (rewrite)
 (implies (and (ws n 1 g)
 (member i (nset n))
 (member j (nset n))
 (a2 n n 1))
 (lessp j i))
 (a2-at-n1-n2 i j 1))
 ((use (a2-a2-at-n2 (i n)))
 (use (a2-at-n2-a2-at-n1-n2 (k i)))))

```

```

(prove-lemma j-eq-k-move-member-g (rewrite)
  (implies (and (listp g)
    (listp m)
    (member i m)
    (member k (nset (length g))))
    (member (nth (move g k i) k) m)))

(prove-lemma j-neg-k-move-member-g (rewrite)
  (implies (and (listp g)
    (listp m)
    (member k (nset (length g)))
    (not (equal j k))
    (member i m)
    (member (nth g j) m))
    (member (nth (move g k i) j) m)))

(prove-lemma move-member-g (rewrite)
  (implies (and (listp g)
    (listp m)
    (member i m)
    (member k (nset (length g)))
    (member (nth g j) m))
    (member (nth (move g k i) j) m))
  ((use (j-neg-k-move-member-g) (j-eq-k-move-member-g))))

(prove-lemma move-member-l (rewrite)
  (implies (and (listp l)
    (listp m)
    (numberp j)
    (member i m)
    (member k (nset (length l)))
    (member (nth l j) m))
    (member (nth (move l k i) j) m)))

(prove-lemma us-union-g (rewrite)
  (implies (ws n 1 g)
    (all-union g n '(0 1 2 3 4)))
  ((enable ws)))

(prove-lemma us-union-l (rewrite)
  (implies (ws n 1 g)
    (all-union l n '(0 1 2 3 4 5 6 7 8 9 10 11 12)))
  ((enable us)))

(prove-lemma rho0-preserves-union-g (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rhoi0 n k 1 g lp gp))
    (all-union gp n '(0 1 2 3 4))))

(prove-lemma rhola-preserves-union-g (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rhoi1a n k 1 g lp gp))
    (all-union gp n '(0 1 2 3 4))))

(prove-lemma rholb-preserves-union-g (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rhoi1b n k 1 g lp gp))
    (all-union gp n '(0 1 2 3 4))))

(prove-lemma lm-rho2-preserves-union-g (rewrite)
  (implies (and (listp g)
    (equal (length g) n)
    (member k (nset n))
    (all-union g j '(0 1 2 3 4))
    (rhoi2 n k 1 g lp gp))
    (all-union gp j '(0 1 2 3 4)))
  ((enable nset all-union union-at-n at)))

(prove-lemma rho2-preserves-union-g (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rhoi2 n k 1 g lp gp))
    (all-union gp n '(0 1 2 3 4)))
  ((disable rhoi2)
  (use (lm-rho2-preserves-union-g (j n)))))

(prove-lemma rho3a-preserves-union-g (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rhoi3a n k 1 g lp gp))
    (all-union gp n '(0 1 2 3 4))))

(prove-lemma rho3b-preserves-union-g (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rhoi3b n k 1 g lp gp))
    (all-union gp n '(0 1 2 3 4))))

(prove-lemma lm-rho4-preserves-union-g (rewrite)
  (implies (and (listp g)
    (equal (length g) n)
    (member k (nset n))
    (all-union g j '(0 1 2 3 4))
    (rhoi4 n k 1 g lp gp))
    (all-union gp j '(0 1 2 3 4)))
  ((enable nset all-union union-at-n at)))

((enable nset all-union union-at-n at)))

(prove-lemma rho4-preserves-union-g (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rhoi4 n k 1 g lp gp))
    (all-union gp n '(0 1 2 3 4)))
  ((disable rhoi4)
  (use (lm-rho4-preserves-union-g (j n)))))

(prove-lemma rho5a-preserves-union-g (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rhoi5a n k 1 g lp gp))
    (all-union gp n '(0 1 2 3 4))))

(prove-lemma rho5b-preserves-union-g (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rhoi5b n k 1 g lp gp))
    (all-union gp n '(0 1 2 3 4))))

(prove-lemma lm-rho6-preserves-union-g (rewrite)
  (implies (and (listp g)
    (equal (length g) n)
    (member k (nset n))
    (all-union g j '(0 1 2 3 4))
    (rhoi6 n k 1 g lp gp))
    (all-union gp j '(0 1 2 3 4)))
  ((enable nset all-union union-at-n at)))

(prove-lemma rho6-preserves-union-g (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rhoi6 n k 1 g lp gp))
    (all-union gp n '(0 1 2 3 4)))
  ((disable rhoi6)
  (use (lm-rho6-preserves-union-g (j n)))))

(prove-lemma rho7a-preserves-union-g (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rhoi7a n k 1 g lp gp))
    (all-union gp n '(0 1 2 3 4))))

(prove-lemma rho7b-preserves-union-g (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rhoi7b n k 1 g lp gp))
    (all-union gp n '(0 1 2 3 4))))

(prove-lemma lm-rho8-preserves-union-g (rewrite)
  (implies (and (listp g)
    (equal (length g) n)
    (member k (nset n))
    (all-union g j '(0 1 2 3 4))
    (rhoi8 n k 1 g lp gp))
    (all-union gp j '(0 1 2 3 4)))
  ((enable nset all-union union-at-n at)))

(prove-lemma rho8-preserves-union-g (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rhoi8 n k 1 g lp gp))
    (all-union gp n '(0 1 2 3 4)))
  ((disable rhoi8)
  (use (lm-rho8-preserves-union-g (j n)))))

(prove-lemma rho9a-preserves-union-g (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rhoi9a n k 1 g lp gp))
    (all-union gp n '(0 1 2 3 4))))

(prove-lemma rho9b-preserves-union-g (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rhoi9b n k 1 g lp gp))
    (all-union gp n '(0 1 2 3 4))))

(prove-lemma lm-rho10-preserves-union-g (rewrite)
  (implies (and (listp g)
    (equal (length g) n)
    (member k (nset n))
    (all-union g j '(0 1 2 3 4))
    (rhoi10 n k 1 g lp gp))
    (all-union gp j '(0 1 2 3 4)))
  ((enable nset all-union union-at-n at)))

(prove-lemma rho10-preserves-union-g (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rhoi10 n k 1 g lp gp))
    (all-union gp n '(0 1 2 3 4)))
  ((enable nset all-union union-at-n at)))

(prove-lemma rho11a-preserves-union-g (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rhoi11a n k 1 g lp gp))
    (all-union gp n '(0 1 2 3 4))))

```

```

(prove-lemma rho2-preserves-union-1 (rewrite)
  (implies (and (ws n l g)
    (member k (nset n))
    (rhoi2 n k l g lp gp))
    (all-union lp n '(0 1 2 3 4 5 6 7 8 9 10 11 12))))
  ((disable rhoi2)
    (use (lm-rho2-preserves-union-1 (j n)))))

(prove-lemma lm-rho3a-preserves-union-1 (rewrite)
  (implies (and (listp l)
    (equal (length l) n)
    (member k (nset n))
    (all-union 1 j '(0 1 2 3 4 5 6 7 8 9 10 11 12))
    (rhoi3a n k l g lp gp))
    (all-union lp j '(0 1 2 3 4 5 6 7 8 9 10 11 12))))
  ((enable nset all-union union-at-n at)))

(prove-lemma rho3a-preserves-union-1 (rewrite)
  (implies (and (ws n l g)
    (member k (nset n))
    (rhoi3a n k l g lp gp))
    (all-union lp n '(0 1 2 3 4 5 6 7 8 9 10 11 12))))
  ((disable rhoi3a)
    (use (lm-rhoi3a-preserves-union-1 (j n)))))

(prove-lemma lm-rho3b-preserves-union-1 (rewrite)
  (implies (and (listp l)
    (equal (length l) n)
    (member k (nset n))
    (all-union 1 j '(0 1 2 3 4 5 6 7 8 9 10 11 2))
    (rhoi3b n k l g lp gp))
    (all-union lp j '(0 1 2 3 4 5 6 7 8 9 10 11 12))))
  ((enable nset all-union union-at-n at)))

(prove-lemma rho3b-preserves-union-1 (rewrite)
  (implies (and (ws n l g)
    (member k (nset n))
    (rhoi3b n k l g lp gp))
    (all-union lp n '(0 1 2 3 4 5 6 7 8 9 10 11 12))))
  ((disable rhoi3b)
    (use (lm-rho3b-preserves-union-1 (j n)))))

(prove-lemma lm-rho4-preserves-union-1 (rewrite)
  (implies (and (listp l)
    (equal (length l) n)
    (member k (nset n))
    (all-union 1 j '(0 1 2 3 4 5 6 7 8 9 10 11 12))
    (rhoi4 n k l g lp gp))
    (all-union lp j '(0 1 2 3 4 5 6 7 8 9 10 11 12))))
  ((enable nset all-union union-at-n at)))

(prove-lemma rho4-preserves-union-1 (rewrite)
  (implies (and (ws n l g)
    (member k (nset n))
    (rhoi4 n k l g lp gp))
    (all-union lp n '(0 1 2 3 4 5 6 7 8 9 10 11 12))))
  ((disable rhoi4)
    (use (lm-rhoi4-preserves-union-1 (j n)))))

) (prove-lemma lm-rho5a-preserves-union-1 (rewrite)
  (implies (and (listp l)
    (equal (length l) n)
    (member k (nset n))
    (all-union 1 j '(0 1 2 3 4 5 6 7 8 9 10 11 12))
    (rhoi5a n k l g lp gp))
    (all-union lp j '(0 1 2 3 4 5 6 7 8 9 10 11 12))))
  ((enable nset all-union union-at-n at)))

(prove-lemma rho5a-preserves-union-1 (rewrite)
  (implies (and (ws n l g)
    (member k (nset n))
    (rhoi5a n k l g lp gp))
    (all-union lp n '(0 1 2 3 4 5 6 7 8 9 10 11 12))))
  ((disable rhoi5a)
    (use (lm-rho5a-preserves-union-1 (j n)))))

) (prove-lemma lm-rho5b-preserves-union-1 (rewrite)
  (implies (and (listp l)
    (equal (length l) n)
    (member k (nset n))
    (all-union 1 j '(0 1 2 3 4 5 6 7 8 9 10 11 12))
    (rhoi5b n k l g lp gp))
    (all-union lp j '(0 1 2 3 4 5 6 7 8 9 10 11 12))))
  ((enable nset all-union union-at-n at)))

(prove-lemma rho5b-preserves-union-1 (rewrite)
  (implies (and (ws n l g)
    (member k (nset n))
    (rhoi5b n k l g lp gp))
    (all-union lp n '(0 1 2 3 4 5 6 7 8 9 10 11 12))))
  ((disable rhoi5b)
    (use (lm-rho5b-preserves-union-1 (j n)))))

) (prove-lemma lm-rho6-preserves-union-1 (rewrite)
  (implies (and (listp l)
    (equal (length l) n)
    (member k (nset n))
    (all-union 1 i '(0 1 2 3 4 5 6 7 8 9 10 11 12))

```

```

(rhoi6 n k 1 g lp gp))
(all-union lp j '(0 1 2 3 4 5 6 7 8 9 10 11 12)))
(enable nset all-union union-at-n at)))

(prove-lemma rho6-preserves-union-1 (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rhoi6 n k 1 g lp gp))
    (all-union lp n '(0 1 2 3 4 5 6 7 8 9 10 11 12)))
  ((disable rhoi6)
   (use (lm-rho6-preserves-union-1 (j n))))))

(prove-lemma lm-rho7a-preserves-union-1 (rewrite)
  (implies (and (listp 1)
    (equal (length 1) nj)
    (member k (nset n))
    (all-union 1 j '(0 1 2 3 4 5 6 7 8 9 10 11 12))
    (rhoi7a n k 1 g lp gpj))
    (all-union lp j '(0 12 3 4 5 6 7 8 9 10 11 12)))
  ((enable nset all-union union-at-n at)))

(prove-lemma rho7a-preserves-union-1 (rewrite)
  (implies (and (ws n 1 gj)
    (member k (nset n))
    (rhoi7a n k 1 g lp gpj))
    (all-union lp n '(0 1 2 3 4 5 6 7 8 9 10 11 12)))
  ((disable rhoi7a)
   (use (lm-rho7a-preserves-union-1 (j n))))))

(prove-lemma lm-rho7b-preserves-union-1 (rewrite)
  (implies (and (listp 1)
    (equal (length 1) nj)
    (member k (nset n))
    (all-union 1 j '(0 1 2 3 4 5 6 7 8 9 10 11 12))
    (rhoi7b n k 1 g lp gpjj))
    (all-union lp j '(0 1 2 3 4 5 6 7 8 9 10 11 12)))
  ((enable nset all-union union-at-n at)))

(prove-lemma rho7b-preserves-union-1 (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rhoi7b n k 1 g lp gp))
    (all-union lp n '(0 12 3 4 5 6 7 8 9 10 11 12)))
  ((disable rhoi7b)
   (use (lm-rho7b-preserves-union-1 (j n))))))

(prove-lemma lm-rho8-preserves-union-1 (rewrite)
  (implies (and (listp 1)
    (equal (length 1) n)
    (member k (nset n))
    (all-union 1 j '(0 1 2 3 4 5 6 7 8 9 10 11 12))
    (rhoi8 n k 1 g lp gp))
    (all-union lp j '(0 1 2 3 4 5 6 7 8 9 10 11 12)))
  ((enable nset all-union union-at-n at)))

(prove-lemma rho8-preserves-union-1 (rewrite)
  (implies (and (ws n 1 gj)
    (member k (nset njj)
    (rhoi8 n k 1 g lp gpjj))
    (all-union lp n '(0 1 2 3 4 5 6 7 8 9 10 11 12)))
  ((disable rhoi8)
   (use (lm-rho8-preserves-union-1 (j n))))))

(prove-lemma lm-rho9a-preserves-union-1 (rewrite)
  (implies (and (listp 1)
    (equal (length 1) nj)
    (member k (nset n))
    (all-union 1 j '(0 1 2 3 4 5 6 7 8 9 10 11 12))
    (rhoi9a n k 1 g lp gp))
    (all-union lp j '(0 1 2 3 4 5 6 7 8 9 10 11 12)))
  ((enable nset all-union union-at-n at)))

(prove-lemma rho9a-preserves-union-1 (rewrite)
  (implies (and (ws n 1 gj)
    (member k (nset n))
    (rhoi9a n k 1 g lp gp))
    (all-union lp n '(0 1 2 3 4 5 6 7 8 9 10 11 12)))
  ((disable rhoi9a)
   (use (lm-rho9a-preserves-union-1 (j njjj))))))

(prove-lemma lm-rho9b-preserves-union-1 (rewrite)
  (implies (and (listp 1)
    (equal (length 1) nj)
    (member k (nset n))
    (all-union 1 j '(0 1 2 3 4 5 6 7 8 9 10 11 12))
    (rhoi9b n k 1 g lp gp))
    (all-union lp j '(0 12 3 4 5 6 7 8 9 10 11 12)))
  ((enable nset all-union union-at-n at)))

(prove-lemma rho9b-preserves-union-1 (rewrite)
  (implies (and (ws n 1 gj)
    (member k (nset nj)
    (rhoi9b n k 1 g lp gp))
    (all-union lp n '(0 1 2 3 4 5 6 7 8 9 10 11 12)))
  ((disable rhoi9b)
   (use (lm-rho9b-preserves-union-1 (j n))))))

(prove-lemma lm-rho10-preserves-union-1 (rewrite)
  (implies (and (listp 1)
    (equal (length 1) nj)
    (member k (nset n))
    (all-union 1 j '(0 1 2 3 4 5 6 7 8 9 10 11 12))
    (rhoi10 n k 1 g lp gp))
    (all-union lp j '(0 1 2 3 4 5 6 7 8 9 10 11 12)))
  ((enable nset all-union union-at-n at)))

(prove-lemma rho10-preserves-union-1 (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset nj)
    (rhoi10 n k 1 g lp gp))
    (all-union lp n '(0 1 2 3 4 5 6 7 8 9 10 11 12)))
  ((disable rhoi10)
   (use (lm-rho10-preserves-union-1 (j n))))))

(prove-lemma lm-rholla-preserves-union-1 (rewrite)
  (implies (and (listp 1)
    (equal (length 1) n)
    (member k (nset n))
    (all-union 1 j '(0 1 2 3 4 5 6 7 8 9 10 11 12))
    (rhoi11a n k 1 g lp gpj))
    (all-union lp j '(0 1 2 3 4 5 6 7 8 9 10 11 12)))
  ((enable nset all-union union-at-n at)))

(prove-lemma rholla-preserves-union-1 (rewrite)
  (implies (and (ws n 1 gj)
    (member k (nset njj)
    (rhoi11a n k 1 g lp gp))
    (all-union lp n '(0 1 2 3 4 5 6 7 8 9 10 11 12)))
  ((disable rhoi11a)
   (use (lm-rholla-preserves-union-1 (j n))))))

(prove-lemma lm-rhollb-preserves-union-1 (rewrite)
  (implies (and (listp 1)
    (equal (length 1) n)
    (member k (nset nj)
    (all-union 1 j '(0 1 2 3 4 5 6 7 8 9 10 11 12))
    (rhoi11b n k 1 g lp gp))
    (all-union lp j '(0 1 2 3 4 5 6 7 8 9 10 11 12)))
  ((enable nset all-union union-at-n at)))

(prove-lemma rhollb-preserves-union-1 (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rhoi11b n k 1 g lp gp))
    (all-union lp n '(0 1 2 3 4 5 6 7 8 9 10 11 12)))
  ((disable rhoi11b)
   (use (lm-rhollb-preserves-union-1 (j n))))))

(prove-lemma lm-rho12-preserves-union-1 (rewrite)
  (implies (and (listp 1)
    (equal (length 1) nj)
    (member k (nset n))
    (all-union 1 j '(0 1 2 3 4 5 6 7 8 9 10 11 12))
    (rhoi12 n k 1 g lp gp))
    (all-union lp j '(0 1 2 3 4 5 6 7 8 9 10 11 12)))
  ((enable nset all-union union-at-n at)))

(prove-lemma rho12-preserves-union-1 (rewrite)
  (implies (and (ws n 1 gj)
    (member k (nset n))
    (rhoi12 n k 1 g lp gpj)
    (all-union lp n '(0 1 2 3 4 5 6 7 8 9 10 11 12)))
  ((disable rhoi12)
   (use (lm-rho12-preserves-union-1 (j njjj))))))

(prove-lemma rho-preserves-union-1 (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset nj)
    (rhoi n k 1 g lp gp))
    (all-union lp n '(0 1 2 3 4 5 6 7 8 9 10 11 12)))
  ((disable rhoi10 rhoi11a rhoi11b rhoi12 rhoi13a rhoi13b rhoi14 rhoi15a rhoi15b rhoi16 rhoi17a rhoi17b rhoi18 rhoi19a rhoi19b rhoi110 rhoi11a rhoi11b rhoi112)
   (enable rhoi)))

(prove-lemma lm-rho-preserves-ln-1 (rewrite)
  (implies (and (listp 1)
    (equal (length 1) n)
    (member k (nset n))
    (rhoi n k 1 g lp gp))
    (equal (length lpj n))
  ((enable rhoi)))

(prove-lemma rho-preserves-ln-1 (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rhoi n k 1 g lp gpj)
    (equal (length lpj n))
  ((use (lm-rho-preserves-ln-1))))))

(prove-lemma lm-rho-preserves-ln-g (rewrite)
  (implies (and (listp g)
    (equal (length gj n)
    (member k (nset n))
    (rhoi n k 1 g lp gp))

```

```

      (equal (length gp) n))
  ((enable rhoijj)

(prove-lemma rho-preserves-ln-g (rewrite)
  (implies (and (ws n 1 gj)
    (member k (nset njj)
      (rhoi n k 1 g lp gpjj)
      (equal (length gpj) njj)
      (use (lm-rho-preserves-ln-gjjjj)

(prove-lemma lm-rho-preserves-us (rewrite)
  (implies (and (numberp nj)
    (listp lp)
    (listp gpj)
    (equal (length lp) nj)
    (equal (length gpj) nj)
    (all-union lp n '(0 1 2 3 4 5 6 7 8 9 10 11 12))
    (all-union gp n '(0 1 2 3 4)))
    (ws n lp gpjj)
    (enable ws)))

(prove-lemma rho-preserves-us (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rhoi n k 1 g lp gpjj)
    (ws n lp gpjj)
    (use (lm-rho-preserves-us))))

```

```

;;;rhoi0
(prove-lemma n-neq-k-rhoi (rewrite)
  (implies (and (listp l)
    (listp g)
    (numberp n)
    (member k (nset (length l)))
    (not (equal k n))
    (at 1 k 0)
    (lg-at-n n 1 g))
    (lg-at-n n (move 1 k 1) g))
  ((enable at lg-at-n lg-1-at-n lg-2-at-n
    lg-3-at-n)))

(disable n-neq-k-rhoi0)

(prove-lemma n-eq-k-rhoi (rewrite)
  (implies (and (listp l)
    (listp g)
    (member k (nset (length l)))
    (at 1 k 0)
    (lg-at-n k 1 g))
    (lg-at-n k (move 1 k 1) g))
  ((enable at lg-at-n lg-1-at-n lg-2-at-n
    lg-3-at-n)))

(disable n-eq-k-rhoi0)

(prove-lemma lg-at-rhoi0 (rewrite)
  (implies (and (listp l)
    (listp g)
    (numberp n)
    (member k (nset (length l)))
    (at 1 k 0)
    (lg-at-n n 1 g))
    (lg-at-n n (move 1 k 1) g))
  ((enable n-neq-k-rhoi n-eq-k-rhoi0)
  (use (n-neq-k-rhoi0))
  (use (n-eq-k-rhoi.0))))

(disable lg-at-rhoi0)

(prove-lemma lg-rhoi0 (rewrite)
  (implies (and (listp l)
    (listp g)
    (member k (nset (length l)))
    (numberp n)
    (at 1 k 0)
    (lg n 1 g))
    (lg n (move 1 k 1) g))
  ((enable lg-at-rhoi0 lg at)))

(disable lg-rhoi0)

(prove-lemma rhoi0-preserves-lg (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rhoi0 n k 1 g lp gp)
    (lg n 1 g))
    (lg n lp gp))
  ((enable lg-rhoi0)))

;;;rhoila
(prove-lemma n-neq-k-rhoila (rewrite)
  (implies (and (listp l)
    (listp g)
    (numberp n)
    (member k (nset (length l)))
    (not (equal k n))
    (at 1 k 1)
    (lg-at-n n 1 g))
    (lg-at-n n (move 1 k 2) g))
  ((enable at lg-at-n lg-1-at-n lg-2-at-n
    lg-3-at-n)))

(disable n-neq-k-rhoila)

(prove-lemma n-eq-k-rhoila (rewrite)
  (implies (and (listp l)
    (listp g)
    (member k (nset (length l)))
    (at 1 k 1)
    (lg-at-n k 1 g))
    (lg-at-n n (move 1 k 2) g))
  ((enable at lg-at-n lg-1-at-n lg-2-at-n
    lg-3-at-n)))

(disable n-eq-k-rhoila)

(prove-lemma lg-at-rhoila (rewrite)
  (implies (and (listp l)
    (listp g)
    (numberp n)
    (member k (nset (length l)))
    (at 1 k 1)
    (lg-at-n n 1 g))
    (lg-at-n n (move 1 k 2) g))
  ((enable n-neq-k-rhoila n-eq-k-rhoila)
  (use (n-neq-k-rhoila))
  (use (n-eq-k-rhoila))))

```

```

  (use (n-eq-k-rhoila))))

(disable lg-at-rhoila)

(prove-lemma lg-rhoila (rewrite)
  (implies (and (listp l)
    (listp g)
    (member k (nset (length l)))
    (numberp n)
    (at 1 k 1)
    (lg n 1 g))
    (lg n (move 1 k 2) g))
  ((enable lg-at-rhoila lg at)))

(disable lg-rhoila)

(prove-lemma rhoila-preserves-lg (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rhoila n k 1 g lp gp)
    (lg n 1 g))
    (lg n lp gp))
  ((enable lg-rhoila)))

;;;rhoilb
(prove-lemma rhoilb-preserves-lg (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rhoilb n k 1 g lp gp)
    (lg n 1 g))
    (lg n lp gp))
  ((enable rhoilb)))

;;;rhoi2
(prove-lemma n-neq-k-rhoi (rewrite)
  (implies (and (listp l)
    (listp g)
    (numberp n)
    (member k (nset (length l)))
    (not (equal k n))
    (at 1 k 2)
    (lg-at-n n 1 g))
    (lg-at-n n (move 1 k 3) (move g k 1)))
  ((enable at lg-at-n lg-1-at-n lg-2-at-n
    lg-3-at-n)))

(disable n-neq-k-rhoi2)

(prove-lemma n-eq-k-rhoi (rewrite)
  (implies (and (listp l)
    (listp g)
    (member k (nset (length l)))
    (at 1 k 2)
    (lg-at-n k 1 g))
    (lg-at-n n (move 1 k 3) (move g k 1)))
  ((enable at lg-at-n lg-1-at-n lg-2-at-n
    lg-3-at-n)))

(disable n-eq-k-rhoi2)

(prove-lemma lg-at-rhoi2 (rewrite)
  (implies (and (listp l)
    (listp g)
    (numberp n)
    (member k (nset (length l)))
    (at 1 k 2)
    (lg-at-n n 1 g))
    (lg-at-n n (move 1 k 3) (move g k 1)))
  ((enable n-neq-k-rhoi n-eq-k-rhoi2)
  (use (n-neq-k-rhoi2))
  (use (n-eq-k-rhoi2))))

(disable lg-at-rhoi2)

(prove-lemma lg-rhoi2 (rewrite)
  (implies (and (listp l)
    (listp g)
    (member k (nset (length l)))
    (numberp n)
    (at 1 k 2)
    (lg n 1 g))
    (lg n (move 1 k 3) (move g k 1)))
  ((enable lg-at-rhoi lg at)))

(disable lg-rhoi2)

(prove-lemma rhoi2-preserves-lg (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rhoi2 n k 1 g lp gp)
    (lg n 1 g))
    (lg n lp gp))
  ((enable lg-rhoi2)))

;;;rhoi3a
(prove-lemma n-neq-k-rhoi3a (rewrite)
  (implies (and (listp l)
    (listp g)

```

```

      (numberp nj
      (member k (nset (length 1)))
      (not (equal k njj)
      (at 1 k 3)
      (lg-at-n n 1 gjj)
      (lg-at-n n (move 1 k 4) gjj)
      ((enable at lg-at-n lg-1-at-n lg-2-at-n
      lg-3-at-n)))

(disable n-neq-k-rhoi3a)

(prove-lemma n-eq-k-rhoi3a (rewrite)
  (implies (and (listp 1)
    (listp g)
    (member k (nset (length 1)))
    (at 1 k 3)
    (lg-at-n k 1 g))
    (lg-at-n k (move 1 k 4) gj)
    ((enable at lg-at-n lg-1-at-n lg-2-at-n
    lg-3-at-n)))

(disable n-eq-k-rhoi3a)

(prove-lemma lg-at-rhoi3a (rewrite)
  (implies (and (listp 1)
    (listp g)
    (numberp nj
    (member k (nset (length 1)))
    (at 1 k 3)
    (lg-at-n n 1 g))
    (lg-at-n n (move 1 k 4) g))
    ((enable n-neq-k-rhoi3a n-eq-k-rhoi3a)
    (use (n-neq-k-rhoi3a))
    (use (n-eq-k-rhoi3a))))

(disable lg-at-rhoi3a)

(prove-lemma lg-rhoi3a (rewrite)
  (implies (and (listp 1)
    (listp g)
    (member k (nset (length 1)))
    (numberp n)
    (at 1 k 3)
    (lg n 1 g))
    (lg n (move 1 k 4) gj)
    ((enable lg-at-rhoi3a lg at)))

(disable lg-rhoi3a)

(prove-lemma rhoi3a-preserves-lg (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rhoi3a n k 1 g lp gp)
    (lg n 1 g))
    (lg n lp gp)
    ((enable lg-rhoi3a)))

;;;rhoi3b
(prove-lemma rhoi3b-preserves-lg (rewrite)
  (implies (and (us n 1 g)
    (member k (nset n))
    (rhoi3b n k 1 g lp gp)
    (lg n 1 g))
    (lg n lp gpj)
    ((enable rhoi3b)))

;;;rhoi4
(prove-lemma n-neq-k-rhoi (rewrite)
  (implies (and (listp 1)
    (listp g)
    (numberp n)
    (member k (nset (length 1)))
    (not (equal k n))
    (at 1 k 4)
    (lg-at-n n 1 g))
    (lg-at-n n (move 1 k 5) (move g k 3)))
    ((enable at lg-at-n lg-1-at-n lg-2-at-n
    lg-3-at-n)))

(disable n-neq-k-rhoi4j)

(prove-lemma n-eq-k-rhoi (rewrite)
  (implies (and (listp 1)
    (listp g)
    (member k (nset (length 1)))
    (at 1 k 4)
    (lg-at-n k 1 g))
    (lg-at-n n (move 1 k 5) (move g k 3)))
    ((enable at lg-at-n lg-1-at-n lg-2-at-n
    lg-3-at-n)))

(disable n-eq-k-rhoi4)

(prove-lemma lg-at-rhoi4 (rewrite)
  (implies (and (listp 1)
    (listp g)
    (numberp nj
    (member k (nset (length 1)))

```

```

      (at 1 k 4)
      (lg-at-n n 1 g))
      (lg-at-n n (move 1 k 5) (move g k 3)))
      ((enable n-neq-k-rhoi n-eq-k-rhoi4)
      (use (n-neq-k-rhoi4j)
      (use (n-eq-k-rhoi4j)))

(disable lg-at-rhoi4)

(prove-lemma lg-rhoi4 (rewrite)
  (implies (and (listp 1)
    (listp g)
    (member k (nset (length 1)))
    (numberp nj
    (at 1 k 4)
    (lg n 1 g))
    (lg n (move 1 k 5) (move g k 3)))
    ((enable lg-at-rhoi lg at)))

(disable lg-rhoi4)

(prove-lemma rhoi4-preserves-lg (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset nj)
    (rhoi4 n k 1 g lp gp)
    (lg n 1 g))
    (lg n lp gpj)
    ((enable lg-rhoi4)))

;;;rhoi5a
(prove-lemma n-neq-k-rhoi5a (rewrite)
  (implies (and (listp 1)
    (listp g)
    (numberp n)
    (member k (nset (length 1)))
    (not (equal k n))
    (at 1 k 5)
    (lg-at-n n 1 g))
    (lg-at-n n (move 1 k 6) g))
    ((enable at lg-at-n lg-1-at-n lg-2-at-n
    lg-3-at-n)))

(disable n-neq-k-rhoi5aj)

(prove-lemma n-eq-k-rhoi5a (rewrite)
  (implies (and (listp 1)
    (listp g)
    (member k (nset (length 1)))
    (at 1 k 5)
    (lg-at-n k 1 gj)
    (lg-at-n k (move 1 k 6) g))
    ((enable at lg-at-n lg-1-at-n lg-2-at-n
    lg-3-at-n)))

(disable n-eq-k-rhoi5aj)

(prove-lemma lg-at-rhoi5a (rewrite)
  (implies (and (listp 1)
    (listp g)
    (numberp n)
    (member k (nset (length 1)))
    (at 1 k 5)
    (lg-at-n n 1 g))
    (lg-at-n n (move 1 k 6) g))
    ((enable n-neq-k-rhoi5a n-eq-k-rhoi5aj)
    (use (n-neq-k-rhoi5aj)
    (use (n-eq-k-rhoi5aj)))

(disable lg-at-rhoi5aj)

(prove-lemma lg-rhoi5a (rewrite)
  (implies (and (listp 1)
    (listp g)
    (member k (nset (length 1)))
    (numberp nj
    (at 1 k 5)
    (lg n 1 gj)
    (lg n (move 1 k 6) gj)
    ((enable lg-at-rhoi5a lg at)))

(disable lg-rhoi5a)

(prove-lemma rhoi5a-preserves-lg (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rhoi5a n k 1 g lp gp)
    (lg n 1 gj)
    (lg n lp gpj)
    ((enable lg-rhoi5a)))

;;;rhoi5b
(prove-lemma n-neq-k-rhoi5b (rewrite)
  (implies (and (listp 1)
    (listp g)
    (numberp n)
    (member k (nset (length 1)))
    (not (equal k n))
    (at 1 k 5)

```

```

      (lg-at-n n 1 gj)
      (lg-at-n n (move 1 k 8) g))
  ((enable at lg-at-n lg-1-at-n lg-2-at-n
    lg-a-at-n)))

(disable n-neq-k-rhoi5b)

(prove-lemma n-eq-k-rhoi5b (rewrite)
  (implies (and (listp 1)
    (listp g)
    (member k (nset (length 1)))
    (at 1 k 5)
    (lg-at-n k 1 g))
    (lg-at-n k (move 1 k 8) g))
  ((enable at lg-at-n lg-1-at-n lg-2-at-n
    lg-3-at-n)))

(disable n-eq-k-rhoi5bj)

(prove-lemma lg-at-rhoi5b (rewrite)
  (implies (and (listp 1)
    (listp g)
    (numberp nj)
    (member k (nset (length 1)))
    (at 1 k 5)
    (lg-at-n n 1 g))
    (lg-at-n n (move 1 k 8) g))
  ((enable n-neq-k-rhoi5b n-eq-k-rhoi5b)
    (use (n-neq-k-rhoi5b))
    (use (n-eq-k-rhoi5b))) j)

(disable lg-at-rhoi5bj)

(prove-lemma lg-rhoi5b (rewrite)
  (implies (and (listp 1)
    (listp g)
    (member k (nset (length 1)))
    (numberp nj)
    (at 1 k 5)
    (lg n 1 g))
    (lg n (move 1 k 8) g))
  ((enable lg-at-rhoi5b lg at)))

(disable lg-rhoi5b)

(prove-lemma rhoi5b-preserves-lg (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rhoi5b n k 1 g lp gpj)
    (lg n 1 g))
    (lg n lp gpj)
  ((enable lg-rhoi5b)))

;;;rhoi6
(prove-lemma n-neq-k-rhoi (rewrite)
  (implies (and (listp 1)
    (listp g)
    (numberp n)
    (member k (nset (length 1)))
    (not (equal k nj))
    (at 1 k 6)
    (lg-at-n n 1 g))
    (lg-at-n n (move 1 k 7) (move g k 2)))
  ((enable at lg-at-n lg-1-at-n lg-2-at-n
    lg-3-at-n)))

(disable n-neq-k-rhoi6)

(prove-lemma n-eq-k-rhoi (rewrite)
  (implies (and (listp 1)
    (listp g)
    (member k (nset (length 1)))
    (at 1 k 6)
    (lg-at-n k 1 g))
    (lg-at-n n (move 1 k 7) (move g k 2)))
  ((enable at lg-at-n lg-1-at-n lg-2-at-n
    lg-3-at-n)))

(disable n-eq-k-rhoi6j)

(prove-lemma lg-at-rhoi (rewrite)
  (implies (and (listp 1)
    (listp g)
    (numberp n)
    (member k (nset (length 1)))
    (at 1 k 6)
    (lg-at-n n 1 g))
    (lg-at-n n (move 1 k 7) (move g k 2)))
  ((enable n-neq-k-rhoi n-eq-k-rhoi6)
    (use (n-neq-k-rhoi6))
    (use (n-eq-k-rhoi6)))

(disable lg-at-rhoi6)

(prove-lemma lg-rhoi6 (rewrite)
  (implies (and (listp 1)
    (listp g)
    (member k (nset (length 1)))

```

```

    (numberp nj)
    (at 1 k 6)
    (lg n 1 g))
  ((enable lg-at-rhoi6 lg at)))

(disable lg-rhoi6)

(prove-lemma rhoi6-preserves-lg (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset nj))
    (rhoi6 n k 1 g lp gp)
    (lg n 1 gj)
    (lg n lp gp))
  ((enable lg-rhoi6)))

;;;rhoi7a
(prove-lemma n-neq-k-rhoi7a (rewrite)
  (implies (and (listp 1)
    (listp g)
    (numberp nj)
    (member k (nset (length 1)))
    (not (equal k nj))
    (at 1 k 7)
    (lg-at-n n 1 g))
    (lg-at-n n (move 1 k 8) g))
  ((enable at lg-at-n lg-1-at-n lg-2-at-n
    lg-3-at-n)))

(disable n-neq-k-rhoi7a)

(prove-lemma n-eq-k-rhoi7a (rewrite)
  (implies (and (listp 1)
    (listp g)
    (member k (nset (length 1)))
    (at 1 k 7)
    (lg-at-n k 1 gj)
    (lg-at-n k (move 1 k 8) gj)
    (lg-at-n lg-1-at-n lg-2-at-n
    lg-3-at-n)))

(disable n-eq-k-rhoi7a)

(prove-lemma lg-at-rhoi7a (rewrite)
  (implies (and (listp 1)
    (listp g)
    (numberp nj)
    (member k (nset (length 1)))
    (at 1 k 7)
    (lg-at-n n 1 g))
    (lg-at-n n (move 1 k 8) g))
  ((enable n-neq-k-rhoi7a n-eq-k-rhoi7a)
    (use (n-neq-k-rhoi7a))
    (use (n-eq-k-rhoi7a))))

(disable lg-at-rhoi7a)

(prove-lemma lg-rhoi7a (rewrite)
  (implies (and (listp 1)
    (listp g)
    (member k (nset (length 1)))
    (numberp n)
    (at 1 k 7)
    (lg n 1 g))
    (lg n (move 1 k 8) g)
  ((enable lg-at-rhoi7a lg at)))

(disable lg-rhoi7a)

(prove-lemma rhoi7a-preserves-lg (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rhoi7a n k 1 g lp gpj)
    (lg n 1 g))
    (lg n lp gp)
  ((enable lg-rhoi7a)))

;;;rhoi7b
(prove-lemma rhoi7b-preserves-lg (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rhoi7b n k 1 g lp gpj)
    (lg n 1 g))
    (lg n lp gp)
  ((enable rhoi7b)))

;;;rhoi8
(prove-lemma n-neq-k-rhoi (rewrite)
  (implies (and (listp 1)
    (listp g)
    (numberp n)
    (member k (nset (length 1)))
    (not (equal k n))
    (at 1 k 8)
    (lg-at-n n 1 gj)
    (lg-at-n n (move 1 k 9) (move g k 4)))
  ((enable at lg-at-n lg-1-at-n lg-2-at-n
    lg-3-at-n)))

```

```

(disable n-neq-k-rhoi8)

(prove-lemma n-eq-k-rhoi (rewrite)
  (implies (and (listp 1)
    (listp g)
    (member k (nset (length 1)))
    (at 1 k 8)
    (lg-at-n k 1 g))
    (lg-at-n n (move 1 k 9) (move g k 4) ))
  ((enable at lg-at-n lg-1-at-n lg-2-at-n
    lg-3-at-n)))

(disable n-eq-k-rhoi8)

(prove-lemma lg-at-rhoi8 (rewrite)
  (implies (and (listp 1)
    (listp g)
    (numberp n)
    (member k (nset (length 1)))
    (at 1 k 8)
    (lg-at-n n 1 g))
    (lg-at-n n (move 1 k 9) (move g k 4)))
  ((enable n-neq-k-rhoi n-eq-k-rhoi8)
  (use (n-neq-k-rhoi8))
  (use (n-eq-k-rhoi8))))

(disable lg-at-rhoi8)

(prove-lemma lg-rhoi8 (rewrite)
  (implies (and (listp 1)
    (listp g)
    (member k (nset (length 1)))
    (numberp n)
    (at 1 k 8)
    (lg n 1 g))
    (lg n (move 1 k 9) (move g k 4)))
  ((enable lg-at-rhoi8 lg at)))

(disable lg-rhoi8)

(prove-lemma rhoi8-preserves-lg (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rhoi8 n k 1 g lp gpj)
    (lg n 1 g))
    (lg n lp gpj))
  ((enable lg-rhoi8)))

:::rhoi9a
(prove-lemma n-neq-k-rhoi9a (rewrite)
  (implies (and (listp 1)
    (listp g)
    (numberp nj)
    (member k (nset (length 1)))
    (not (equal k n))
    (at 1 k 9)
    (lg-at-n n 1 g))
    (lg-at-n n (move 1 k 10) g))
  ((enable at lg-at-n lg-1-at-n lg-2-at-n
    lg-3-at-n)))

(disable n-neq-k-rhoi9aj)

(prove-lemma n-eq-k-rhoi9a (rewrite)
  (implies (and (listp 1)
    (listp g)
    (member k (nset (length 1)))
    (at 1 k 9)
    (lg-at-n k 1 g))
    (lg-at-n k (move 1 k 10) g))
  ((enable at lg-at-n lg-1-at-n lg-2-at-n
    lg-3-at-n)))

(disable n-eq-k-rhoi9aj)

(prove-lemma lg-at-rhoi9a (rewrite)
  (implies (and (listp 1)
    (listp g)
    (numberp n)
    (member k (nset (length 1)))
    (at 1 k 9)
    (lg-at-n n 1 g))
    (lg-at-n n (move 1 k 10) g))
  ((enable n-neq-k-rhoi9a n-eq-k-rhoi9a)
  (use (n-neq-k-rhoi9a))
  (use (n-eq-k-rhoi9a))))

(disable lg-at-rhoi9a)

(prove-lemma lg-rhoi9a (rewrite)
  (implies (and (listp 1)
    (listp g)
    (member k (nset (length 1)))
    (numberp n)
    (at 1 k 9)
    (lg n 1 g))
    (lg n (move 1 k 10) g))
  ((enable lg-at-rhoi9a lg at)))

```

```

((enable lg-at-rhoi9a lg at)))

(disable lg-rhoi9a)

(prove-lemma rhoi9a-preserves-lg (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rhoi9a n k 1 g lp gp)
    (lg n 1 g))
    (lg n lp gp))
  ((enable lg-rhoi9a)))

:::rhoi9b
(prove-lemma rhoi9b-preserves-lg (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset nj))
    (rhoi9b n k 1 g lp gpj)
    (lg n 1 g))
    (lg n lp gp))
  ((enable rhoi9b)))

:::rhoi10
(prove-lemma n-neq-k-rhoi (rewrite)
  (implies (and (listp 1)
    (listp g)
    (numberp nj)
    (member k (nset (length 1)))
    (not (equal k n))
    (at 1 k 10)
    (lg-at-n n 1 g))
    (lg-at-n n (move 1 k 11) g))
  ((enable at lg-at-n lg-1-at-n lg-2-at-n
    lg-3-at-n)))

(disable n-neq-k-rhoi10)

(prove-lemma n-eq-k-rhoi (rewrite)
  (implies (and (listp 1)
    (listp g)
    (member k (nset (length 1)))
    (at 1 k 10)
    (lg-at-n k 1 g))
    (lg-at-n k (move 1 k 11) g))
  ((enable at lg-at-n lg-1-at-n lg-2-at-n
    lg-3-at-n)))

(disable n-eq-k-rhoi10)

(prove-lemma lg-at-rhoi10 (rewrite)
  (implies (and (listp 1)
    (listp g)
    (numberp n)
    (member k (nset (length 1)))
    (at 1 k 10)
    (lg-at-n n 1 g))
    (lg-at-n n (move 1 k 11) g))
  ((enable n-neq-k-rhoi n-eq-k-rhoi10)
  (use (n-neq-k-rhoi10))
  (use (n-eq-k-rhoi10))))

(disable lg-at-rhoi10)

(prove-lemma lg-rhoi10 (rewrite)
  (implies (and (listp 1)
    (listp g)
    (member k (nset (length 1)))
    (numberp n)
    (at 1 k 10)
    (lg n 1 g))
    (lg n (move 1 k 11) g))
  ((enable lg-at-rhoi10 lg at)))

(disable lg-rhoi10)

(prove-lemma rhoi10-preserves-lg (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rhoi10 n k 1 g lp gp)
    (lg n 1 g))
    (lg n lp gp))
  ((enable lg-rhoi10))

:::rhoi11a
(prove-lemma n-neq-k-rhoi11a (rewrite)
  (implies (and (listp 1)
    (listp g)
    (numberp n)
    (member k (nset (length 1)))
    (not (equal k n))
    (at 1 k 11)
    (lg-at-n n 1 g))
    (lg-at-n n (move 1 k 12) g))
  ((enable at lg-at-n lg-1-at-n lg-2-at-n
    lg-3-at-n)))

(disable n-neq-k-rhoi11aj)

(prove-lemma n-eq-k-rhoi11a (rewrite)

```

```

(implies (and (listp 1)
              (listp g)
              (member k (nset (length 1)))
              (at 1 k 11)
              (lg-at-n k 1 g))
         (lg-at-n k (move 1 k 12) g))
  ((enable at lg-at-n lg-1-at-n lg-2-at-n
    lg-3-at-n)))

(disable n-eq-k-rhoilla)

(prove-lemma lg-at-rhoilla (rewrite)
  (implies (and (listp 1)
                (listp g)
                (numberp n)
                (member k (nset (length 1)))
                (at 1 k 11)
                (lg-at-n n 1 g))
           (lg-at-n n (move 1 k 12) g))
    ((enable n-neq-k-rhoilla n-eq-k-rhoilla)
     (use (n-neq-k-rhoilla))
     (use (n-eq-k-rhoilla)))))

(disable lg-at-rhoilla)

(prove-lemma lg-rhoilla (rewrite)
  (implies (and (listp 1)
                (listp g)
                (member k (nset (length 1)))
                (numberp n)
                (at 1 k 11)
                (lg n 1 g))
           (lg n (move 1 k 12) g))
    ((enable lg-at-rhoilla lg at))))

(disable lg-rhoilla)

(prove-lemma rhoilla-preserves-lg (rewrite)
  (implies (and (ws n 1 g)
                (member k (nset n))
                (rhoilla n k 1 g lp gp)
                (lg n 1 g))
           (lg n lp gp))
    ((enable lg-rhoilla))))

;;;rhoillb
(prove-lemma rhoillb-preserves-lg (rewrite)
  (implies (and (ws n 1 g)
                (member k (nset n))
                (rhoillb n k 1 g lp gp)
                (lg n 1 g))
           (lg n lp gp))
    ((enable rhoillb))))

;;;rhoil2
(prove-lemma n-neq-k-rhoil2 (rewrite)
  (implies (and (listp 1)
                (listp g)
                (numberp n)
                (member k (nset (length 1)))
                (not (equal k n))
                (at 1 k 12)
                (lg-at-n n 1 g))
           (lg-at-n n (move 1 k 0) (move g k 0)))
    ((enable at lg-at-n lg-1-at-n lg-2-at-n
      lg-3-at-n)))

(disable n-neq-k-rhoil2)

(prove-lemma n-eq-k-rhoil2 (rewrite)
  (implies (and (listp 1)
                (listp g)
                (member k (nset (length 1)))
                (at 1 k 12)
                (lg-at-n k 1 g))
           (lg-at-n n (move 1 k 0) (move g k 0)))
    ((enable at lg-at-n lg-1-at-n lg-2-at-n
      lg-3-at-n)))

(disable n-eq-k-rhoil2)

(prove-lemma lg-at-rhoil2 (rewrite)
  (implies (and (listp 1)
                (listp g)
                (numberp n)
                (member k (nset (length 1)))
                (at 1 k 12)
                (lg-at-n n 1 g))
           (lg-at-n n (move 1 k 0) (move g k 0)))
    ((enable n-neq-k-rhoil2 n-eq-k-rhoil2)
     (use (n-neq-k-rhoil2))
     (use (n-eq-k-rhoil2)))))

(disable lg-at-rhoil2)

(prove-lemma lg-rhoil2 (rewrite)
  (implies (and (listp 1)
                (listp g)
                (member k (nset (length 1)))
                (numberp n)
                (at 1 k 12)
                (lg n 1 g))
           (lg n lp gp))
    ((enable lg-rhoil2))))

(disable rhoi0-preserves-lg)
(disable rhoilla-preserves-lg)
(disable rhoilb-preserves-lg)
(disable rhoil2-preserves-lg)
(disable rhoi3a-preserves-lg)
(disable rhoi3b-preserves-lg)
(disable rhoi4-preserves-lg)
(disable rhoi5a-preserves-lg)
(disable rhoi5b-preserves-lg)
(disable rhoi6-preserves-lg)
(disable rhoi7a-preserves-lg)
(disable rhoi7b-preserves-lg)
(disable rhoi8-preserves-lg)
(disable rhoi9a-preserves-lg)
(disable rhoi9b-preserves-lg)
(disable rhoi10-preserves-lg)
(disable rhoilla-preserves-lg)
(disable rhoillb-preserves-lg)
(disable rhoil2-preserves-lg)

```

```

::: (exist-union lp n '(8 9 10 11 12)) and
::: (not (exist-union 1 n '(8 9 10 11 12)))
::: implies that k is the witness of
::: (exist-union lp n '(8 9 10 11 12)). This
::: proposition would have been more natural
::: if we had been able to prove:
::: (prove-lemma exist-18-12 (rewrite)
:::   (implies (and (ws n 1 g)
:::     (member k (nset n))
:::     (rho1 n k 1 g lp gp)
:::     (exist-union lp n '(8 9 10 11 12))
:::     (not (exist-union 1 n '(8 9 10 11 12))))
:::     (equal k (exist-union lp n '(8 9 10 11 12))))).
::: However Bmp refused to rewrite equal clause.

(prove-lemma exist-18-12 (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rho1 n k 1 g lp gp)
    (exist-union lp n '(8 9 10 11 12))
    (not (equal k (exist-union
      lp n '(8 9 10 11 12))))
    (exist-union 1 n '(8 9 10 11 12)))
    ((use (j-ex-18-12 (j (exist-union lp n '(8 9 10 11 12))))
      (use (ex-lp8-12-in-lp8-12))))))

::: If (exist-union lp n '(8 9 10 11 12)) and
::: (not (exist-union 1 n '(8 9 10 11 12))) hold,
::: then the k's entry of lp is between 8..12.
(prove-lemma k-in-lp8-12 (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rho1 n k 1 g lp gp)
    (exist-union lp n '(8 9 10 11 12))
    (not (exist-union 1 n '(8 9 10 11 12)))
    (union-at-n lp k '(8 9 10 11 12)))
    ((disable member-ex-union)
      (use (exist-18-12) (ex-lp8-12-in-lp8-12))))))

::: If k's entry in lp is between 8..12 and
::: k's entry of 1 is not between 8..12,
::: then k's entry of 1 is either 5 or 7.
(prove-lemma k-not-in-18-12-then-157 (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rho1 n k 1 g lp gp)
    (union-at-n lp k '(8 9 10 11 12))
    (not (union-at-n 1 k '(8 9 10 11 12)))
    (not (at 1 k 5)))
    ((enable rho1 union-at-n at))))

::: If k's entry in lp is between 8..12 and
::: (not (exist-union 1 n '(8 9 10 11 12))) holds,
::: then k's entry of 1 is either 5 or 7.
(prove-lemma k-in-157 (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rho1 n k 1 g lp gp)
    (union-at-n lp k '(8 9 10 11 12))
    (not (exist-union 1 n '(8 9 10 11 12)))
    (not (at 1 k 7)))
    ((at 1 k 5))
    ((use (k-not-in-18-12-then-157))
      (use (j-ex-18-12 (j k))))))

::: If (exist-union lp n '(8 9 10 11 12)) and
::: (not (exist-union 1 n '(8 9 10 11 12))) hold,
::: then the k's entry of 1 is between either 5 or 7.
(prove-lemma ex-k-in-157 (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rho1 n k 1 g lp gp)
    (exist-union lp n '(8 9 10 11 12))
    (not (exist-union 1 n '(8 9 10 11 12)))
    (not (at 1 k 7)))
    ((at 1 k 5))
    ((use (k-in-157) (k-in-lp8-12))))))

::: Auxiliary lemma for ex-cond-rho15.
(prove-lemma cond-rho15 (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rho1 n k 1 g lp gp)
    (union-at-n lp k '(8 9 10 11 12))
    (at 1 k 5))
    (not (exist-union g n '(1))))
  ((enable rho1 union-at-n at))))

::: If (exist-union lp n '(8 9 10 11 12)) and
::: (not (exist-union 1 n '(8 9 10 11 12))) and
::: the k's entry in 1 is 5 then
::: (not (exist-union g n '(1))) holds.
(prove-lemma ex-cond-rho15 (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rho1 n k 1 g lp gp)
    (exist-union lp n '(8 9 10 11 12))
    (not (exist-union 1 n '(8 9 10 11 12)))
    (at 1 k 5))
    (not (exist-union g n '(1))))
  ((enable rho1 union-at-n at))))

::: Auxiliary lemma for ex-cond-rho17.
(prove-lemma cond-rho17 (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rho1 n k 1 g lp gp)
    (union-at-n lp k '(8 9 10 11 12))
    (at 1 k 7))
    (exist-union g n '(4)))
  ((enable rho1 union-at-n at))))

::: If (exist-union lp n '(8 9 10 11 12)) and
::: (not (exist-union 1 n '(8 9 10 11 12))) and
::: the k's entry in 1 is 7, then
::: (exist-union g n '(4)) holds.
(prove-lemma ex-cond-rho17 (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rho1 n k 1 g lp gp)
    (exist-union lp n '(8 9 10 11 12))
    (not (exist-union 1 n '(8 9 10 11 12)))
    (at 1 k 7))
    (exist-union g n '(4)))
  ((use (cond-rho17) (k-in-lp8-12))))))

::: If (exist-union lp n '(8 9 10 11 12))
::: and (not (exist-union 1 n '(8 9 10 11 12))),
::: then (not (exist-union g n '(1))) holds.
(prove-lemma 15-only-lp8 (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rho1 n k 1 g lp gp)
    (not (exist-union 1 n '(8 9 10 11 12)))
    (lg n 1 g)
    (exist-union lp n '(8 9 10 11 12)))
    (not (exist-union g n '(1))))
  ((disable member-ex-union)
    (use (exist-18-12) (ex-k-in-157) (ex-cond-rho15)
      (ex-cond-rho17) (ex-if4))))))

::: If j is not equal to k and j's entry of 1
::: is neither 3 or 4, then j's entry of lp
::: is not 4.
(prove-lemma j-neq-k-j-not-in-lp4 (rewrite)
  (implies (and (ws n 1 g)
    (member j (nset n))
    (member k (nset n))
    (rho1 n k 1 g lp gp)
    (not (equal j k))
    (not (union-at-n 1 j '(3 4))))
    (not (at lp j 4)))
  ((use (lp4-then-un34))))))

::: If k's entry of 1 is neither 3 or 4, then
::: k's entry of lp is not 4.
(prove-lemma j-eq-k-j-not-in-lp4 (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rho1 n k 1 g lp gp)
    (not (union-at-n 1 k '(3 4))))
    (not (at lp k 4)))
  ((enable union-at-n rho1 at))))

::: If j's entry of 1 is neither 3 or 4, then
::: j's entry of lp is not 4.
(prove-lemma lp4-empty (rewrite)
  (implies (and (ws n 1 g)
    (member j (nset n))
    (member k (nset n))
    (rho1 n k 1 g lp gp)
    (not (union-at-n 1 j '(3 4))))
    (not (at lp j 4)))
  ((use (j-neq-k-j-not-in-lp4))
    (use (j-eq-k-j-not-in-lp4))))))

::: If (not (exist-union 1 n '(8 9 10 11 12)))
::: and (exist-union lp n '(8 9 10 11 12)) hold,
::: then there is no entry 4 in 1.
(prove-lemma 18-112-empty (rewrite)
  (implies (and (ws n 1 g)
    (member j (nset n))
    (member k (nset n))
    (rho1 n k 1 g lp gp)
    (not (exist-union 1 n '(8 9 10 11 12)))
    (lg n 1 g)
    (a0 n lp j))
    ((enable a0)
      (use (lp4-empty) (l34-empty) (15-only-lp8))))))

::: If (exist-union g n '(3 4)) holds and
::: the k's entry in 1 is not 4, then
::: the k's entry in lp is not 4 either.
::: (Doorway is locked.)
(prove-lemma dwy-lckd (rewrite)

```

```

(implies (and (ws n 1 g)
              (member k (nset n))
              (rhoi n k 1 g lp gp)
              (exist-union g n '(3 4))
              (not (at 1 k 4))))
  (not (at lp k 4)))
(enable rhoi at))

;;; If (exist-union 1 n '(8 9 10 11 12))
;;; holds and j is equal to k, then
;;; j's entry in lp is not 4.
(prove-lemma j-eq-k-l8-l12-nonemp (rewrite)
  (implies (and (ws n 1 g)
                (member k (nset n))
                (rhoi n k 1 g lp gp)
                (exist-union 1 n '(8 9 10 11 12))
                (a0 n 1 k)
                (a1 n 1 g))
            (a0 n lp k))
    (enable a0 al)
    (use (dwy-lckd) (int-8-12-3-4-then-un34))))

;;; If (exist-union 1 n '(8 9 10 11 12))
;;; holds and j is not equal to k, then
;;; the j's entry in lp is not 4.
(prove-lemma j-neq-k-l8-l12-nonemp (rewrite)
  (implies (and (ws n 1 g)
                (member j (nset n))
                (member k (nset n))
                (rhoi n k 1 g lp gp)
                (not (equal j k))
                (a0 n 1 j)
                (exist-union 1 n '(8 9 10 11 12)))
            (a0 n lp j))
    (enable a0 at))

;;; If (exist-union 1 n '(8 9 10 11 12))
;;; holds then there is no entry 4 in lp.
;;; The order of the use hints is critical.
;;; Change the order and we fail.
(prove-lemma l8-l12-nonemp (rewrite)
  (implies (and (ws n 1 g)
                (member j (nset n))
                (member k (nset n))
                (rhoi n k 1 g lp gp)
                (exist-union 1 n '(8 9 10 11 12))
                (a0 n 1 j)
                (a1 n 1 g))
            (a0 n lp j))
    (use (j-neq-k-l8-l12-nonemp))
    (use (j-eq-k-l8-l12-nonemp))))

;;; If (exist-union lp n '(8 9 10 11 12))
;;; holds, then there is no entry 4 in lp.
(prove-lemma rho-preserves-a0 ()
  (implies (and (ws n 1 g)
                (member j (nset n))
                (member k (nset n))
                (rhoi n k 1 g lp gp)
                (lg n 1 g)
                (a0 n 1 j)
                (a1 n 1 g))
            (a0 n lp j))
    (use (l8-l12-nonemp) (l8-l12-empty))))

```

```

(not (union-at-n 1 k '(8 9 10 11 12)))
  (al n 1 g))
  (exist-intersect-E12-3-4 n lp gp))
(enable al)
  (use (int-k-not-ex-int))
  (use (intersect-8-12-3-4-then-8-12))))

;; k-in-18-11
;; If the k's entry in 1 is between 8 and 11,
;; then the k's entry in lp is between 9 and 12.
;; We need rho-preserves-lg.
(prove-lemma 18-11-k-in-lp9-12 (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rho1 n k 1 g lp gp)
    (union-at-n 1 k '(8 9 10 11)))
    (union-at-n lp k '(9 10 11 12)))
  (enable rho1 union-at-n at)))

;; If the k's entry in 1 is between 8 and 11,
;; then the k's entry in lp is between 8 and 12
;; and the entry in gp is either 3 or 4.
;; 18-11-k-in-lp9-12, un9-12-then-un8-12 and
;; rho-preserves-lg are used.
(prove-lemma 1m-al-k-in-18-11-nep-18-12 (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rho1 n k 1 g lp gp)
    (union-at-n 1 k '(8 9 10 11))
    (lg n 1 g))
    (and (union-at-n lp k '(8 9 10 11 12))
      (union-at-n gp k '(3 4))))
  ((use (if4 (j k) (1 lp) (g gp)))
  (use (lp4-then-un34 (lp gp)))))

;; If (exist-union lp n '(8 9 10 11 12)) holds,
;; and the k's entry in 1 is between 8 and 11 then
;; (exist-intersect-8-12-3-4 n lp gp) holds.
;; j \eq k and k \in 18-11
(prove-lemma al-k-in-18-11-nep-18-12 (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rho1 n k 1 g lp gp)
    (lg n 1 g))
    (exist-union lp n '(8 9 10 11 12))
    (union-at-n 1 k '(8 9 10 11)))
    (exist-intersect-E12-3-4 n lp gp))
  ((use (un8-12-and-un34-then-int (j k)))
  (use (int-wtn (j k)))
  (use (1m-al-k-in-18-11-nep-18-12))))

;; If the k's entry in 1 is 12 then the k's entry in 1 is 0.
(prove-lemma k-in-lp0 (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rho1 n k 1 g lp gp)
    (at 1 k 12))
    (at lp k 0))
  (enable rho1 at)))

;; If (exist-union lp n '(8 9 10 11 12)) holds
;; and k's entry in 1 is 12, then k is not the
;; witness of (exist-union lp n '(8 9 10 11 12)).
(prove-lemma k-not-ex-lp8-12 (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rho1 n k 1 g lp gp)
    (exist-union lp n '(8 9 10 11 12))
    (at 1 k 12))
    (not (equal k
      (exist-union lp n '(8 9 10 11 12)))))
  ((use (ex-lp8-12-not-in-lp0) (k-in-lp0)))))

;; If the k's entry in lp is 8,
;; then k's entry in 1 is 12, then k is not the
;; witness of (exist-union lp n '(8 9 10 11 12)).
(prove-lemma lp8-k-in-157 (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rho1 n k 1 g lp gp)
    (at lp k 8))
    (union-at-n 1 k '(5 7)))
  (enable rho1 union-at-n at)))

;; If the k's entry in lp is 8,
;; then k's entry in 1 is between 5 and 12.
(prove-lemma k-in-lp8-then-15-12 (rewrite)
  (implies (and (ws n 1 g)
    (rho1 n k 1 g lp gp)
    (member k (nset n))
    (at lp k 8))
    (union-at-n 1 k '(5 6 7 8 9 10 11 12)))
  ((use (un57-then-un5-12) (lp8-k-in-157)))))

;; If the k's entry in lp is between 9 and 12,
;; then the k's entry in 1 is between 8 and 11.
(prove-lemma lp9-12-k-in-18-11 (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rho1 n k 1 g lp gp)
    (union-at-n 1 k '(8 9 10 11 12)))
    (union-at-n lp k '(9 10 11 12)))
  (enable rho1 union-at-n at)))

```

```

(rho1 n k 1 g lp gp)
(union-at-n lp k '(8 9 10 11 12)))
(union-at-n 1 k '(8 9 10 11)))
(enable rho1 union-at-n at)))

;;:If the k's entry in lp is between 9 and 12,
;;:then the k's entry in 1 is between 5 and 12.
(prove-lemma k-in-lp9-12-then-15-12 (rewrite)
  (implies (and (ws n 1 g)
    (rho1 n k 1 g lp gp)
    (member k (nset n))
    (union-at-n lp k '(8 9 10 11 12)))
    (union-at-n 1 k '(5 6 7 8 9 10 11 12)))
    ((use (un8-11-then-un5-12))
      (use (lp9-12-k-in-18-11))))))

;;:If the k's entry in lp is between 8 and 12,
;;:then the k's entry in 1 is between 5 and 12.
(prove-lemma k-in-15-12 (rewrite)
  (implies (and (ws n 1 g)
    (rho1 n k 1 g lp gp)
    (member k (nset n))
    (union-at-n lp k '(8 9 10 11 12)))
    (union-at-n 1 k '(5 6 7 8 9 10 11 12)))
    ((use (k-in-lp9-12-or-lp8))
      (use (k-in-lp9-12-then-15-12))
      (use (k-in-lp8-then-15-12))))))

;;:If (exist-union lp n '(8 9 10 11 12)) holds
;;:and k is not its witness, then the witness has
;;:its entry in 1 between 5 and 12.
;;:ex-lp8-12-in-lp8-12, member-ex-union used.
(prove-lemma k-neq-ex-lp8-12-in-15-12 (rewrite)
  (implies (and (ws n 1 g)
    (rho1 n k 1 g lp gp)
    (member k (nset n))
    (exist-union lp n '(8 9 10 11 12))
    (not (equal k (exist-union lp
      n '(8 9 10 11 12))))))
    (union-at-n 1 (exist-union lp n
      '(8 9 10 11 12)) '(5 6 7 8 9 10 11 12)))
    ((use (un8-12-then-un5-12)
      (j (exist-union lp n '(8 9 10 11 12)))))))

;;:If (exist-union lp n '(8 9 10 11 12)) holds and
;;:the witness has its entry in lp between 8 and 12,
;;:then its entry in 1 is between 5 and 12.
;;:ex-lp8-12-in-lp8-12, member-ex-union used.
(prove-lemma ex-lp8-12-then-15-12 (rewrite)
  (implies (and (ws n 1 g)
    (rho1 n k 1 g lp gp)
    (member k (nset n))
    (exist-union lp n '(8 9 10 11 12)))
    (union-at-n 1 (exist-union lp n
      '(8 9 10 11 12)) '(5 6 7 8 9 10 11 12)))
    ((use (k-neq-ex-lp8-12-in-15-12))
      (use (k-in-15-12))))))

;;:If (exist-union lp n '(8 9 10 11 12)) holds
;;:and k is not the witness of
;;:(exist-union lp n '(8 9 10 11 12)). then
;;:the witness has its entry 4 in gp.
(prove-lemma ex-lp8-12-in-gp4 (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rho1 n k 1 g lp gp)
    (at 1 k 12)
    (a3-at-n1-n2 k (exist-union lp n
      '(8 9 10 11 12)) 1 g)
    (not (equal k (exist-union lp n
      '(8 9 10 11 12))))))
    (exist-union lp n '(8 9 10 11 12)))
    (at gp (exist-union lp n '(8 9 10 11 12)) 4))
    (enable a3-at-n1-n2 at)
    (use (ex-lp8-12-then-15-12))))))

;;:If (exist-union lp n '(8 9 10 11 12)) holds and
;;:k's entry in 1 is 12 then the witness has its
;;:entry in lp between 8 and 12 and in gp either 3 or 4.
(prove-lemma lm-al-k-in-112-nep-18-12 (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rho1 n k 1 g lp gp)
    (exist-union lp n '(8 9 10 11 12))
    (at 1 k 12)
    (a3-at-n1-n2 k (exist-union lp n
      '(8 9 10 11 12)) 1 g)
    (union-at-n gp (exist-union lp n
      '(8 9 10 11 12)) '(3 4))))
    ((use (k-not-ex-lp8-12) (ex-lp8-12-in-gp4))
      (use (lp4-then-un34) (lp gp)
        (j (exist-union lp n '(8 9 10 11 12))))))
    (use (ex-lp8-12-in-lp8-12))))))

;;:If (exist-union lp n '(8 9 10 11 12)) holds
;;:and k's entry in 1 is 12, then
;;:(exist-intersect-E-12-3-4 n lp gp) holds.
(prove-lemma al-k-in-112-nep-18-12 (rewrite)

```

```

(implies (and (ws n 1 g)
  (member k (nset n))
  (rho1 n k 1 g lp gp)
  (exist-union lp n '(8 9 10 11 12))
  (at 1 k 12)
  (a3-at-n1-n2 k (exist-union lp n
    '(8 9 10 11 12)) 1 g)
  (exist-intersect-E-12-3-4 n lp gp)))
  ((use (int-wtn
    (j (exist-union lp n '(8 9 10 11 12))))))
    (use (lm-al-k-in-112-nep-18-12))
    (use (un8-12-and-un34-then-int
      (j (exist-union lp n '(8 9 10 11 12)))))))

;;:Auxiliary lemma for al-nep-18-12.
;;:We have an instance of a3 in the lemma.
(prove-lemma lml-al-nep-18-12 (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rho1 n k 1 g lp gp)
    (lg n 1 g)
    (al n 1 g)
    (a3-at-n1-n2 k (exist-union lp n
      '(8 9 10 11 12)) 1 g)
    (exist-union lp n '(8 9 10 11 12))
    (exist-union 1 n '(8 9 10 11 12)))
    (exist-intersect-E-12-3-4 n lp gp)))
    ((use (case-k))
      (use (al-k-not-in-18-12-nep-18-12))
      (use (al-k-in-18-11-nep-18-12))
      (use (al-k-in-112-nep-18-12))))))

(prove-lemma lm-al-nep-18-12 (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rho1 n k 1 g lp gp)
    (lg n 1 g)
    (al n 1 g)
    (a3 n n 1 g)
    (exist-union 1 n '(8 9 10 11 12))
    (exist-union lp n '(8 9 10 11 12)))
    (exist-intersect-E-12-3-4 n lp gp))
    ((use (lml-al-nep-18-12) (a3-ex-a3-at-n1-n2))))))

;;:If (exist-union lp n '(8 9 10 11 12)) and
;;:(exist-union 1 n '(8 9 10 11 12)) hold,
;;:then so does (exist-intersect-E-12-3-4 n lp gp).
(prove-lemma al-nep-18-12 (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rho1 n k 1 g lp gp)
    (lg n 1 g)
    (al n 1 g)
    (a3 n n 1 g)
    (exist-union 1 n '(8 9 10 11 12)))
    (al n lp gp))
    (enable al)))

;;:If (exist-union lp n '(8 9 10 11 12)) holds,
;;:then so does (exist-intersect-E-12-3-4 n lp gp).
(prove-lemma rho-preserves-al ()
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rho1 n k 1 g lp gp)
    (lg n 1 g)
    (al n 1 g)
    (a3 n n 1 g)
    (al n lp gp))
    ((use (al-nep-18-12))
      (use (al-ep-18-12))))))

```

```

;* i-eq-k-j-neq-k

;;;If the k's entry in lp is between 10 and 12
;;;and the k's entry in 1 is between 10 and 12,
;;; then (phi9 k n g) holds.
(prove-lemma k-in-110-11-or-phi9 (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rho1 n k 1 g lp gp)
    (union-at-n lp k '(10 11 12))
    (not (union-at-n 1 k '(10 11))))
    (phi9 k n g))
  ((enable rho1 union-at-n at)))

;;;If j is less than k and (phi9 k n g) holds,
;;;then the j's entry in g is either 0 or 1.
(prove-lemma phi9-j-in-g01 (rewrite)
  (implies (and (member j (nset n))
    (lessp j k)
    (phi9 k n g))
    (union-at-n g j '(0 1)))
  ((enable nset phi9)))

;;;If j is less than k and (phi9 k n g) holds,
;;;then the j's entry in lp is not between 5 and 12.
;;;lp-same-1-not is used.
(prove-lemma case-k-in-phi9 (rewrite)
  (implies (and (ws n 1 g)
    (member j (nset n))
    (member k (nset n))
    (rho1 n k 1 g lp gp)
    (not (equal j k))
    (lessp j k)
    (lg n 1 g)
    (phi9 k n g))
    (not (union-at-n lp j
      '(5 6 7 8 9 10 11 12))))
  ((use (phi9-j-in-g01) (if1))))

;;;If j is not equal to k and the k's entry in 1 is
;;;either 10 or 11, then the j's entry in lp is not
;;;between 5 and 12.
(prove-lemma case-k-in-110-11 (rewrite)
  (implies (and (ws n 1 g)
    (member j (nset n))
    (member k (nset n))
    (rho1 n k 1 g lp gp)
    (a2-at-n1-n2 k j 1)
    (not (equal j k))
    (union-at-n 1 k '(10 11)))
    (not (union-at-n lp j
      '(5 6 7 8 9 10 11 12))))
  ((enable a2-at-n1-n2)
  (use (un10-11-then-un10-12))))

;;;Auxiliary lemma for lm-i-eq-k-j-neq-k with
;;;(a2-at-n1-n2 k j 1).
(prove-lemma lml-i-eq-k-j-neq-k (rewrite)
  (implies (and (ws n 1 g)
    (member j (nset n))
    (member k (nset n))
    (rho1 n k 1 g lp gp)
    (not (equal j k))
    (lessp j k)
    (lg n 1 g)
    (a2-at-n1-n2 k j 1)
    (union-at-n lp k '(10 11 12)))
    (not (union-at-n lp j '(5 6 7 8 9 10 11 12))))
  ((use (k-in-110-11-or-phi9)
  (use (case-k-in-110-11))
  (use (case-k-in-phi9)))))

;;;If j is less than k and the k's entry in lp is
;;;between 10 and 12, then the j's entry in lp is
;;;not between 5 and 12.
(prove-lemma lm-i-eq-k-j-neq-k (rewrite)
  (implies (and (ws n 1 g)
    (member j (nset n))
    (member k (nset n))
    (rho1 n k 1 g lp gp)
    (not (equal j k))
    (lessp j k)
    (lg n 1 g)
    (a2-at-n1-n2 k j 1)
    (a2-at-n1-n2 k j lp))
    (enable a2-at-n1-n2)))

;;;If j is less than k,
;;;then (a2-at-n1-n2 k j lp) holds.
(prove-lemma i-eq-k-j-neq-k (rewrite)
  (implies (and (ws n 1 g)
    (member j (nset n))
    (member k (nset n))
    (rho1 n k 1 g lp gp)
    (not (equal j k))
    (lessp j k)
    (lg n 1 g)
    (a2 n n 1))

    (a2-at-n1-n2 k j lp))
  ((use (lm-i-eq-k-j-neq-k)
  (use (a2-i-j-a2-at-n1-n2 (i k))))))

;* j-eq-k-i-neq-k

;;;If the k's entry in 1 is not 4 and the k's entry in lp
;;;is between 5 and 7, then the k's entry in 1 is
;;;between 5 and 7.
(prove-lemma k-in-lp5-7-not-14-then-15-7 (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rho1 n k 1 g lp gp)
    (not (at 1 k 4))
    (union-at-n lp k '(5 6 7)))
    (union-at-n 1 k '(5 6 7)))
  ((enable union-at-n at rho1)))

;;;If the k's entry in lp is between 5 and 7 then
;;;the k's entry in 1 is certainly between 5 and 12.
(prove-lemma k-in-lp5-7-then-15-11 (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rho1 n k 1 g lp gp)
    (not (at 1 k 4))
    (union-at-n lp k '(5 6 7)))
    (union-at-n 1 k '(5 6 7 8 9 10 11)))
  ((use (k-in-lp5-7-not-14-then-15-7)
  (use (un5-7-then-un5-11))))))

;;;If the k's entry in lp is 8,
;;;then the k's entry in 1 is between 5 and 11.
(prove-lemma k-in-lp8-then-15-11 (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rho1 n k 1 g lp gp)
    (at lp k 8))
    (union-at-n 1 k '(5 6 7 8 9 10 11)))
  ((use (un5-7-then-un5-11) (lp8-k-in-15-7))))

;;;If the k's entry in lp is between 9 and 12,
;;;then the k's entry in 1 is between 5 and 12.
(prove-lemma k-in-lp9-12-then-15-11 (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rho1 n k 1 g lp gp)
    (union-at-n lp k '(9 10 11 12)))
    (union-at-n 1 k '(5 6 7 8 9 10 11)))
  ((use (lp9-12-k-in-18-11) (un8-11-then-un5-11))))

;;;If the k's entry in 1 is not 4 and the k's entry in lp is
;;;between 5 and 12, then the k's entry in 1 is
;;;between 5 and 11.
(prove-lemma k-in-15-11 (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rho1 n k 1 g lp gp)
    (not (at 1 k 4))
    (union-at-n lp k '(5 6 7 8 9 10 11 12)))
    (union-at-n 1 k '(5 6 7 8 9 10 11)))
  ((use (k-in-lp5-7-or-lp8-or-lp9-12)
  (use (k-in-lp8-then-15-11))
  (use (k-in-lp5-7-then-15-11))
  (use (k-in-lp9-12-then-15-11))))))

;;;If the k's entry in 1 is not 4, and the k's entry
;;;in lp is not between 5 and 12, then the k's entry
;;;in lp is not between 5 and 12.
(prove-lemma k-not-in-14 (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rho1 n k 1 g lp gp)
    (not (at 1 k 4))
    (not (union-at-n 1 k '(5 6 7 8 9 10 11 12))))
    (not (union-at-n lp k '(5 6 7 8 9 10 11 12))))
  ((use (un5-11-then-un5-12) (k-in-15-11))))

;;;If a0 holds, and the k's entry in 1 is not
;;;between 5 and 12, then the k's entry in lp is not
;;;between 5 and 12.
(prove-lemma k-not-in-lp5-12 (rewrite)
  (implies (and (ws n 1 g)
    (member i (nset n))
    (member k (nset n))
    (rho1 n k 1 g lp gp)
    (a0 n 1 k)
    (union-at-n 1 i '(10 11 12))
    (not (union-at-n 1 k '(5 6 7 8 9 10 11 12))))
    (not (union-at-n lp k
      '(5 6 7 8 9 10 11 12))))
  ((enable a0)
  (use (un10-12-then-un8-12) (k-not-in-14))))

;;;Auxiliary lemma for lm-i-neq-k-j-eq-k.
;;;There is (a2-at-n1-n2 i k 1) in the lemma.
(prove-lemma lml-i-neq-k-j-eq-k (rewrite)
  (implies (and (ws n 1 g)
    (member i (nset n))

```

```

(member k (nset n))
(rho1 n k 1 g lp gp)
(not (equal i k))
(lessp k i)
(a0 n 1 k)
(a2-at-n1-n2 i k 1)
(union-at-n lp i '(10 11 12)))
(not (union-at-n lp k
      '(5 6 7 8 9 10 11 12))))
(enable a2-at-n1-n2) (use (k-not-in-lp5-12))))

;;;If k is less than i and the i's entry in lp is
;;;between 10 and 12, then the k's entry in lp is
;;;between 5 and 12.
(prove-lemma lm-i-neq-k-j-eq-k (rewrite)
  (implies (and (ws n 1 g)
    (member i (nset n))
    (member k (nset n))
    (rho1 n k 1 g lp gp)
    (not (equal i k))
    (lessp k i)
    (a0 n 1 k)
    (a2-at-n1-n2 i k 1))
    (a2-at-n1-n2 i k lp))
  (enable a2-at-n1-n2)
  (use (lm1-i-neq-k-j-eq-k))))

;;;If k is less than i then (a2-at-n1-n2 i k lp) holds.
(prove-lemma i-neq-k-j-eq-k (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (member i (nset n))
    (rho1 n k 1 g lp gp)
    (not (equal i k))
    (lessp k i)
    (a0 n 1 k)
    (a2 n n 1))
    (a2-at-n1-n2 i k lp))
  ((use (lm-i-neq-k-j-eq-k))
   (use (a2-i-j-a2-at-n1-n2 (j k))))))

;* i-j-neq-k

;;;If i and j are not equal to k and the i's entry in lp is
;;;between 10 and 12, then the j's entry in lp is
;;;between 5 and 12.
(prove-lemma lm-i-j-neq-k (rewrite)
  (implies (and (ws n 1 g)
    (member i (nset n))
    (member j (nset n))
    (member k (nset n))
    (rho1 n k 1 g lp gp)
    (not (equal i k))
    (not (equal j k))
    (lessp j i)
    (a2-at-n1-n2 i j 1))
    (a2-at-n1-n2 i j lp))
  (enable a2-at-n1-n2)))

;;;If i and j are not equal to k,
;;; then (a2-at-n1-n2 i j lp) holds.
(prove-lemma i-j-neq-k (rewrite)
  (implies (and (ws n 1 g)
    (member i (nset n))
    (member j (nset n))
    (member k (nset n))
    (rho1 n k 1 g lp gp)
    (not (equal i k))
    (not (equal j k))
    (lessp j i)
    (a2 n n 1))
    (a2-at-n1-n2 i j lp))
  ((use (lm-i-j-neq-k))
   (use (a2-i-j-a2-at-n1-n2))))

;;;If i is not equal to k and j is less than i,
;;;then (a2-at-n1-n2 i j lp) holds.
;;;The order of the hints is crucial.
(prove-lemma i-neq-k (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (member i (nset n))
    (member j (nset n))
    (rho1 n k 1 g lp gp)
    (not (equal i k))
    (lessp j i)
    (a0 n 1 k)
    (a2 n n 1))
    (a2-at-n1-n2 i j lp))
  ((use (i-j-neq-k) (i-neq-k-j-eq-k))))

;;;If j is less than k then (a2-at-n1-n2 k j lp) holds.
(prove-lemma i-eq-k (rewrite)
  (implies (and (ws n 1 g)
    (member j (nset n))
    (member k (nset n))
    (rho1 n k 1 g lp gp)
    (lessp j k)

```

```

(lg n 1 g)
(a2 n n 1))
(a2-at-n1-n2 k j lp))
((use (i-eq-k-j-neq-k)))

;;;If i is less than j then (a2-at-n1-n2 k j lp) holds.
;;;Again the order of the hints is crucial.
(prove-lemma rho-preserves-a2 ()
  (implies (and (ws n 1 g)
    (member k (nset n))
    (member i (nset n))
    (member j (nset n))
    (rho1 n k 1 g lp gp)
    (lessp j i)
    (lg n 1 g)
    (a0 n 1 k)
    (a2 n n 1))
    (a2-at-n1-n2 i j lp))
  ((use (i-neq-k) (i-eq-k))))

```

```
;* j-eq-k-i-neq-k
```

```
;;If the i's entry in 1 is 12 and the k's entry in lp is
;;between 5 and 12 then the k's entry in 1 is between 9
;;and 11.
(prove-lemma lm-k-in-19-11 (rewrite)
  (implies (and (ws n 1 g)
    (member i (nset n))
    (member k (nset n))
    (rhoi n k 1 g lp gp)
    (lg n 1 g)
    (a0 n 1 k)
    (a3-at-n1-n2 i k 1 g)
    (at 1 i 12)
    (union-at-n 1 k '(5 6 7 8 9 10 11)))
    (union-at-n 1 k '(9 10 11)))
  ((enable a3-at-n1-n2)
  (use (un5-11-then-un5-12))
  (use (k-in-15-11-g4-then-19-11))))

;;If i is not equal to k, the i's entry in 1 is 12,
;;and the k's entry in lp is between 5 and 12,
;;then the k's entry in 1 is between 9 and 11.
(prove-lemma k-in-19-11 (rewrite)
  (implies (and (ws n 1 g)
    (member i (nset n))
    (member k (nset n))
    (rhoi n k 1 g lp gp)
    (lg n 1 g)
    (a0 n 1 k)
    (a3-at-n1-n2 i k 1 g)
    (at 1 i 12)
    (union-at-n lp k '(5 6 7 8 9 10 11 12)))
    (union-at-n 1 k '(9 10 11)))
  ((enable a0)
  (use (lm-k-in-19-11) (l12-then-un8-12) (k-in-15-11))))

;;If the k's entry in lp is between 9 and 11
;;then the k's entry in lp is between 9 and 12.
(prove-lemma k-in-lp9-12 (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rhoi n k 1 g lp gp)
    (union-at-n 1 k '(9 10 11)))
    (union-at-n lp k '(9 10 11 12)))
  ((enable union-at-n at rhoi)))

;;Auxiliary lemma for lm-a3-i-neq-k-j-eq-k.
;;There is (a3-at-n1-n2 i k 1 g) in the lemma.
(prove-lemma lml-a3-i-neq-k-j-eq-k (rewrite)
  (implies (and (ws n 1 g)
    (member i (nset n))
    (member k (nset n))
    (rhoi n k 1 g lp gp)
    (not (equal i k))
    (lg n 1 g)
    (lg n lp gp)
    (a0 n 1 k)
    (a3-at-n1-n2 i k 1 g)
    (at 1 i 12)
    (union-at-n lp k '(5 6 7 8 9 10 11 12)))
    (at gp k 4))
  ((disable rho-preserves-lg)
  (use (k-in-19-11) (k-in-lp9-12))
  (use (if4 (j k) (1 lp) (g gp)))))

;;If i is not equal to k, the i's entry in lp is 12,
;;and the k's entry in lp is between 5 and 12,
;;then the k's entry in gp is 4.
(prove-lemma lm-a3-i-neq-k-j-eq-k (rewrite)
  (implies (and (ws n 1 g)
    (member i (nset n))
    (member k (nset n))
    (rhoi n k 1 g lp gp)
    (not (equal i k))
    (lg n 1 g)
    (a0 n 1 k)
    (a3-at-n1-n2 i k 1 g)
    (a3-at-n1-n2 i k lp gp))
    (enable a3-at-n1-n2)
    (use (lml-a3-i-neq-k-j-eq-k))))

;;If i is not equal to k,
;;then (a3-at-n1-n2 i k lp gp) holds.
;;The order of the hypotheses is crucial.
(prove-lemma a3-i-neq-k-j-eq-k (rewrite)
  (implies (and (ws n 1 g)
    (member i (nset n))
    (member k (nset n))
    (rhoi n k 1 g lp gp)
    (lg n 1 g)
    (a0 n 1 k)
    (a3 n 1 g)
    (not (equal i k)))
    (a3-at-n1-n2 i k lp gp))
  ((use (lm-a3-i-neq-k-j-eq-k))
  (use (a3-i-j-a3-at-n1-n2 (j k))))))
```

```
;* i-eq-k-j-neq-k
```

```
;;If the k's entry in lp is 12 then (phill k n g) holds.
(prove-lemma cond-rho11 (rewrite)
  (implies (and (ws n 1 g)
    (member k (nset n))
    (rhoi n k 1 g lp gp)
    (at lp k 12))
    (phill k n g))
  ((enable rhoi at)))

;;If the k's entry in 1 is between 10 and 12,
;;the j's entry in 1 is between 5 and 12, and
;;(a2-at-n2 k n 1) holds, then k is less than j.
;;Because Bmp does not rewrite the clause
;;(lessp k j), we take its contrapositive.
(prove-lemma k-lt-j (rewrite)
  (implies (and (member j (nset n))
    (not (equal j k))
    (union-at-n 1 k '(10 11 12))
    (union-at-n 1 j '(5 6 7 8 9 10 11 12))
    (not (lessp k j)))
    (not (a2-at-n2 k n 1)))
  ((enable nset a2-at-n2 a2-at-n1-n2)))

;;If k is less than j and (phill k n g) holds,
;;then the j's entry in g is either 2 or 3.
(prove-lemma phill-j-not-in-g23 (rewrite)
  (implies (and (member j (nset n))
    (lessp k j)
    (phill k n g))
    (not (union-at-n g j '(2 3))))
  ((enable nset phill)))

;;If j is not equal to k, (a2-at-n2 k n 1), (phill k n g)
;;the k's entry in 1 is between 10 and 12 and
;;the j's entry in 1 is between 5 and 12,
;;then the j's entry in g is either 2 or 3.
(prove-lemma lml-j-not-in-g23 (rewrite)
  (implies (and (member j (nset n))
    (not (equal j k))
    (a2-at-n2 k n 1)
    (phill k n g)
    (union-at-n 1 k '(10 11 12))
    (union-at-n 1 j '(5 6 7 8 9 10 11 12)))
    (not (union-at-n g j '(2 3))))
  ((use (k-lt-j) (phill-j-not-in-g23))))

;;If j is not equal to k, the k's entry in 1 is
;;between 10 and 12, the k's entry in lp is 12 and
;;the j's entry in 1 is between 5 and 12,
;;then the j's entry in g is either 2 or 3.
(prove-lemma lm2-j-not-in-g23 (rewrite)
  (implies (and (ws n 1 g)
    (member j (nset n))
    (member k (nset n))
    (rhoi n k 1 g lp gp)
    (not (equal j k))
    (a2-at-n2 k n 1)
    (at lp k 12)
    (at 1 k 11)
    (union-at-n 1 j '(5 6 7 8 9 10 11 12)))
    (not (union-at-n g j '(2 3))))
  ((use (cond-rho11) (lml-j-not-in-g23)
  (Ill-then-un10-12))))

;;If j is not equal to k, the k's entry in lp is 12,
;;and the j's entry in 1 is between 5 and 12,
;;then the j's entry in g is either 2 or 3.
(prove-lemma j-not-in-g23 (rewrite)
  (implies (and (ws n 1 g)
    (member j (nset n))
    (member k (nset n))
    (rhoi n k 1 g lp gp)
    (not (equal j k))
    (a2 n n 1)
    (at 1 k 11)
    (at lp k 12)
    (union-at-n 1 j '(5 6 7 8 9 10 11 12)))
    (not (union-at-n g j '(2 3))))
  ((use (lm2-j-not-in-g23) (a2-n-a2-at-n2))))

(prove-lemma j-in-g4 (rewrite)
  (implies (and (member j (nset n))
    (lg n 1 g)
    (not (union-at-n g j '(2 3)))
    (union-at-n 1 j '(5 6 7 8 9 10 11 12)))
    (at g j 4))
  ((enable at)
  (use (if4) (15-12-eq-15-8-or-19-12) (if3))))

;;un9-12-then-un8-12, if3, j-not-in-g23,
;;15-12-eq-15-8-or-19-12, un8-12-then-un5-12,
;;and j-in-g4 are used.
;;If j is not equal to k, the k's entry in lp is 12,
;;the j's entry in 1 is between 5 and 12,
;;then the j's entry in g is 4.
```

```

(prove-lemma a3-j-in-15-12 (rewrite)
  (implies (and (ws n 1 q)
    (member j (nset n))
    (member k (nset n))
    (rhoi n k 1 q lp qp)
    (not (equal j k))
    (lg n 1 q)
    (a2 n n 1)
    (at 1 k 11)
    (at lp k 12)
    (union-at-n 1 j '(5 6 7 8 9 10 11 12)))
    (at g j 4))
    ((use (j-not-in-q23) (j-in-q4))))

;;: If the k's entry in lp is 12,
;;: then the k's entry in 1 is 11.
(prove-lemma k-in-111 (rewrite)
  (implies (and (ws n 1 q)
    (member k (nset n))
    (rhoi n k 1 q lp qp)
    (at lp k 12))
    (at 1 k 11))
    ((enable rhoi at)))

;;: If k is not equal to j and the j's entry in q is 4,
;;: then the j's entry in qp is 4.
(prove-lemma lml-a3-i-eq-k-j-neq-k (rewrite)
  (implies (and (ws n 1 q)
    (member j (nset n))
    (member k (nset n))
    (rhoi n k 1 q lp qp)
    (not (equal j k))
    (lg n 1 q)
    (a2 n n 1)
    (at lp k 12)
    (union-at-n lp j '(5 6 7 8 9 10 11 12))
    (at g j 4))
    ((use (a3-j-in-15-12) (k-in-111))))

;;: If j is not equal to k, the k's entry in lp is 12,
;;: and the j's entry in lp is between 5 and 12,
;;: then the j's entry in qp is 4.
(prove-lemma lm-a3-i-eq-k-j-neq-k (rewrite)
  (implies (and (ws n 1 q)
    (member j (nset n))
    (member k (nset n))
    (rhoi n k 1 q lp qp)
    (not (equal j k))
    (lg n 1 q)
    (a2 n n 1)
    (a3-at-n1-n2 k j 1 g))
    (a3-at-n1-n2 k j lp qp))
    ((enable a3-at-n1-n2)
    (use (lml-a3-i-eq-k-j-neq-k))))

;;: If j is not equal to k then (a3-at-n1-n2 k j lp qp).
(prove-lemma a3-i-eq-k-j-neq-k (rewrite)
  (implies (and (ws n 1 q)
    (member j (nset n))
    (member k (nset n))
    (rhoi n k 1 q lp qp)
    (not (equal j k))
    (lg n 1 q)
    (a2 n n 1)
    (a3 n n 1 g))
    (a3-at-n1-n2 k j lp qp))
    ((use (a3-i-j-a3-at-n1-n2 (i k)))
    (use (lm-a3-i-eq-k-j-neq-k))))

;* i-j-neq-k

;;: If i,j are not equal to k, the i's entry in lp is 12
;;: and the j's entry in lp between 5 and 12
;;: then the j's entry in qp is 4.
(prove-lemma lm-a3-i-j-neq-k (rewrite)
  (implies (and (ws n 1 q)
    (member i (nset n))
    (member j (nset n))
    (member k (nset n))
    (rhoi n k 1 q lp qp)
    (not (equal i k))
    (not (equal j k))
    (a3-at-n1-n2 i j 1 g))
    (a3-at-n1-n2 i j lp qp))
    ((enable a3-at-n1-n2)))

;;: If i,j are not equal to k,
;;: then (a3-at-n1-n2 i j lp qp).
(prove-lemma a3-i-j-neq-k (rewrite)
  (implies (and (ws n 1 q)
    (member i (nset n))
    (member j (nset n))
    (member k (nset n))
    (rhoi n k 1 q lp qp)
    (a3 n n 1 g)
    (not (equal i k))
    (not (equal j k)))
    (a3-at-n1-n2 i j lp qp))
    (a3-at-n1-n2 i j lp qp))

```

```

((use (lm-a3-i-j-neq-k))
  (use (a3-i-j-a3-at-n1-n2))))

;* i-j-eq-k

(prove-lemma lm-a3-i-j-eq-k (rewrite)
  (implies (and (ws n 1 q)
    (member k (nset n))
    (rhoi n k 1 q lp qp)
    (lg n 1 q)
    (a2 n n 1)
    (a3-at-n1-n2 k k 1 q))
    (a3-at-n1-n2 k k lp qp))
    ((enable a3-at-n1-n2)
    (use (if4 (j k) (1 lp) (g gp)))
    (use (l12-then-un9-12))))

;;: (a3-at-n1-n2 k k lp qp) holds by lg.
(prove-lemma a3-i-j-eq-k (rewrite)
  (implies (and (ws n 1 q)
    (member k (nset n))
    (rhoi n k 1 q lp qp)
    (lg n 1 g)
    (a2 n n 1)
    (a3 n n 1 g))
    (a3-at-n1-n2 k k lp qp))
    ((use (lm-a3-i-j-eq-k))
    (use (a3-i-j-a3-at-n1-n2 (i k) (j k)))))

;;: (a3-at-n1-n2 k j lp qp) holds.
(prove-lemma a3-i-eq-k (rewrite)
  (implies (and (ws n 1 q)
    (member j (nset n))
    (member k (nset n))
    (rhoi n k 1 q lp qp)
    (lg n 1 g)
    (a2 n n 1)
    (a3 n n 1 g))
    (a3-at-n1-n2 k j lp qp))
    ((use (a3-i-eq-k-j-neq-k) (a3-i-j-eq-k))))

;;: If i is not equal to k,
;;: then (a3-at-n1-n2 i j lp qp) holds.
(prove-lemma a3-i-neq-k (rewrite)
  (implies (and (ws n 1 q)
    (member k (nset n))
    (member i (nset n))
    (member j (nset n))
    (rhoi n k 1 q lp qp)
    (lg n 1 g)
    (a0 n 1 k)
    (a2 n n 1)
    (a3 n n 1 q)
    (not (equal i k)))
    (a3-at-n1-n2 i j lp qp))
    ((use (a3-i-j-neq-k) (a3-i-neq-k-j-eq-k))))

;;: (a3-at-n1-n2 i j lp qp) holds.
(prove-lemma rho-preserves-a3 ()
  (implies (and (ws n 1 q)
    (member k (nset n))
    (member i (nset n))
    (member j (nset n))
    (rhoi n k 1 q lp qp)
    (lg n 1 g)
    (a0 n 1 k)
    (a2 n n 1)
    (a3 n n 1 g))
    (a3-at-n1-n2 i j lp qp))
    ((use (a3-i-neq-k) (a3-i-eq-k)))))

```

```

;* Well-formed states

(defn molws (n 1 g h)
  (and (numberp n)
    (listp 1)
    (listp g)
    (listp h)
    (equal (length 1) n)
    (equal (length g) n)
    (equal (length h) n)
    (all-union 1 n '(0 1 2 3 4 5 6 7 8 9 10 11 12))
    (all-union g n '(0 1 2 3 4))
    (all-union h n (nset (add1 n))))))

(disable molws)

;* Transitions

(defn mrhoi0 (n 1 1 q h lp qp hp)
  (and (at 1 i 0)
    (equal qp g)
    (equal lp (move 1 i 1))
    (equal hp h)))

(defn mrhoila (n 1 1 q h lp qp hp)
  (and (at 1 i 1)
    (equal qp g)
    (equal lp (move 1 i 2))
    (equal hp h)))

(defn mrhoilb (n 1 1 q h lp qp hp)
  (and (at 1 i 1)
    (equal qp g)
    (equal lp 1)
    (equal hp h)))

(defn mrhoi2 (n 1 1 q h lp qp hp)
  (and (at 1 i 2)
    (equal lp (move 1 i 3))
    (equal qp (move g i 1))
    (equal hp (move h i 1))))

(defn mrhoi3a (n 1 1 g h lp qp hp)
  (and (at 1 i 3)
    (equal qp g)
    (equal hp h)
    (at h i (add1 n))
    (equal lp (move 1 i 4))))

(defn mrhoi3b (n 1 1 q h lp qp hp)
  (and (at 1 i 3)
    (equal qp g)
    (equal lp 1)
    (lessp (nth h i) (add1 n))
    (equal hp (move h i (add1 (nth h i)))))
    (union-at-n q (nth h i) '(0 1 2))))

(defn mrhoi4 (n 1 1 q h lp qp hp)
  (and (at 1 i 4)
    (equal gp (move g i 3))
    (equal lp (move 1 i 5))
    (equal hp (move h i 1))))

(defn mrhoi5a (n 1 1 q h lp qp hp)
  (and (at 1 i 5)
    (equal qp g)
    (equal hp h)
    (at h i (add1 n))
    (equal lp (move 1 i 8))))

(defn mrhoi5b (n 1 1 g h lp qp hp)
  (and (at 1 i 5)
    (equal qp g)
    (equal hp h)
    (lessp (nth h i) (add1 n))
    (at q (nth h i) 1)
    (equal lp (move 1 i 6))))

(defn mrhoi5c (n 1 1 g h lp qp hp)
  (and (at 1 i 5)
    (equal qp g)
    (equal lp 1)
    (lessp (nth h i) (add1 n))
    (not (at g (nth h i) 1))
    (equal hp (move h i (add1 (nth h i))))))

(defn mrhoi6 (n 1 1 g h lp qp hp)
  (and (at 1 i 6)
    (equal gp (move q i 2))
    (equal lp (move 1 i 7))
    (equal hp (move h i 1))))

(defn mrhoi7a (n 1 1 g h lp qp hp)
  (and (at 1 i 7)
    (equal lp (move 1 i 8))
    (at q (nth h i) 4)
    (equal qp q)
    (equal hp h)))

(defn mrhoi7b (n 1 1 g h lp qp hp)
  (and (at 1 i 7)
    (not (at g (nth h i) 4))
    (equal lp 1)
    (equal qp g)
    (equal hp (move h i
      (add1 (remainder (sub1 (nth h i)) n))))))

(defn mrhoi8 (n 1 1 q h lp qp hp)
  (and (at 1 i 8)
    (equal gp (move q i 4))
    (equal lp (move 1 i 9))
    (equal hp (move h i 1)))

(defn mrhoi9a (n 1 1 g h lp qp hp)
  (and (at 1 i 9)
    (at h i i)
    (equal lp (move 1 i 10))
    (equal qp g)
    (equal hp h)))

(defn mrhoi9b (n 1 1 q h lp qp hp)
  (and (at 1 i 9)
    (lessp (nth h i) i)
    (union-at-n g (nth h i) '(0 1))
    (equal hp (move h i (add1 (nth h i)))))
    (equal qp q)
    (equal lp 1)))

(defn mrhoi10 (n 1 1 g h lp qp hp)
  (and (at 1 i 10)
    (equal lp (move 1 i 11))
    (equal qp g)
    (equal hp (move h i (add1 i)))))

(defn mrhoi11a (n 1 1 q h lp qp hp)
  (and (at 1 i 11)
    (at h i (add1 n))
    (equal lp (move 1 i 12))
    (equal qp g)
    (equal hp h)))

(defn mrhoi11b (n 1 1 q h lp qp hp)
  (and (at 1 i 11)
    (lessp (nth h i) (add1 n))
    (not (union-at-n g (nth h i) '(2 3)))
    (equal hp (move h i (add1 (nth h i)))))
    (equal qp g)
    (equal lp 1)))

(defn mrhoi12 (n 1 1 q h lp qp hp)
  (and (at 1 i 12)
    (equal hp h)
    (equal gp (move q i 0))
    (equal lp (move 1 i 0))))

(defn mrhoi (n 1 1 q h lp qp hp)
  (or (mrhoi0 n 1 1 q h lp qp hp)
    (mrhoila n 1 1 g h lp qp hp)
    (mrhoilb n 1 1 q h lp qp hp)
    (mrhoi2 n 1 1 q h lp qp hp)
    (mrhoi3a n 1 1 g h lp qp hp)
    (mrhoi3b n 1 1 q h lp qp hp)
    (mrhoi4 n 1 1 q h lp qp hp)
    (mrhoi5a n 1 1 q h lp qp hp)
    (mrhoi5b n 1 1 g h lp qp hp)
    (mrhoi5c n 1 1 g h lp qp hp)
    (mrhoi6 n 1 1 q h lp qp hp)
    (mrhoi7a n 1 1 q h lp qp hp)
    (mrhoi7b n 1 1 g h lp qp hp)
    (mrhoi8 n 1 1 q h lp qp hp)
    (mrhoi9a n 1 1 q h lp qp hp)
    (mrhoi9b n 1 1 q h lp qp hp)
    (mrhoi10 n 1 1 g h lp qp hp)
    (mrhoi11a n 1 1 g h lp qp hp)
    (mrhoi11b n 1 1 q h lp qp hp)
    (mrhoi12 n 1 1 q h lp qp hp)))

(disable mrhoi)

;* Invariants

;;;b0
(defn b0a (n 1 h i j)
  (implies (and (at 1 i 5)
    (lessp j (nth h i)))
    (not (at 1 j 4))))

(disable b0a)

(defn b0b (n 1 h i j)
  (implies (and (at 1 i 5)
    (lessp j (nth h i))
    (at 1 j 3))
    (not (lessp i (nth h j)))))

(disable b0b)

;;;b1
(defn bla (1 i j)
  (implies (union-at-n 1 i '(8 9 10 11 12))

```

```

(not (at 1 j 4)))
(disable bla)

(defn hint-8-12-3-4-at-n (n 1 q h j)
  (and (intersect-8-12-3-4-at-n n 1 q)
        (not (lessp n (nth h j)))))

(disable hint-8-12-3-4-at-n)

(defn exist-hint-8-12-3-4 (n 1 q h j)
  (if (zerop n) F
      (if (hint-8-12-3-4-at-n n 1 q h j) n
          (exist-hint-8-12-3-4 (sub1 n) 1 q h j))))

(disable exist-hint-8-12-3-4)

(defn blb (n 1 q h i j)
  (implies (and (union-at-n 1 i '(8 9 10 11 12))
                (at 1 j 3))
            (exist-hint-a-12-3-4 n 1 q h j)))
(disable blb)

(defn blc (n 1 q h i)
  (implies (and (union-at-n 1 i '(8 9 10 11 12))
                (not (union-at-n q i '(3 4))))
            (and (member (nth h i) (nset n))
                  (at q (nth h i) 4))))
(disable blc)

(defn bld (n 1 h i)
  (implies (at 1 i 7)
            (member (nth h i) (nset n))))
(disable bld)

::: b2
(defn b2a (1 i j)
  (implies (and (lessp j i)
                (union-at-n 1 i '(10 11 12)))
            (not (union-at-n 1 j '(5 6 7 8 9 10 11 12)))))
(disable b2a)

(defn b2b (1 h i j)
  (implies (and (lessp j i)
                (at 1 i 9)
                (lessp j (nth h i)))
            (not (union-at-n 1 j '(5 6 7 8 9 10 11 12)))))
(disable b2b)

::: b3
(defn b3a (1 g i j)
  (implies (and (at 1 i 12)
                (union-at-n 1 j '(5 6 7 8 9 10 11 12)))
            (at g j 4)))
(disable b3a)

(defn b3b (1 q h i j)
  (implies (and (at 1 i 11)
                (lessp j (nth h i))
                (union-at-n 1 j '(5 6 7 8 9 10 11 12)))
            (at q j 4)))
(disable b3b)

```

```
(prove-lemma hint-member (rewrite)
  (implies (exist-hint-8-12-3-4 n 1 q h j)
    (member (exist-hint-8-12-3-4 n 1 g h j) (nset n)))
  ((enable nset exist-hint-8-12-3-4 hint-8-12-3-4-at-n)))
```

```
(prove-lemma n-not-less-j (rewrite)
  (implies (lessp n j)
    (not (member j (nset n))))
  ((enable nset)))
```

;;molws implies that n is a number.

```
(prove-lemma molws-num-n (rewrite)
  (implies (molws n 1 q h)
    (numberp n))
  ((enable molws)))
```

;;molws implies that l is a list.

```
(prove-lemma molws-list-1 (rewrite)
  (implies (molws n 1 q h)
    (listp 1))
  ((enable molws)))
```

;;molws implies that g is a list.

```
(prove-lemma molws-list-g(rewrite)
  (implies (molws n 1 q h)
    (listp g))
  ((enable molws)))
```

;;molws implies that h is a list.

```
(prove-lemma molws-list-h(rewrite)
  (implies (molws n 1 q h)
    (listp h))
  ((enable molws)))
```

;;molws implies that length of l is n.

```
(prove-lemma molws-ln-1(rewrite)
  (implies (molws n 1 q h)
    (equal (length l) n))
  ((enable molws)))
```

;;molws implies that length of q is n.

```
(prove-lemma molws-ln-q(rewrite)
  (implies (molws n 1 g h)
    (equal (length q) n))
  ((enable molws)))
```

;;molws implies that length of h is n.

```
(prove-lemma molws-ln-h(rewrite)
  (implies (molws n 1 q h)
    (equal (length h) n))
  ((enable molws)))
```

;;molws and mrho imply that lp is a list.

```
(prove-lemma molws-ln-lp (rewrite)
  (implies (and (molws n 1 g h)
    (member k (nset n))
    (mrhoi n k 1 q h lp qp hp))
    (listp lp))
  ((enable mrhoi)))
```

;;molws and mrho imply that gp is a list.

```
(prove-lemma molws-ln-gp (rewrite)
  (implies (and (molws n 1 g h)
    (member k (nset n))
    (mrhoi n k 1 q h lp qp hp))
    (listp gp))
  ((enable mrhoi)))
```

;;molws and mrho imply that hp is a list.

```
(prove-lemma molws-ln-hp (rewrite)
  (implies (and (molws n 1 q h)
    (member k (nset n))
    (mrhoi n k 1 g h lp qp hp))
    (listp hp))
  ((enable mrhoi)))
```

;;Another version of nset-number.

;;This is available in the theorem

;;where molws is disabled.

```
(prove-lemma molws-num-k (rewrite)
  (implies (and (molws n 1 g h)
    (member k (nset n)))
    (numberp k))
  ((enable nset)))
```

```
(prove-lemma molws-union-h (rewrite)
  (implies (molws n 1 g h)
    (all-union h n (nset (add1 n))))
  ((enable molws)))
```

```
(prove-lemma lm-nth-numberp (rewrite)
  (implies (and (numberp i)
    (all-union h n (nset i))
    (member k (nset n)))
    (numberp (nth h k)))
  ((enable nset all-union union-at-n at)))
```

```
(prove-lemma nth-numberp (rewrite)
```

```
(implies (and (molws n 1 q h)
  (member k (nset n)))
  (numberp (nth h k)))
  ((use (lm-nth-numberp (i (add1 n))))))
```

;;molws implies that n is nonzero.

```
(prove-lemma molws-n-not-0 (rewrite)
  (implies (molws n 1 g h)
    (not (equal n 0)))
  ((enable molws)))
```

;;Auxiliary lemma.

```
(prove-lemma lm-1-mrholemma (rewrite)
  (implies (and (listp 1)
    (member j (nset (length 1)))
    (member k (nset (length 1)))
    (mrhoi n k 1 q h lp qp hp)
    (not (equal k j)))
    (equal (nth 1 j) (nth lp j)))
  ((enable mrhoi)))
```

(disable lm-1-mrholemma)

;;Mrholemma for list 1.

```
(prove-lemma l-mrholemma (rewrite)
  (implies (and (molws n 1 q h)
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp qp hp)
    (not (equal k j)))
    (equal (nth 1 j) (nth lp j)))
  ((enable lm-1-mrholemma) (use (lm-1-mrholemma))))
```

;;Auxiliary lemma.

```
(prove-lemma lm-q-mrholemma (rewrite)
  (implies (and (listp g)
    (member j (nset (length g)))
    (member k (nset (length g)))
    (mrhoi n k 1 q h lp qp hp)
    (not (equal k j)))
    (equal (nth g j) (nth qp j)))
  ((enable mrhoi)))
```

(disable lm-q-mrholemma)

;;Mrholemma for list q.

```
(prove-lemma q-mrholemma (rewrite)
  (implies (and (molws n 1 q h)
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp qp hp)
    (not (equal k j)))
    (equal (nth q j) (nth qp j)))
  ((enable lm-q-mrholemma) (use (lm-q-mrholemma))))
```

;;Auxiliary lemma.

```
(prove-lemma lm-h-mrholemma (rewrite)
  (implies (and (listp h)
    (member j (nset (length h)))
    (member k (nset (length h)))
    (mrhoi n k 1 q h lp qp hp)
    (not (equal k j)))
    (equal (nth h j) (nth hp j)))
  ((enable mrhoi)))
```

(disable lm-h-mrholemma)

;;Mrholemma for list g.

```
(prove-lemma h-mrholemma (rewrite)
  (implies (and (molws n 1 q h)
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 q h lp qp hp)
    (not (equal k j)))
    (equal (nth h j) (nth hp j)))
  ((enable lm-h-mrholemma) (use (lm-h-mrholemma))))
```

;; lp-qp-same-1-g

;;Another version of Rholemma for 1.

;;It applies to (union-at-n 1 j m) in stead of

;;(nth 1 j).

```
(prove-lemma m-lp-same-1 (rewrite)
  (implies (and (molws n 1 g h)
    (listp m)
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 q h lp qp hp)
    (not (equal j k))
    (union-at-n 1 j m))
    (union-at-n lp j m))
  ((enable union-at-n at) (use (l-mrholemma))))
```

;;Contrast to the one above,

;;the order of 1 and lp is reversed.

```
(prove-lemma m-1-same-lp (rewrite)
  (implies (and (molws n 1 g h)
    (listp m)
```

```

(member j (nset n))
(member k (nset n))
(mrhoi n k l q h lp qp hp)
(not (equal j k))
(union-at-n lp j m))
(union-at-n l j m))
((enable union-at-n at) (use (l-mrholemma))))

(prove-lemma m-lp-same-l-not (rewrite)
  (implies (and (molws n l g h)
    (listp m)
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k l q h lp qp hp)
    (not (equal j k))
    (not (union-at-n lp j m)))
    (not (union-at-n l j m)))
    ((use (m-lp-same-l))))

;;;Another version of Rholemma for q.
(prove-lemma m-qp-same-g (rewrite)
  (implies (and (molws n l q h)
    (listp m)
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k l q h lp qp hp)
    (not (equal j k))
    (union-at-n q j m))
    (union-at-n qp j m))
    ((enable union-at-n at) (use (q-mrholemma))))

;;;Contrast to the one above,
;;;the order of g and qp is reversed.
(prove-lemma m-q-same-gp (rewrite)
  (implies (and (molws n l g h)
    (listp m)
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k l g h lp qp hp)
    (not (equal j k))
    (union-at-n qp j m))
    (union-at-n q j m))
    ((enable union-at-n at) (use (q-mrholemma))))

(prove-lemma m-qp-same-g-not (rewrite)
  (implies (and (molws n l q h)
    (listp m)
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k l q h lp qp hp)
    (not (equal j k))
    (not (union-at-n qp j m)))
    (not (union-at-n q j m))
    ((use (m-qp-same-g))))

;;;Another version of Rholemma for h.
(prove-lemma m-hp-same-h (rewrite)
  (implies (and (molws n l g h)
    (listp m)
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k l g h lp qp hp)
    (not (equal j k))
    (union-at-n h j m))
    (union-at-n hp j m))
    ((enable union-at-n at) (use (h-mrholemma))))

;;;Contrast to the one above,
;;;the order of q and qp is reversed.
(prove-lemma m-h-same-hp (rewrite)
  (implies (and (molws n l g h)
    (listp m)
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k l q h lp qp hp)
    (not (equal j k))
    (union-at-n hp j m))
    (union-at-n h j m))
    ((enable union-at-n at) (use (h-mrholemma))))

;;;It applies to (at l j m) in stead of
;;;(nth l j).
(prove-lemma m-l-same-lp-at (rewrite)
  (implies (and (molws n l q h)
    (member j (nset n))
    (member k (nset n))
    (numberp m)
    (mrhoi n k l q h lp qp hp)
    (not (equal k j))
    (at lp j m))
    (at l j m))
    ((enable at) (use (l-mrholemma))))

(prove-lemma m-qp-same-q-at (rewrite)
  (implies (and (molws n l q h)
    (member j (nset n))
    (member k (nset n))
    (numberp m)
    (mrhoi n k l g h lp qp hp)
    (not (equal k j))
    (at lp j m))
    (at l j m))
    ((enable at) (use (q-mrholemma))))

(mrhoi n k l g h lp qp hp)
(not (equal k j))
(at g j m))
(at g j m))
((enable at) (use (q-mrholemma))))

(prove-lemma m-l-same-lp-at-not (rewrite)
  (implies (and (molws n l g h)
    (numberp m)
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k l q h lp qp hp)
    (not (equal j k))
    (not (at l j m)))
    (not (at lp j m))
    ((use (m-l-same-lp-at))))

```

```

:::mrhoi0
(prove-lemma n-neq-k-mrhoi0 (rewrite)
  (implies (and (listp 1)
    (listp g)
    (numberp n)
    (member k (nset (length 1)))
    (not (equal k n))
    (at 1 k 0)
    (lg-at-n n 1 g))
    (lg-at-n n (move 1 k 1) q))
  ((enable at lg-at-n lg-1-at-n lg-2-at-n lg-3-at-n)))

(disable n-neq-k-mrhoi0)

(prove-lemma n-eq-k-mrhoi0 (rewrite)
  (implies (and (listp 1)
    (listp g)
    (member k (nset (length 1)))
    (at 1 k 0)
    (lg-at-n k 1 g))
    (lg-at-n n (move 1 k 1) q))
  ((enable at lg-at-n lg-1-at-n lg-2-at-n lg-3-at-n)))

(disable n-eq-k-mrhoi0)

(prove-lemma lg-at-mrhoi0 (rewrite)
  (implies (and (listp 1)
    (listp g)
    (numberp n)
    (member k (nset (length 1)))
    (at 1 k 0)
    (lg-at-n n 1 q))
    (lg-at-n n (move 1 k 1) g))
  ((enable n-neq-k-mrhoi0 n-eq-k-mrhoi0)
  (use (n-neq-k-mrhoi0))
  (use (n-eq-k-mrhoi0))))

(disable lg-at-mrhoi0)

(prove-lemma lg-mrhoi0 (rewrite)
  (implies (and (listp 1)
    (listp g)
    (member k (nset (length 1)))
    (numberp n)
    (at 1 k 0)
    (lg n 1 g))
    (lg n (move 1 k 1) q))
  ((enable lg-at-mrhoi0 lg at)))

(disable lg-mrhoi0)

(prove-lemma mrhoi0-preserves-lg (rewrite)
  (implies (and (molws n 1 g h)
    (member k (nset n))
    (mrhoi0 n k 1 q h lp qp hp)
    (lg n 1 g))
    (lg n lp qp))
  ((enable lg-mrhoi0)))

:::mrhoila
(prove-lemma n-neq-k-mrhoila (rewrite)
  (implies (and (listp 1)
    (listp g)
    (numberp n)
    (member k (nset (length 1)))
    (not (equal k n))
    (at 1 k 1)
    (lg-at-n n 1 q))
    (lg-at-n n (move 1 k 2) q))
  ((enable at lg-at-n lg-1-at-n lg-2-at-n lg-3-at-n)))

(disable n-neq-k-mrhoila)

(prove-lemma n-eq-k-mrhoila (rewrite)
  (implies (and (listp 1)
    (listp g)
    (member k (nset (length 1)))
    (at 1 k 1)
    (lg-at-n k 1 g))
    (lg-at-n n (move 1 k 2) g))
  ((enable at lg-at-n lg-1-at-n lg-2-at-n lg-3-at-n)))

(disable n-eq-k-mrhoila)

(prove-lemma lg-at-mrhoila (rewrite)
  (implies (and (listp 1)
    (listp g)
    (numberp n)
    (member k (nset (length 1)))
    (at 1 k 1)
    (lg-at-n n 1 g))
    (lg-at-n n (move 1 k 2) g))
  ((enable n-neq-k-mrhoila n-eq-k-mrhoila)
  (use (n-neq-k-mrhoila))
  (use (n-eq-k-mrhoila))))

(disable lg-at-mrhoila)

(prove-lemma lq-mrhoila (rewrite)
  (implies (and (listp 1)
    (listp g)
    (numberp n)
    (member k (nset (length 1)))
    (at 1 k 1)
    (lg n 1 g))
    (lg n (move 1 k 2) g))
  ((enable lq-at-mrhoila lq at)))

(disable lq-mrhoila)

(prove-lemma lq-mrhoila-preserves-lq (rewrite)
  (implies (and (molws n 1 q h)
    (member k (nset n))
    (mrhoila n k 1 g h lp qp hp)
    (lg n 1 g))
    (lg n lp qp))
  ((enable lq-mrhoila)))

:::mrhoilb
(prove-lemma mrhoilb-preserves-lg (rewrite)
  (implies (and (molws n 1 q h)
    (member k (nset n))
    (mrhoilb n k 1 g h lp qp hp)
    (lg n 1 g))
    (lg n lp qp))
  ((enable mrhoilb)))

:::mrhoi2
(prove-lemma n-neq-k-mrhoi2 (rewrite)
  (implies (and (listp 1)
    (listp g)
    (numberp n)
    (member k (nset (length 1)))
    (not (equal k n))
    (at 1 k 2)
    (lg-at-n n 1 g))
    (lg-at-n n (move 1 k 3) (move g k 1)))
  ((enable at lg-at-n lg-1-at-n lg-2-at-n lg-3-at-n)))

(disable n-neq-k-mrhoi2)

(prove-lemma n-eq-k-mrhoi2 (rewrite)
  (implies (and (listp 1)
    (listp g)
    (member k (nset (length 1)))
    (at 1 k 2)
    (lg-at-n k 1 q))
    (lg-at-n n (move 1 k 3) (move q k 1)))
  ((enable at lg-at-n lg-1-at-n lg-2-at-n lg-3-at-n)))

(disable n-eq-k-mrhoi2)

(prove-lemma lg-at-mrhoi2 (rewrite)
  (implies (and (listp 1)
    (listp g)
    (numberp n)
    (member k (nset (length 1)))
    (at 1 k 2)
    (lg-at-n n 1 g))
    (lg-at-n n (move 1 k 3) (move q k 1)))
  ((enable n-neq-k-mrhoi2 n-eq-k-mrhoi2)
  (use (n-neq-k-mrhoi2))
  (use (n-eq-k-mrhoi2))))

(disable lg-at-mrhoi2)

(prove-lemma lq-mrhoi2 (rewrite)
  (implies (and (listp 1)
    (listp g)
    (member k (nset (length 1)))
    (numberp n)
    (at 1 k 2)
    (lg n 1 g))
    (lg n (move 1 k 3) (move q k 1)))
  ((enable lq-at-mrhoi2 lq at)))

(disable lq-mrhoi2)

(prove-lemma mrhoi2-preserves-lg (rewrite)
  (implies (and (molws n 1 g h)
    (member k (nset n))
    (mrhoi2 n k 1 q h lp qp hp)
    (lg n 1 g))
    (lg n lp qp))
  ((enable lq-mrhoi2)))

:::mrhoi3a
(prove-lemma n-neq-k-mrhoi3a (rewrite)
  (implies (and (listp 1)
    (listp g)
    (numberp n)
    (member k (nset (length 1)))
    (not (equal k n))
    (at 1 k 3)
    (lg-at-n n 1 q))
    (lg-at-n n (move 1 k 4) g))
  ((enable lq-at-mrhoi2)))

```

```

((enable at lg-at-n lg-1-at-n lg-2-at-n lg-3-at-n)))

(disable n-neq-k-mrhoi3a)

(prove-lemma n-eq-k-mrhoi3a (rewrite)
  (implies (and (listp 1)
    (listp g)
    (member k (nset (length 1)))
    (at 1 k 3)
    (lg-at-n k 1 g))
    (lg-at-n k (move 1 k 4) g))
    ((enable at lg-at-n lg-1-at-n lg-2-at-n lg-3-at-n))))

(disable n-eq-k-mrhoi3a)

(prove-lemma lg-at-mrhoi3a (rewrite)
  (implies (and (listp 1)
    (listp g)
    (numberp n)
    (member k (nset (length 1)))
    (at 1 k 3)
    (lg-at-n n 1 g))
    (lg-at-n n (move 1 k 4) g))
    ((enable n-neq-k-mrhoi3a n-eq-k-mrhoi3a)
    (use (n-neq-k-mrhoi3a))
    (use (n-eq-k-mrhoi3a)))))

(disable lg-at-mrhoi3a)

(prove-lemma lg-mrhoi3a (rewrite)
  (implies (and (listp 1)
    (listp g)
    (member k (nset (length 1)))
    (numberp n)
    (at 1 k 3)
    (lg n 1 q))
    (lg n (move 1 k 4) g))
    ((enable lg-at-mrhoi3a lg at))))

(disable lg-mrhoi3a)

(prove-lemma mrhoi3a-preserves-lg (rewrite)
  (implies (and (molws n 1 q h)
    (member k (nset n))
    (mrhoi3a n k 1 q h lp qp hp)
    (lg n 1 g))
    (lg n lp qp))
    ((enable lg-mrhoi3a))))

;;mrhoi3b
(prove-lemma mrhoi3b-preserves-lg (rewrite)
  (implies (and (molws n 1 g h)
    (member k (nset n))
    (mrhoi3b n k 1 g h lp qp hp)
    (lg n 1 g))
    (lg n lp qp))
    ((enable mrhoi3b))))

;;mrhoi4
(prove-lemma n-neq-k-mrhoi4 (rewrite)
  (implies (and (listp 1)
    (listp g)
    (numberp n)
    (member k (nset (length 1)))
    (not (equal k n))
    (at 1 k 4)
    (lg-at-n n 1 q))
    (lg-at-n n (move 1 k 5) (move q k 3)))
    ((enable at lg-at-n lg-1-at-n lg-2-at-n lg-3-at-n)))

(disable n-neq-k-mrhoi4)

(prove-lemma n-eq-k-mrhoi4 (rewrite)
  (implies (and (listp 1)
    (listp g)
    (member k (nset (length 1)))
    (at 1 k 4)
    (lg-at-n k 1 q))
    (lg-at-n n (move 1 k 5) (move q k 3)))
    ((enable at lg-at-n lg-1-at-n lg-2-at-n lg-3-at-n)))

(disable n-eq-k-mrhoi4)

(prove-lemma lg-at-mrhoi4 (rewrite)
  (implies (and (listp 1)
    (listp g)
    (numberp n)
    (member k (nset (length 1)))
    (at 1 k 4)
    (lg-at-n n 1 q))
    (lg-at-n n (move 1 k 5) (move q k 3)))
    ((enable n-neq-k-mrhoi4 n-eq-k-mrhoi4)
    (use (n-neq-k-mrhoi4))
    (use (n-eq-k-mrhoi4)))))

(disable lg-at-mrhoi4)

(prove-lemma lg-mrhoi4 (rewrite)

```

```

  (implies (and (listp 1)
    (listp g)
    (member k (nset (length 1)))
    (numberp n)
    (at 1 k 4)
    (lg n 1 g))
    (lg n (move 1 k 5) (move g k 3)))
    ((enable lg-at-mrhoi4 lg at))))

(disable lg-mrhoi4)

(prove-lemma mrhoi4-preserves-lg (rewrite)
  (implies (and (molws n 1 g h)
    (member k (nset n))
    (mrhoi4 n k 1 g h lp qp hp)
    (lg n 1 g))
    (lg n lp qp))
    ((enable lg-mrhoi4))))

;;mrhoi5a
(prove-lemma n-neq-k-mrhoi5a (rewrite)
  (implies (and (listp 1)
    (listp g)
    (numberp n)
    (member k (nset (length 1)))
    (not (equal k n))
    (at 1 k 5)
    (lg-at-n n 1 q))
    (lg-at-n n (move 1 k 8) g))
    ((enable at lg-at-n lg-1-at-n lg-2-at-n lg-3-at-n)))

(disable n-neq-k-mrhoi5a)

(prove-lemma n-eq-k-mrhoi5a (rewrite)
  (implies (and (listp 1)
    (listp g)
    (member k (nset (length 1)))
    (at 1 k 5)
    (lg-at-n k 1 g))
    (lg-at-n k (move 1 k 8) g))
    ((enable at lg-at-n lg-1-at-n lg-2-at-n lg-3-at-n)))

(disable n-eq-k-mrhoi5a)

(prove-lemma lg-at-mrhoi5a (rewrite)
  (implies (and (listp 1)
    (listp g)
    (numberp n)
    (member k (nset (length 1)))
    (at 1 k 5)
    (lg-at-n n 1 q))
    (lg-at-n n (move 1 k 8) g))
    ((enable n-neq-k-mrhoi5a n-eq-k-mrhoi5a)
    (use (n-neq-k-mrhoi5a))
    (use (n-eq-k-mrhoi5a)))))

(disable lg-at-mrhoi5a)

(prove-lemma lg-mrhoi5a (rewrite)
  (implies (and (listp 1)
    (listp g)
    (member k (nset (length 1)))
    (numberp n)
    (at 1 k 5)
    (lg n 1 g))
    (lg n (move 1 k 8) q))
    ((enable lg-at-mrhoi5a lg at))))

(disable lg-mrhoi5a)

(prove-lemma mrhoi5a-preserves-lg (rewrite)
  (implies (and (molws n 1 q h)
    (member k (nset n))
    (mrhoi5a n k 1 g h lp qp hp)
    (lg n 1 g))
    (lg n lp qp))
    ((enable lg-mrhoi5a))))

;;mrhoi5b
(prove-lemma n-neq-k-mrhoi5b (rewrite)
  (implies (and (listp 1)
    (listp g)
    (numberp n)
    (member k (nset (length 1)))
    (not (equal k n))
    (at 1 k 5)
    (lg-at-n n 1 g))
    (lg-at-n n (move 1 k 6) g))
    ((enable at lg-at-n lg-1-at-n lg-2-at-n lg-3-at-n)))

(disable n-neq-k-mrhoi5b)

(prove-lemma n-eq-k-mrhoi5b (rewrite)
  (implies (and (listp 1)
    (listp g)
    (member k (nset (length 1)))
    (at 1 k 5)
    (lg-at-n k 1 g))

```

```

(lg-at-n k (move 1 k 6) g))
((enable at lg-at-n lg-1-at-n lg-2-at-n lg-3-at-n)))

(disable n-eq-k-mrhoi5b)

(prove-lemma lg-at-mrhoi5b (rewrite)
  (implies (and (listp 1)
    (listp g)
    (numberp n)
    (member k (nset (length 1)))
    (at 1 k 5)
    (lg-at-n n 1 g))
    (lg-at-n n (move 1 k 6) g))
  ((enable n-neq-k-mrhoi5b n-eq-k-mrhoi5b)
  (use (n-neq-k-mrhoi5b))
  (use (n-eq-k-mrhoi5b))))

(disable lg-at-mrhoi5b)

(prove-lemma lg-mrhoi5b (rewrite)
  (implies (and (listp 1)
    (listp g)
    (member k (nset (length 1)))
    (numberp n)
    (at 1 k 5)
    (lg n 1 g))
    (lg n (move 1 k 6) g))
  ((enable lg-at-mrhoi5b lg at)))

(disable lg-mrhoi5b)

(prove-lemma mrhoi5b-preserves-lg (rewrite)
  (implies (and (molws n 1 g h)
    (member k (nset n))
    (mrhoi5b n k 1 g h lp gp hp)
    (lg n 1 g))
    (lg n lp gp))
  ((enable lg-mrhoi5b)))

;;;mrhoi5c
(prove-lemma mrhoi5c-preserves-lg (rewrite)
  (implies (and (molws n 1 g h)
    (member k (nset n))
    (mrhoi5c n k 1 g h lp gp hp)
    (lg n 1 g))
    (lg n lp gp))
  ((enable mrhoi5c)))

;;;mrhoi6
(prove-lemma n-neq-k-mrhoi6 (rewrite)
  (implies (and (listp 1)
    (listp g)
    (numberp n)
    (member k (nset (length 1)))
    (not (equal k n))
    (at 1 k 6)
    (lg-at-n n 1 g))
    (lg-at-n n (move 1 k 7) (move g k 2)))
  ((enable at lg-at-n lg-1-at-n lg-2-at-n lg-3-at-n)))

(disable n-neq-k-mrhoi6)

(prove-lemma n-eq-k-mrhoi6 (rewrite)
  (implies (and (listp 1)
    (listp g)
    (member k (nset (length 1)))
    (at 1 k 6)
    (lg-at-n k 1 g))
    (lg-at-n n (move 1 k 7) (move g k 2)))
  ((enable at lg-at-n lg-1-at-n lg-2-at-n lg-3-at-n)))

(disable n-eq-k-mrhoi6)

(prove-lemma lg-at-mrhoi6 (rewrite)
  (implies (and (listp 1)
    (listp g)
    (numberp n)
    (member k (nset (length 1)))
    (at 1 k 6)
    (lg-at-n n 1 g))
    (lg-at-n n (move 1 k 7) (move g k 2)))
  ((enable n-neq-k-mrhoi6 n-eq-k-mrhoi6)
  (use (n-neq-k-mrhoi6))
  (use (n-eq-k-mrhoi6))))

(disable lg-at-mrhoi6)

(prove-lemma lg-mrhoi6 (rewrite)
  (implies (and (listp 1)
    (listp g)
    (member k (nset (length 1)))
    (numberp n)
    (at 1 k 6)
    (lg n 1 g))
    (lg n (move 1 k 7) (move g k 2)))
  ((enable lg-at-mrhoi6 lg at)))

(disable lg-mrhoi6)

```

```

(prove-lemma mrhoi6-preserves-lg (rewrite)
  (implies (and (molws n 1 g h)
    (member k (nset n))
    (mrhoi6 n k 1 g h lp gp hp)
    (lg n 1 g))
    (lg n lp gp))
  ((enable lg-mrhoi6)))

;;;mrhoi7a
(prove-lemma n-neq-k-mrhoi7a (rewrite)
  (implies (and (listp 1)
    (listp g)
    (numberp n)
    (member k (nset (length 1)))
    (not (equal k n))
    (at 1 k 7)
    (lg-at-n n 1 g))
    (lg-at-n n (move 1 k 8) g))
  ((enable at lg-at-n lg-1-at-n lg-2-at-n lg-3-at-n)))

(disable n-neq-k-mrhoi7a)

(prove-lemma n-eq-k-mrhoi7a (rewrite)
  (implies (and (listp 1)
    (listp g)
    (member k (nset (length 1)))
    (at 1 k 7)
    (lg-at-n k 1 g))
    (lg-at-n k (move 1 k 8) g))
  ((enable at lg-at-n lg-1-at-n lg-2-at-n lg-3-at-n)))

(disable n-eq-k-mrhoi7a)

(prove-lemma lg-at-mrhoi7a (rewrite)
  (implies (and (listp 1)
    (listp g)
    (numberp n)
    (member k (nset (length 1)))
    (at 1 k 7)
    (lg-at-n n 1 g))
    (lg-at-n n (move 1 k 8) g))
  ((enable n-neq-k-mrhoi7a n-eq-k-mrhoi7a)
  (use (n-neq-k-mrhoi7a))
  (use (n-eq-k-mrhoi7a))))

(disable lg-at-mrhoi7a)

(prove-lemma lg-mrhoi7a (rewrite)
  (implies (and (listp 1)
    (listp g)
    (member k (nset (length 1)))
    (numberp n)
    (at 1 k 7)
    (lg n 1 g))
    (lg n (move 1 k 8) g))
  ((enable lg-at-mrhoi7a lg at)))

(disable lg-mrhoi7a)

(prove-lemma mrhoi7a-preserves-lg (rewrite)
  (implies (and (molws n 1 g h)
    (member k (nset n))
    (mrhoi7a n k 1 g h lp gp hp)
    (lg n 1 g))
    (lg n lp gp))
  ((enable lg-mrhoi7a)))

;;;mrhoi7b
(prove-lemma mrhoi7b-preserves-lg (rewrite)
  (implies (and (molws n 1 g h)
    (member k (nset n))
    (mrhoi7b n k 1 g h lp gp hp)
    (lg n 1 g))
    (lg n lp gp))
  ((enable mrhoi7b)))

;;;mrhoi8
(prove-lemma n-neq-k-mrhoi8 (rewrite)
  (implies (and (listp 1)
    (listp g)
    (numberp n)
    (member k (nset (length 1)))
    (not (equal k n))
    (at 1 k 8)
    (lg-at-n n 1 g))
    (lg-at-n n (move 1 k 9) (move g k 4)))
  ((enable at lg-at-n lg-1-at-n lg-2-at-n lg-3-at-n)))

(disable n-neq-k-mrhoi8)

(prove-lemma n-eq-k-mrhoi8 (rewrite)
  (implies (and (listp 1)
    (listp g)
    (member k (nset (length 1)))
    (at 1 k 8)
    (lg-at-n k 1 g))
    (lg-at-n n (move 1 k 9) (move g k 4)))

```

```

((enable at lg-at-n lg-1-at-n lg-2-at-n lg-3-at-n)))

(disable n-eq-k-mrhoi8)

(prove-lemma lg-at-mrhoi8 (rewrite)
  (implies (and (listp 1)
    (listp g)
    (numberp n)
    (member k (nset (length 1)))
    (at 1 k 8)
    (lg-at-n n 1 g))
    (lg-at-n n (move 1 k 9) (move g k 4)))
  ((enable n-neq-k-mrhoi8 n-eq-k-mrhoi8)
  (use (n-neq-k-mrhoi8))
  (use (n-eq-k-mrhoi8))))

(disable lg-at-mrhoi8)

(prove-lemma lg-mrhoi8 (rewrite)
  (implies (and (listp 1)
    (listp g)
    (member k (nset (length 1)))
    (numberp n)
    (at 1 k 8)
    (lg n 1 g))
    (lg n (move 1 k 9) (move g k 4)))
  ((enable lg-at-mrhoi8 lg at)))

(disable lg-mrhoi8)

(prove-lemma mrhoi8-preserves-lg (rewrite)
  (implies (and (molws n 1 g h)
    (member k (nset n))
    (mrhoi8 n k 1 g h lp gp hp)
    (lg n 1 g))
    (lg n lp gp))
  ((enable lg-mrhoi8)))

:::mrhoi9a
(prove-lemma n-neq-k-mrhoi9a (rewrite)
  (implies (and (listp 1)
    (listp g)
    (numberp n)
    (member k (nset (length 1)))
    (not (equal k n))
    (at 1 k 9)
    (lg-at-n n 1 g))
    (lg-at-n n (move 1 k 10) g))
  ((enable at lg-at-n lg-1-at-n lg-2-at-n lg-3-at-n)))

(disable n-neq-k-mrhoi9a)

(prove-lemma n-eq-k-mrhoi9a (rewrite)
  (implies (and (listp 1)
    (listp g)
    (member k (nset (length 1)))
    (at 1 k 9)
    (lg-at-n k 1 g))
    (lg-at-n k (move 1 k 10) g))
  ((enable at lg-at-n lg-1-at-n lg-2-at-n lg-3-at-n)))

(disable n-eq-k-mrhoi9a)

(prove-lemma lg-at-mrhoi9a (rewrite)
  (implies (and (listp 1)
    (listp g)
    (numberp n)
    (member k (nset (length 1)))
    (at 1 k 9)
    (lg-at-n n 1 g))
    (lg-at-n n (move 1 k 10) g))
  ((enable n-neq-k-mrhoi9a n-eq-k-mrhoi9a)
  (use (n-neq-k-mrhoi9a))
  (use (n-eq-k-mrhoi9a))))

(disable lg-at-mrhoi9a)

(prove-lemma lg-mrhoi9a (rewrite)
  (implies (and (listp 1)
    (listp g)
    (member k (nset (length 1)))
    (numberp n)
    (at 1 k 9)
    (lg n 1 g))
    (lg n (move 1 k 10) g))
  ((enable lg-at-mrhoi9a lg at)))

(disable lg-mrhoi9a)

(prove-lemma mrhoi9a-preserves-lg (rewrite)
  (implies (and (molws n 1 g h)
    (member k (nset n))
    (mrhoi9a n k 1 g h lp gp hp)
    (lg n 1 g))
    (lg n lp gp))
  ((enable lg-mrhoi9a)))

:::mrhoi9b

```

```

(prove-lemma mrhoi9b-preserves-lg (rewrite)
  (implies (and (molws n 1 g h)
    (member k (nset n))
    (mrhoi9b n k 1 g h lp gp hp)
    (lg n 1 g))
    (lg n lp gp))
  ((enable mrhoi9b)))

:::mrhoi10
(prove-lemma n-neq-k-mrhoi10 (rewrite)
  (implies (and (listp 1)
    (listp g)
    (numberp n)
    (member k (nset (length 1)))
    (not (equal k n))
    (at 1 k 10)
    (lg-at-n n 1 g))
    (lg-at-n n (move 1 k 11) g))
  ((enable at lg-at-n lg-1-at-n lg-2-at-n lg-3-at-n)))

(disable n-neq-k-mrhoi10)

(prove-lemma n-eq-k-mrhoi10 (rewrite)
  (implies (and (listp 1)
    (listp g)
    (member k (nset (length 1)))
    (at 1 k 10)
    (lg-at-n k 1 g))
    (lg-at-n k (move 1 k 11) g))
  ((enable at lg-at-n lg-1-at-n lg-2-at-n lg-3-at-n)))

(disable n-eq-k-mrhoi10)

(prove-lemma lg-at-mrhoi10 (rewrite)
  (implies (and (listp 1)
    (listp g)
    (numberp n)
    (member k (nset (length 1)))
    (at 1 k 10)
    (lg-at-n n 1 g))
    (lg-at-n n (move 1 k 11) g))
  ((enable n-neq-k-mrhoi10 n-eq-k-mrhoi10)
  (use (n-neq-k-mrhoi10))
  (use (n-eq-k-mrhoi10))))

(disable lg-at-mrhoi10)

(prove-lemma lg-mrhoi10 (rewrite)
  (implies (and (listp 1)
    (listp g)
    (member k (nset (length 1)))
    (numberp n)
    (at 1 k 10)
    (lg n 1 g))
    (lg n (move 1 k 11) g))
  ((enable lg-at-mrhoi10 lg at)))

(disable lg-mrhoi10)

(prove-lemma mrhoi10-preserves-lg (rewrite)
  (implies (and (molws n 1 g h)
    (member k (nset n))
    (mrhoi10 n k 1 g h lp gp hp)
    (lg n 1 g))
    (lg n lp gp))
  ((enable lg-mrhoi10)))

:::mrhoi11a
(prove-lemma n-neq-k-mrhoi11a (rewrite)
  (implies (and (listp 1)
    (listp g)
    (numberp n)
    (member k (nset (length 1)))
    (not (equal k n))
    (at 1 k 11)
    (lg-at-n n 1 g))
    (lg-at-n n (move 1 k 12) g))
  ((enable at lg-at-n lg-1-at-n lg-2-at-n lg-3-at-n)))

(disable n-neq-k-mrhoi11a)

(prove-lemma n-eq-k-mrhoi11a (rewrite)
  (implies (and (listp 1)
    (listp g)
    (member k (nset (length 1)))
    (at 1 k 11)
    (lg-at-n k 1 g))
    (lg-at-n k (move 1 k 12) g))
  ((enable at lg-at-n lg-1-at-n lg-2-at-n lg-3-at-n)))

(disable n-eq-k-mrhoi11a)

(prove-lemma lg-at-mrhoi11a (rewrite)
  (implies (and (listp 1)
    (listp g)
    (numberp n)
    (member k (nset (length 1)))
    (at 1 k 11)

```

```

      (lg-at-n n 1 g))
      (lg-at-n n (move 1 k 12) g))
    ((enable n-neq-k-mrhoilla n-eq-k-mrhoilla)
     (use (n-neq-k-mrhoilla))
     (use (n-eq-k-mrhoilla))))
(disable lg-at-mrhoilla)

(prove-lemma lg-mrhoilla (rewrite)
  (implies (and (listp 1)
                (listp g)
                (member k (nset (length 1)))
                (numberp n)
                (at 1 k 11)
                (lg n 1 g))
            (lg n (move 1 k 12) g))
    ((enable lg-at-mrhoilla lg at))))
(disable lg-mrhoilla)

(prove-lemma mrhoilla-preserves-lg (rewrite)
  (implies (and (molws n 1 g h)
                (member k (nset n))
                (mrhoilla n k 1 g h lp gp hp)
                (lg n 1 g))
            (lg n lp gp))
    ((enable lg-mrhoilla))))

:::mrhoillb
(prove-lemma mrhoillb-preserves-lg (rewrite)
  (implies (and (molws n 1 g h)
                (member k (nset n))
                (mrhoillb n k 1 g h lp gp hp)
                (lg n 1 g))
            (lg n lp gp))
    ((enable mrhoillb))))

:::mrhoil2
(prove-lemma n-neq-k-mrhoil2 (rewrite)
  (implies (and (listp 1)
                (listp g)
                (numberp n)
                (member k (nset (length 1)))
                (not (equal k n))
                (at 1 k 12)
                (lg-at-n n 1 g))
            (lg-at-n n (move 1 k 0) (move g k 0)))
    ((enable at lg-at-n lg-1-at-n lg-2-at-n lg-3-at-n))))
(disable n-neq-k-mrhoil2)

(prove-lemma n-eq-k-mrhoil2 (rewrite)
  (implies (and (listp 1)
                (listp g)
                (member k (nset (length 1)))
                (at 1 k 12)
                (lg-at-n k 1 g))
            (lg-at-n n (move 1 k 0) (move g k 0)))
    ((enable at lg-at-n lg-1-at-n lg-2-at-n lg-3-at-n))))
(disable n-eq-k-mrhoil2)

(prove-lemma lg-at-mrhoil2 (rewrite)
  (implies (and (listp 1)
                (listp g)
                (numberp n)
                (member k (nset (length 1)))
                (at 1 k 12)
                (lg-at-n n 1 g))
            (lg-at-n n (move 1 k 0) (move g k 0)))
    ((enable n-neq-k-mrhoil2 n-eq-k-mrhoil2)
     (use (n-neq-k-mrhoil2))
     (use (n-eq-k-mrhoil2))))
(disable lg-at-mrhoil2)

(prove-lemma lg-mrhoil2 (rewrite)
  (implies (and (listp 1)
                (listp g)
                (member k (nset (length 1)))
                (numberp n)
                (at 1 k 12)
                (lg n 1 g))
            (lg n (move 1 k 0) (move g k 0)))
    ((enable lg-at-mrhoil2 lg at))))
(disable lg-mrhoil2)

(prove-lemma mrhoil2-preserves-lg (rewrite)
  (implies (and (molws n 1 g h)
                (member k (nset n))
                (mrhoil2 n k 1 g h lp gp hp)
                (lg n 1 g))
            (lg n lp gp))
    ((enable lg-mrhoil2))))

(prove-lemma mrho-preserves-lg (rewrite)
  (implies (and (molws n 1 g h)
                (member k (nset n))
                (mrho n k 1 g h lp gp hp)
                (lg n 1 g))
            (lg n lp gp))
    ((enable mrho))))

```

```

(member k (nset n))
(mrhoi n k 1 g h lp gp hp)
(lg n 1 g))
(lg n lp gp))
((disable mrhoi0 mrhoila mrhoilb mrhoi2 mrhoi3a
  mrhoi3b mrhoi4 mrhoi5a mrhoi5b mrhoi5c
  mrhoi6 mrhoi7a mrhoi7b mrhoi8 mrhoi9a
  mrhoi9b mrhoi10 mrhoilla mrhoillb mrhoil2)
 (enable mrhoi)))

(disable mrhoi0-preserves-lg)
(disable mrhoila-preserves-lg)
(disable mrhoilb-preserves-lg)
(disable mrhoi2-preserves-lg)
(disable mrhoi3a-preserves-lg)
(disable mrhoi3b-preserves-lg)
(disable mrhoi4-preserves-lg)
(disable mrhoi5a-preserves-lg)
(disable mrhoi5b-preserves-lg)
(disable mrhoi5c-preserves-lg)
(disable mrhoi6-preserves-lg)
(disable mrhoi7a-preserves-lg)
(disable mrhoi7b-preserves-lg)
(disable mrhoi8-preserves-lg)
(disable mrhoi9a-preserves-lg)
(disable mrhoi9b-preserves-lg)
(disable mrhoi10-preserves-lg)
(disable mrhoilla-preserves-lg)
(disable mrhoillb-preserves-lg)
(disable mrhoil2-preserves-lg)

```

```

##### b0a #####

#####Common in mole and atom.
(prove-lemma b0a-1f1 (rewrite)
  (implies (and (member j (nset n))
    (lg n 1 g)
    (not (at 4 j 1)))
    (not (at 1 j 4)))
  (enable lg lg-at-n lg-1-at-n nset at)))
#####:comm end.

(prove-lemma if1-nth-h-k (rewrite)
  (implies (and (molws n 1 g h)
    (member k (nset n))
    (member j (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (lg n 1 g)
    (at h k j)
    (not (at g (nth h k) 1)))
    (not (at 1 j 4)))
  ((enable at) (use (b0a-1f1)))))

(prove-lemma 15-not-g1 (rewrite)
  (implies (and (molws n 1 g h)
    (member k (nset n))
    (member j (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (at h k j)
    (at 1 k 5)
    (at lp k 5))
    (not (at g (nth h k) 1)))
  ((enable mrhoi at)))

(prove-lemma 15-nth-h-k-eq-j (rewrite)
  (implies (and (at h k j)
    (molws n 1 g h)
    (member k (nset n))
    (member j (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (lg n 1 g)
    (at 1 k 5)
    (at lp k 5))
    (not (at 1 j 4)))
  ((use (if1-nth-h-k) (15-not-g1)))))

(prove-lemma 15-j-1t-nth-k (rewrite)
  (implies (and (lessp j (nth h k))
    (molws n 1 g h)
    (member k (nset n))
    (member j (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (b0a n 1 h k j)
    (at 1 k 5))
    (not (at 1 j 4)))
  ((enable b0a)))))

(prove-lemma nth-k-1t-j-or-eq-j (rewrite)
  (implies (and (molws n 1 g h)
    (member k (nset n))
    (member j (nset n))
    (lessp (sub1 j) (nth h k))
    (not (lessp j (nth h k))))
    (at h k j))
  ((enable at)))

(prove-lemma lm-j-not-in-14 (rewrite)
  (implies (and (molws n 1 g h)
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (lg n 1 g)
    (b0a n 1 h k j)
    (at 1 k 5)
    (at lp k 5)
    (lessp (sub1 j) (nth h k)))
    (not (at 1 j 4)))
  ((use (nth-k-1t-j-or-eq-j)
    (use (15-j-1t-nth-k))
    (use (15-nth-h-k-eq-j)))))

(prove-lemma cond-15 (rewrite)
  (implies (and (molws n 1 g h)
    (member k (nset n))
    (member j (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (at 1 k 5)
    (lessp j (nth hp k)))
    (lessp (sub1 j) (nth h k)))
  ((enable mrhoi at)))

(prove-lemma j-not-in-14 (rewrite)
  (implies (and (molws n 1 g h)
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (lg n 1 g)
    (b0a n 1 h k j)
    (at 1 k 5)
    (at lp k 5))
    (not (at 1 j 4)))
  ((enable at)))

#####
(at lp k 5)
(lessp j (nth hp k)))
(not (at 1 j 4)))
((use (lm-j-not-in-14) (cond-15)))

(prove-lemma k-in-15 (rewrite)
  (implies (and (molws n 1 g h)
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (at lp k 5)
    (lessp j (nth hp k)))
    (at 1 k 5))
  ((enable mrhoi at)))

;;The order of the hints is crucial.
(prove-lemma lm-b0a-i-eq-k-j-neq-k (rewrite)
  (implies (and (molws n 1 g h)
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (lg n 1 g)
    (b0a n 1 h k j)
    (at lp k 5)
    (lessp j (nth hp k)))
    (not (at 1 j 4)))
  ((use (k-in-15) (j-not-in-14)))))

(prove-lemma b0a-i-eq-k-j-neq-k (rewrite)
  (implies (and (molws n 1 g h)
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (not (equal j k))
    (lg n 1 g)
    (b0a n 1 h k j))
    (b0a n lp hp k j))
  ((enable b0a)
  (use (lm-b0a-i-eq-k-j-neq-k))
  (use (m-1-same-lp-at-not (m 4)))))

(prove-lemma b0a-i-j-eq-k (rewrite)
  (implies (and (molws n 1 g h)
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (b0a n 1 h k k))
    (b0a n lp hp k k))
  ((enable at b0a)))

(prove-lemma b0a-i-eq-k (rewrite)
  (implies (and (molws n 1 g h)
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (lg n 1 g)
    (b0a n 1 h k j))
    (b0a n lp hp k j))
  ((use (b0a-i-eq-k-j-neq-k))
  (use (b0a-i-j-eq-k)))))

;;n-not-less-j is necessary.
(prove-lemma cond-lp4 (rewrite)
  (implies (and (molws n 1 g h)
    (member i (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (at 1 k 3)
    (not (lessp i (nth h k))))
    (not (at lp k 4)))
  ((enable mrhoi at)))

(prove-lemma not-13-then-lp4 (rewrite)
  (implies (and (molws n 1 g h)
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (not (at 1 k 3))
    (not (at lp k 4)))
  ((enable mrhoi at)))

(prove-lemma i-in-15 (rewrite)
  (implies (and (molws n 1 g h)
    (member i (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (b0b n 1 h i k)
    (at 1 i 5)
    (lessp k (nth h i)))
    (not (at lp k 4)))
  ((enable b0b)
  (use (cond-lp4) (not-13-then-lp4)))))

(prove-lemma lm-b0a-i-neq-k-j-eq-k (rewrite)
  (implies (and (molws n 1 g h)
    (member i (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (not (equal i k))
    (b0b n 1 h i k))
  ((enable at)))

```

```

(at lp i 5)
(lessp k (nth h i)))
(not (at lp k 4)))
((use (i-in-15))
 (use (m-l-same-lp-at (j i) (m 5)))))

(prove-lemma boa-i-neq-k-j-eq-k (rewrite)
 (implies (and (molws n 1 g h)
 (member i (nset n))
 (member k (nset n))
 (mrhoi n k 1 g h lp gp hp)
 (not (equal i k))
 (b0a n 1 h i k)
 (b0b n 1 h i k))
 (b0a n lp hp i k))
 (enable b0a)
 (use (lm-b0a-i-neq-k-j-eq-k))))

(prove-lemma boa-i-j-neq-k (rewrite)
 (implies (and (molws n 1 g h)
 (member i (nset n))
 (member j (nset n))
 (member k (nset n))
 (mrhoi n k 1 g h lp gp hp)
 (not (equal i k))
 (not (equal j k))
 (b0a n 1 h i j)
 (b0b n 1 h i j))
 (b0a n lp hp i j))
 (enable b0a)))

(prove-lemma boa-i-neq-k (rewrite)
 (implies (and (molws n 1 g h)
 (member i (nset n))
 (member j (nset n))
 (member k (nset n))
 (mrhoi n k 1 g h lp gp hp)
 (not (equal i k))
 (b0a n 1 h i j)
 (b0b n 1 h i j))
 (b0a n lp hp i j))
 (use (boa-i-j-neq-k))
 (use (boa-i-neq-k-j-eq-k))))

(prove-lemma rho-preserves-boa ()
 (implies (and (molws n 1 g h)
 (member i (nset n))
 (member j (nset n))
 (member k (nset n))
 (mrhoi n k 1 g h lp gp hp)
 (lg n 1 g)
 (b0a n 1 h i j)
 (b0b n 1 h i j))
 (b0a n lp hp i j))
 (use (boa-i-neq-k) (b0a-i-eq-k))))

::: b0b :::

:::Common in mole and atom.
(prove-lemma b0b-if1 (rewrite)
 (implies (and (member j (nset n))
 (lg n 1 g)
 (at 1 j 3))
 (at g j 1))
 (enable nset at lg lg-at-n lg-l-at-n)))

(prove-lemma b0b-if3 (rewrite)
 (implies (and (member i (nset n))
 (lg n 1 g)
 (at 1 i 5))
 (not (union-at-n g i '(0 1 2))))
 (enable lg lg-at-n lg-2-at-n union-at-n
 at nset)))
:::Common in mole and atom end.

(prove-lemma lm-j-neq-h-k (rewrite)
 (implies (and (molws n 1 g h)
 (member k (nset n))
 (member j (nset n))
 (mrhoi n k 1 g h lp gp hp)
 (lg n 1 g)
 (at 1 j 3)
 (not (at g (nth h k) 1)))
 (not (at h k j)))
 (enable at) (use (b0b-if1))))

(prove-lemma h-k-not-g1 (rewrite)
 (implies (and (molws n 1 g h)
 (member k (nset n))
 (mrhoi n k 1 g h lp gp hp)
 (at 1 k 5)
 (at lp k 5))
 (not (at g (nth h k) 1)))
 (enable mrhoi at)))

(prove-lemma j-neq-h-k (rewrite)
 (implies (and (molws n 1 g h)

```

```

(member j (nset n))
(member k (nset n))
(mrhoi n k 1 g h lp gp hp)
(lg n 1 g)
(at lp k 5)
(at 1 k 5)
(at 1 j 3))
(not (at h k j)))
((use (h-k-not-g1) (lm-j-neq-h-k))))

(prove-lemma n-k-leq-sub1-i (rewrite)
 (implies (and (molws n 1 g h)
 (member k (nset n))
 (member i (nset n))
 (mrhoi n k 1 g h lp gp hp)
 (not (at h k i))
 (not (lessp i (nth h k))))
 (not (lessp (sub1 i) (nth h k))))
 (enable at)))

:::This is proved with help of n-k-leq-sub1-i.
(prove-lemma lml-j-in-13 (rewrite)
 (implies (and (molws n 1 g h)
 (member j (nset n))
 (member k (nset n))
 (mrhoi n k 1 g h lp gp hp)
 (lg n 1 g)
 (b0b n 1 h k j)
 (at 1 k 5)
 (at lp k 5)
 (at 1 j 3)
 (lessp (sub1 j) (nth h k)))
 (not (lessp k (nth h j))))
 (enable b0b) (use (j-neq-h-k))))

(prove-lemma lm-j-in-13 (rewrite)
 (implies (and (molws n 1 g h)
 (member j (nset n))
 (member k (nset n))
 (mrhoi n k 1 g h lp gp hp)
 (lg n 1 g)
 (b0b n 1 h k j)
 (at 1 k 5)
 (at lp k 5)
 (at 1 j 3)
 (lessp j (nth hp k)))
 (not (lessp k (nth h j))))
 (use (lml-j-in-13) (cond-15))))

:::The order of hints is crucial.
(prove-lemma j-in-13 (rewrite)
 (implies (and (molws n 1 g h)
 (member j (nset n))
 (member k (nset n))
 (mrhoi n k 1 g h lp gp hp)
 (lg n 1 g)
 (b0b n 1 h k j)
 (at lp k 5)
 (at 1 j 3)
 (lessp j (nth hp k)))
 (not (lessp k (nth h j))))
 (use (k-in-15) (lm-j-in-13))))

(prove-lemma lm-bob-i-eq-k-j-neq-k (rewrite)
 (implies (and (molws n 1 g h)
 (member j (nset n))
 (member k (nset n))
 (mrhoi n k 1 g h lp gp hp)
 (not (equal j k))
 (lg n 1 g)
 (b0b n 1 h k j)
 (at lp k 5)
 (at lp j 3)
 (lessp j (nth hp k)))
 (not (lessp k (nth h j))))
 (use (j-in-13))))

(prove-lemma b0b-i-eq-k-j-neq-k (rewrite)
 (implies (and (molws n 1 g h)
 (member j (nset n))
 (member k (nset n))
 (mrhoi n k 1 g h lp gp hp)
 (not (equal j k))
 (lg n 1 g)
 (b0b n 1 h k j)
 (b0b n lp hp k j))
 (enable b0b) (use (lm-bob-i-eq-k-j-neq-k))))

(prove-lemma b0b-i-j-eq-k (rewrite)
 (implies (and (molws n 1 g h)
 (member k (nset n))
 (mrhoi n k 1 g h lp gp hp)
 (lg n 1 g)
 (b0b n lp hp k k))
 (enable b0b at)))

(prove-lemma b0b-i-eq-k (rewrite)
 (implies (and (molws n 1 g h)

```

```

(member j (nset n))
(member k (nset n))
(mrhoi n k l g h lp gp hp)
(lg n l g)
(b0b n l h k j))
(b0b n lp hp k j))
((use (b0b-i-eq-k-j-neq-k) (b0b-i-j-eq-k))))

(prove-lemma lm-i-neq-h-k (rewrite)
  (implies (and (molws n l g h)
    (member k (nset n))
    (member i (nset n))
    (mrhoi n k l g h lp gp hp)
    (lg n l g)
    (at l i 5)
    (union-at-n g (nth h k) '(0 1 2)))
    (not (at h k i))))
  ((enable at) (use (b0b-if3))))

(prove-lemma h-k-g02 (rewrite)
  (implies (and (molws n l g h)
    (member k (nset n))
    (mrhoi n k l g h lp gp hp)
    (at l k 3)
    (at lp k 3))
    (union-at-n g (nth h k) '(0 1 2)))
  ((enable at mrhoi)))

(prove-lemma i-neq-h-k (rewrite)
  (implies (and (molws n l g h)
    (member k (nset n))
    (member i (nset n))
    (mrhoi n k l g h lp gp hp)
    (lg n l g)
    (at l i 5)
    (at l k 3)
    (at lp k 3))
    (not (at h k i))))
  ((use (h-k-g02) (lm-i-neq-h-k))))

(prove-lemma lml-k-in-13-imp (rewrite)
  (implies (and (molws n l g h)
    (member k (nset n))
    (member i (nset n))
    (mrhoi n k l g h lp gp hp)
    (lg n l g)
    (at l i 5)
    (at l k 3)
    (at lp k 3)
    (not (lessp i (nth h k))))
    (not (lessp (sub1 i) (nth h k))))
  ((use (i-neq-h-k) (n-k-leq-sub1-i))))

(Drove-lemma lm-k-in-13-imp (rewrite)
  (implies (and (molws n l g h)
    (member k (nset n))
    (member i (nset n))
    (mrhoi n k l g h lp gp hp)
    (lg n l g)
    (b0b n l h i k)
    (at l i 5)
    (at l k 3)
    (at lp k 3)
    (lessp k (nth h i)))
    (not (lessp (sub1 i) (nth h k))))
  ((enable b0b) (use (lml-k-in-13-imp))))

(prove-lemma cond-13 (rewrite)
  (implies (and (molws n l g h)
    (member k (nset n))
    (member i (nset n))
    (mrhoi n k l g h lp gp hp)
    (at l k 3)
    (lessp i (nth hp k)))
    (lessp (sub1 i) (nth h k)))
  ((enable mrhoi at)))

(prove-lemma k-in-13-imp (rewrite)
  (implies (and (molws n l g h)
    (member i (nset n))
    (member k (nset n))
    (mrhoi n k l g h lp gp hp)
    (lg n l g)
    (b0b n l h i k)
    (at l i 5)
    (at l k 3)
    (at lp k 3)
    (lessp k (nth h i)))
    (not (lessp i (nth hp k))))
  ((use (lm-k-in-13-imp) (cond-13))))

(prove-lemma k-in-12-imp (rewrite)
  (implies (and (molws n l g h)
    (member i (nset n))
    (member k (nset n))
    (mrhoi n k l g h lp gp hp)
    (at lp k 3)
    (at l k 2)))

(not (lessp i (nth hp k))))
((enable mrhoi at)))

(not (lessp i (nth hp k))))
((enable mrhoi at)))

(prove-lemma lp3-then-12-or-13 (rewrite)
  (implies (and (molws n l g h)
    (member i (nset n))
    (member k (nset n))
    (mrhoi n k l g h lp gp hp)
    (at lp k 3)
    (not (at l k 2)))
    (at l k 3))
  ((enable mrhoi at)))

(prove-lemma bob-i-in-15 (rewrite)
  (implies (and (molws n l g h)
    (member i (nset n))
    (member k (nset n))
    (mrhoi n k l g h lp gp hp)
    (lg n l g)
    (b0b n l h i k)
    (at l i 5)
    (at lp k 3)
    (lessp k (nth h i)))
    (not (lessp i (nth hp k))))
  ((use (lp3-then-12-or-13) (k-in-13-imp)
    (k-in-12-imp))))

(prove-lemma lm-bob-i-neq-k-j-eq-k (rewrite)
  (implies (and (molws n l g h)
    (member i (nset n))
    (member k (nset n))
    (mrhoi n k l g h lp gp hp)
    (not (equal i k))
    (b0b n l h i k)
    (lg n l g)
    (at lp i 5)
    (at lp k 3)
    (lessp k (nth h i)))
    (not (lessp i (nth hp k))))
  ((use (bob-i-in-15) (m-l-same-lp-at (j i) (m 5))))

(prove-lemma bob-i-neq-k-j-eq-k (rewrite)
  (implies (and (molws n l g h)
    (member i (nset n))
    (member k (nset n))
    (mrhoi n k l g h lp gp hp)
    (not (equal i k))
    (b0b n l h i k)
    (lg n l g)
    (b0b n lp hp i k))
  ((enable b0b) (use (lm-bob-i-neq-k-j-eq-k))))

(prove-lemma bob-i-j-neq-k (rewrite)
  (implies (and (molws n l g h)
    (member i (nset n))
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k l g h lp gp hp)
    (not (equal i k))
    (not (equal j k))
    (b0b n l h i j))
    (b0b n lp hp i j))
  ((enable b0b)))

(prove-lemma bob-i-neq-k (rewrite)
  (implies (and (molws n l g h)
    (member i (nset n))
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k l g h lp gp hp)
    (not (equal i k))
    (lg n l g)
    (b0b n l h i j))
    (b0b n lp hp i j))
  ((use (bob-i-j-neq-k) (bob-i-neq-k-j-eq-k))))

(prove-lemma rho-preserves-bob ()
  (implies (and (molws n l g h)
    (member i (nset n))
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k l g h lp gp hp)
    (lg n l g)
    (b0b n l h i j))
    (b0b n lp hp i j))
  ((use (bob-i-neq-k) (b0b-i-eq-k))))

```

```

::: The order of the hints is crucial.
(prove-lemma lm-h-k-eq-addl-n-nex-hint (rewrite)
  (implies (and (not (zerop n))
    (listp h)
    (not (lessp n i))
    (member k (nset n))
    (lessp n (nth h k)))
    (not (exist-hint-8-12-3-4 i 1 g h k)))
  ((enable exist-hint-8-12-3-4
    hint-8-12-3-4-at-n at)))

(prove-lemma h-k-eq-addl-n-nex-hint (rewrite)
  (implies (and (molws n 1 g h)
    (member k (nset n))
    (at h k (addl n)))
    (not (exist-hint-8-12-3-4 n 1 g h k)))
  ((enable at)
    (use (lm-h-k-eq-addl-n-nex-hint (i n)))))

(prove-lemma h-k-eq-addl-n-k-not-in-13 (rewrite)
  (implies (and (molws n 1 g h)
    (member i (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (blb n 1 g h i k)
    (at h k (addl n))
    (union-at-n 1 i '(8 9 10 11 12)))
    (not (at 1 k 3)))
  ((enable blb)
    (use (h-k-eq-addl-n-nex-hint))))

(prove-lemma not-13-then-not-lp4 (rewrite)
  (implies (and (molws n 1 g h)
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (not (at 1 k 3)))
    (not (at lp k 4)))
  ((enable at mrhoi)))

(prove-lemma h-k-eq-addl-n (rewrite)
  (implies (and (molws n 1 g h)
    (member i (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (blb n 1 g h i k)
    (at h k (addl n))
    (union-at-n 1 i '(8 9 10 11 12)))
    (not (at lp k 4)))
  ((use (h-k-eq-addl-n-k-not-in-13))
    (use (not-13-then-not-lp4))))

(prove-lemma h-k-neq-addl-n (rewrite)
  (implies (and (molws n 1 g h)
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (not (at h k (addl n))))
    (not (at lp k 4)))
  ((enable at mrhoi)))

:::need m-1-same-lp.
(prove-lemma bla-i-neq-k-j-eq-k (rewrite)
  (implies (and (molws n 1 g h)
    (member i (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (blb n 1 g h i k)
    (union-at-n 1 i '(8 9 10 11 12)))
    (not (at lp k 4)))
  ((use (h-k-eq-addl-n) (h-k-neq-addl-n))))

(prove-lemma bla-i-j-neq-k (rewrite)
  (implies (and (molws n 1 g h)
    (member i (nset n))
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (not (equal i k))
    (not (equal j k))
    (bla 1 i j))
    (bla lp i j))
  ((enable bla)))

(prove-lemma bla-i-neq-k (rewrite)
  (implies (and (molws n 1 g h)
    (member i (nset n))
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (not (equal i k))
    (not (equal j k))
    (bla 1 i j))
    (bla lp i j))
  ((enable bla)))

(member k (nset n))
(mrhoi n k 1 g h lp gp hp)
(not (equal i k))
(bla 1 i j)
(blb n 1 g h i k))

(bla lp i j)
((use (bla-i-j-neq-k))
  (use (bla-i-neq-k-j-eq-k)))

(prove-lemma cond-17 (rewrite)
  (implies (and (molws n 1 g h)
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (at 1 k 7)
    (union-at-n lp k '(8 9 10 11 12)))
    (at g (nth h k) 4))
  ((enable mrhoi union-at-n at)))

(prove-lemma k-in-17-imp (rewrite)
  (implies (and (molws n 1 g h)
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (bld n 1 h k)
    (lg n 1 g)
    (at 1 k 7)
    (bla 1 (nth h k) j)
    (union-at-n lp k '(8 9 10 11 12)))
    (not (at 1 j 4)))
  ((enable bla bld)
    (use (cond-17) (bla-if4 (u (nth h k))))))

(prove-lemma 15-j-lt-h-k (rewrite)
  (implies (and (molws n 1 g h)
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (at 1 k 5)
    (union-at-n lp k '(8 9 10 11 12)))
    (lessp j (nth h k)))
  ((enable union-at-n at mrhoi)))

(prove-lemma k-in-15-then-j-not-14 (rewrite)
  (implies (and (molws n 1 g h)
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (at 1 k 5)
    (union-at-n lp k '(8 9 10 11 12))
    (b0a n 1 h k j))
    (not (at 1 j 4)))
  ((enable b0a) (use (15-j-lt-h-k))))

(prove-lemma lp9-12-k-in-18-12 (rewrite)
  (implies (and (molws n 1 g h)
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (union-at-n lp k '(9 10 11 12)))
    (union-at-n 1 k '(8 9 10 11 12)))
  ((enable at union-at-n mrhoi)))

(prove-lemma k-in-lp9-12-then-j-not-14 (rewrite)
  (implies (and (molws n 1 g h)
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (bla 1 k j)
    (union-at-n lp k '(9 10 11 12)))
    (not (at 1 j 4)))
  ((enable bla) (use (lp9-12-k-in-18-12))))

(prove-lemma k-not-in-17-then-lp9-12-or-15 (rewrite)
  (implies (and (molws n 1 g h)
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (union-at-n lp k '(8 9 10 11 12))
    (not (at 1 k 7))
    (not (union-at-n lp k '(9 10 11 12))))
    (at 1 k 5))
  ((enable at union-at-n mrhoi)))

(prove-lemma k-in-not-17-imp (rewrite)
  (implies (and (molws n 1 g h)
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (not (at 1 k 7))
    (b0a n 1 h k j)
    (bla 1 k j)
    (union-at-n lp k '(8 9 10 11 12)))
    (not (at 1 j 4)))
  ((use (k-not-in-17-then-lp9-12-or-15))
    (use (k-in-lp9-12-then-j-not-14))
    (use (k-in-15-then-j-not-14))))

::: I wonder why the following two do not imply
::: lm-bla-i-eq-k-j-neq-k although those without
::: (member u (nset n)) are perfectly able to

```

```

:::imply it.
:::(prove-lemma k-in-lp9-12-then-j-not-14 (rewrite)
::: (implies (and (molws n 1 g h)
::: (member j (nset n))
::: (member k (nset n))
::: (mrhoi n k 1 g h lp gp hp)
::: (bla 1 k j)
::: (union-at-n lp k '(9 10 11 12)))
::: (not tat 1 j 4))))
:::
:::(prove-lemma k-in-lp8-then-j-not-14 (rewrite)
::: (implies (and (at lp k 8)
::: (molws n 1 g h)
::: (member j (nset n))
::: (member u (nset n))
::: (member k (nset n))
::: (mrhoi n k 1 g h lp gp hp)
::: (lg n 1 g)
::: (b0a n 1 h k j)
::: (bla 1 (nth h k) j))
::: (not tat 1 j 4))))

```

```

(prove-lemma lm-bla-i-eq-k-j-neq-k (rewrite)
(implies (and (molws n 1 g h)
(member j (nset n))
(member k (nset n))
(mrhoi n k 1 g h lp gp hp)
(bld n 1 h k)
(lg n 1 g)
(b0a n 1 h k j)
(bla 1 k j)
(bla 1 (nth h k) j)
(union-at-n lp k '(8 9 10 11 12)))
(not tat 1 j 4)))
((use (k-in-17-imp))
(use (k-in-not-17-imp))))

```

;;m-1-same-lp-at-not is used.

```

(prove-lemma bla-i-eq-k-j-neq-k (rewrite)
(implies (and (molws n 1 g h)
(member j (nset n))
(member k (nset n))
(mrhoi n k 1 g h lp gp hp)
(not (equal j k))
(bld n 1 h k)
(lg n 1 g)
(b0a n 1 h k j)
(bla 1 k j)
(bla 1 (nth h k) j))
(bla lp k j))
(enable bla) (use (lm-bla-i-eq-k-j-neq-k))))

```

```

(prove-lemma bla-i-j-eq-k (rewrite)
(implies (and (molws n 1 g h)
(member k (nset n))
(mrhoi n k 1 g h lp gp hp)
(bla 1 k k))
(bla lp k k))
(enable bla at union-at-n))

```

```

(prove-lemma bla-i-eq-k (rewrite)
(implies (and (molws n 1 g h)
(member j (nset n))
(member k (nset n))
(mrhoi n k 1 g h lp gp hp)
(bld n 1 h k)
(lg n 1 g)
(b0a n 1 h k j)
(bla 1 k j)
(bla 1 (nth h k) j))
(bla lp k j))
((use (bla-i-eq-k-j-neq-k))
(use (bla-i-j-eq-k))))

```

```

(prove-lemma mrho-preserves-bla ()
(implies (and (molws n 1 g h)
(member i (nset n))
(member j (nset n))
(member k (nset n))
(mrhoi n k 1 g h lp gp hp)
(bld n 1 h i)
(lg n 1 g)
(b0a n 1 h i j)
(bla 1 i j)
(bla 1 (nth h i) j)
(bld n 1 g h i j))
(bla lp i j))
((use (bla-i-neq-k) (bla-i-eq-k))))

```

:::::::::::::::: blb ::::::::::::::

::::::::::::::::common in mole and atom.

```

(prove-lemma un8-11-then-un8-12 (rewrite)
(implies (union-at-n lp r '(8 9 10 11))
(union-at-n lp r '(8 9 10 11 12)))
(enable union-at-n))

```

```

(prove-lemma l8-11-k-in-gp34 (rewrite)
(implies (and (member r (nset n))
(lg n 1 g)
(union-at-n 1 r '(8 9 10 11)))
(union-at-n gp r '(3 4)))
(enable lg lg-at-n lg-2-at-n lg-3-at-n
union-at-n at nset)))

```

```

(prove-lemma u-if4 (rewrite)
(implies (and (member u (nset n))
(lg n 1 g)
tat g u 4)
(not (at 1 u 2)))
(enable lg lg-at-n lg-3-at-n at nset)))

```

```

(prove-lemma l12-then-un10-12 (rewrite)
(implies (at 1 u 12)
(union-at-n 1 u '(10 11 12)))
(enable at union-at-n))

```

```

(prove-lemma r-neq-k (rewrite)
(implies (and (union-at-n 1 k '(8 9 10 11))
(at 1 r 12))
(not (equal k r)))
(enable union-at-n at))

```

::::::::::::::::common in mole and atom end.
::::::::::::::::Lemmas on hints.

```

(prove-lemma ex-hint-in-18-12 (rewrite)
(implies (exist-hint-8-12-3-4 n 1 g h j)
(union-at-n 1 (exist-hint-8-12-3-4 n
1 g h j) '(8 9 10 11 12)))
(enable exist-hint-8-12-3-4 union-at-n at
hint-8-12-3-4-at-n
intersect-8-12-3-4-at-n)))

```

```

(prove-lemma ex-hint-in-g34 (rewrite)
(implies (exist-hint-8-12-3-4 n 1 g h k)
(union-at-n g (exist-hint-8-12-3-4 n
1 g h k) '(3 4)))
(enable exist-hint-8-12-3-4 union-at-n at
hint-8-12-3-4-at-n
intersect-8-12-3-4-at-n)))

```

```

(prove-lemma ex-hint-1-g-h (rewrite)
(implies (exist-hint-8-12-3-4 n 1 g h j)
(not (lessp (exist-hint-8-12-3-4 n
1 g h j) (nth h j))))
(enable exist-hint-8-12-3-4 hint-8-12-3-4-at-n)))

```

```

(prove-lemma ex-hint-lp-gp-h-in-int-8-12-3-4 (rewrite)
(implies (exist-hint-8-12-3-4 n lp gp h j)
(intersect-8-12-3-4-at-n
(exist-hint-8-12-3-4 n lp gp h j) lp gp))
(enable hint-8-12-3-4-at-n exist-hint-8-12-3-4))

```

```

(prove-lemma ex-hint-lp-gp-h-leq-h-j (rewrite)
(implies (exist-hint-8-12-3-4 n lp gp h j)
(not (lessp (exist-hint-8-12-3-4 n
lp gp h j) (nth h j))))
(enable hint-8-12-3-4-at-n exist-hint-8-12-3-4))

```

```

(prove-lemma ex-hint-not-in-g02 (rewrite)
(implies (exist-hint-8-12-3-4 n 1 g h k)
(not (union-at-n g (exist-hint-8-12-3-4 n
1 g h k) '(0 1 2))))
(enable union-at-n)
(use (ex-hint-in-g34)))

```

```

(prove-lemma hint-wtn (rewrite)
(implies (and (member r (nset n))
(intersect-8-12-3-4-at-n r lp gp)
(not (lessp r (nth h j))))
(exist-hint-8-12-3-4 n lp gp h j))
(enable nset exist-hint-8-12-3-4
hint-8-12-3-4-at-n))

```

::::::::::::::::Lemmas on hints end.

```

(prove-lemma lm-k-in-17-imp (rewrite)
(implies (and (molws n 1 g h)
(member k (nset n))
(mrhoi n k 1 g h lp gp hp)
(bld n 1 h k)
(lg n 1 g)
(at 1 k 7)
(bldnlg h (nth h k) j)
tat 1 j 3)
(union-at-n lp k '(8 9 10 11 12)))
(exist-hint-8-12-3-4 n 1 g h j))
(enable blb bld)
(use (cond-17) (bla-if4 (u (nth h k))))))

```

```

(prove-lemma ex-hint-neq-k-imp (rewrite)
(implies (and (molws n 1 g h)
(member k (nset n))
(mrhoi n k 1 g h lp gp hp)
(exist-hint-8-12-3-4 n 1 g h j)

```

```

(not (equal k
  (exist-hint-8-12-3-4 n 1 g h j))))
(intersect-8-12-3-4-at-n
  (exist-hint-8-12-3-4 n 1 g h j) lp gp))
(enable intersect-8-12-3-4-at-n
  (use (ex-hint-in-g34))
  (use (ex-hint-in-18-12))))

(prove-lemma ex-hint-neq-k-in-17 (rewrite)
  (implies (and (at 1 k 7)
    (exist-hint-8-12-3-4 n 1 g h j))
    (not (equal k (exist-hint-8-12-3-4 n
      1 g h j))))
    ((enable at union-at-n) (use (ex-hint-in-18-12))))

(prove-lemma ex-hint-in-int-8-12-3-4-17 (rewrite)
  (implies (and (molws n 1 g h)
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (at 1 k 7)
    (exist-hint-8-12-3-4 n 1 g h j))
    (intersect-8-12-3-4-at-n
      (exist-hint-8-12-3-4 n 1 g h j) lp gp))
    ((use (ex-hint-neq-k-imp))
    (use (ex-hint-neq-k-in-17))))

(prove-lemma ex-hint-wtn-17 (rewrite)
  (implies (and (molws n 1 g h)
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (at 1 k 7)
    (exist-hint-8-12-3-4 n 1 g h j))
    (exist-hint-8-12-3-4 n lp gp h j))
    ((use (hint-wtn
      (r (exist-hint-8-12-3-4 n 1 g h j))))
    (use (ex-hint-in-int-8-12-3-4-17))
    (use (ex-hint-1-g-h))))

(prove-lemma blb-k-in-17-imp (rewrite)
  (implies (and (molws n 1 g h)
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (bld n 1 h k)
    (lg n 1 g)
    (at 1 k 7)
    (blbnlgh (nthhk) j)
    (at 1 j 3)
    (union-at-n lp k '(8 9 10 11 12)))
    (exist-hint-8-12-3-4 n lp gp h j))
    ((use (lm-k-in-17-imp))
    (use (ex-hint-wtn-17))))

(prove-lemma h-j-leq-k (rewrite)
  (implies (and (molws n 1 g h)
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (at 1 j 3)
    (at 1 k 5)
    (b0b n 1 h k j)
    (union-at-n lp k '(8 9 10 11 12)))
    (not (lessp k (nth h j))))
    ((enable b0b)
    (use (15-j-1t-h-k))))

(prove-lemma lm-lp8-then-k-in-g34 (rewrite)
  (implies (and (listp 1)
    (numberp n)
    (equal (length 1) n)
    (mrhoi n k 1 g h lp gp hp)
    (member k (nset n))
    (lg n 1 g)
    (at 1 k 5)
    (tat lp k 8))
    (union-at-n gp k '(3 4)))
    ((enable mrhoi at union-at-n lg lg-at-n
      lg-2-at-n lg-3-at-n nset)))

(disable lm-lp8-then-k-in-g34)

(prove-lemma lp8-then-k-in-g34 (rewrite)
  (implies (and (molws n 1 g h)
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (lg n 1 g)
    (at 1 k 5)
    (at lp k 8))
    (union-at-n gp k '(3 4)))
    ((enable lm-lp8-then-k-in-g34)
    (use (lm-lp8-then-k-in-g34))))

(prove-lemma lm-k-in-g34 (rewrite)
  (implies (and (molws n 1 g h)
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (lg n 1 g)
    (lg n lp gp)
    (at 1 k 5)

```

```

    (union-at-n lp k '(8 9 10 11 12)))
    (union-at-n gp k '(3 4)))
    ((use (un8-12-then-18-or-19-12))
    (use (lp8-then-k-in-g34))
    (use (lp9-12-then-k-in-g34))))

(prove-lemma k-in-g34 (rewrite)
  (implies (and (molws n 1 g h)
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (lg n 1 g)
    (at 1 k 5)
    (union-at-n lp k '(8 9 10 11 12)))
    (union-at-n gp k '(3 4)))
    ((use (lm-k-in-g34))
    (use (mrho-preserves-lg))))

(prove-lemma k-in-int (rewrite)
  (implies (and (molws n 1 g h)
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (lg n 1 g)
    (at 1 k 5)
    (union-at-n lp k '(8 9 10 11 12)))
    (intersect-8-12-3-4-at-n k lp gp))
    ((enable intersect-8-12-3-4-at-n)
    (use (k-in-g34))))

(prove-lemma k-in-15-imp (rewrite)
  (implies (and (molws n 1 g h)
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (lg n 1 g)
    (tat 1 j 3)
    (at 1 k 5)
    (b0b n 1 h k j)
    (union-at-n lp k '(8 9 10 11 12)))
    (exist-hint-8-12-3-4 n lp gp h j))
    ((use (k-in-int))
    (use (h-j-leq-k))
    (use (hint-wtn (r k)))))

:::This is slow.
(prove-lemma ex-hint-in-112 (rewrite)
  (implies (and (molws n 1 g h)
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (union-at-n 1 k '(8 9 10 11))
    (exist-hint-8-12-3-4 n 1 g h j)
    (at 1 (exist-hint-8-12-3-4 n 1 g h j) 12))
    (intersect-8-12-3-4-at-n
      (exist-hint-8-12-3-4 n 1 g h j) lp gp))
    ((use (r-neq-k (r (exist-hint-8-12-3-4 n 1 g h j))))
    (use (ex-hint-neq-k-imp))))

(prove-lemma r-neq-k-18-11-k-in-lp8-12 (rewrite)
  (implies (and (molws n 1 g h)
    (member r (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (not (equal r k))
    (union-at-n 1 r '(8 9 10 11)))
    (union-at-n lp r '(8 9 10 11)))
    ((use (m-lp-same-1 (j r) (m '(8 9 10 11)))))

(prove-lemma r-eq-k-18-11-k-in-lp8-12 (rewrite)
  (implies (and (molws n 1 g h)
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (union-at-n 1 k '(8 9 10 11)))
    (union-at-n lp k '(8 9 10 11 12)))
    ((enable union-at-n at mrhoi)))

(prove-lemma 18-11-k-in-lp8-12 (rewrite)
  (implies (and (molws n 1 g h)
    (member r (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (union-at-n 1 r '(8 9 10 11)))
    (union-at-n lp r '(8 9 10 11 12)))
    ((use (r-neq-k-18-11-k-in-lp8-12))
    (use (un8-11-then-un8-12))
    (use (r-eq-k-18-11-k-in-lp8-12))))

(prove-lemma hint-in-18-11 (rewrite)
  (implies (and (molws n 1 g h)
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (lg n 1 g)
    (exist-hint-8-12-3-4 n 1 g h j)
    (union-at-n 1 (exist-hint-8-12-3-4
      n 1 g h j) '(8 9 10 11)))
    (intersect-8-12-3-4-at-n
      (exist-hint-8-12-3-4 n 1 g h j) lp gp))
    ((enable intersect-8-12-3-4-at-n)
    (use (18-11-k-in-gp34
      (r (exist-hint-8-12-3-4 n 1 g h j)))))

```

```

(use (l8-11-k-in-lp8-12
  (r (exist-hint-8-12-3-4 n 1 g h j))))))

(prove-lemma ex-hint-not-in-l12 (rewrite)
  (implies (and (molws n 1 g h)
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (lg n 1 g)
    (exist-hint-8-12-3-4 n 1 g h j)
    (not (at 1 (exist-hint-8-12-3-4 n 1 g h j) 12)))
    (intersect-8-12-3-4-at-n
      (exist-hint-8-12-3-4 n 1 g h j) lp gp))
    ((use (hint-in-l8-11))
     (use (ex-hint-in-l8-12))
     (use (case-k (k (exist-hint-8-12-3-4 n 1 g h j))))))

(prove-lemma ex-hint-in-int-8-12-3-4-18-11 (rewrite)
  implies (and (molws n 1 g h)
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (lg n 1 g)
    (union-at-n 1 k '(8 9 10 11))
    (exist-hint-8-12-3-4 n 1 g h j))
    (intersect-8-12-3-4-at-n
      (exist-hint-8-12-3-4 n 1 g h j) lp gp))
    ((use (ex-hint-not-in-l12))
     (use (ex-hint-in-l12))))

(prove-lemma ex-hint-wtn-l8-11 (rewrite)
  (implies (and (molws n 1 g h)
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (lg n 1 g)
    (union-at-n 1 k '(8 9 10 11))
    (exist-hint-8-12-3-4 n 1 g h j))
    (exist-hint-8-12-3-4 n lp gp h j))
    ((use (hint-wtn (r (exist-hint-8-12-3-4 n 1 g h j))))
     (use (ex-hint-in-int-8-12-3-4-18-11))
     (use (ex-hint-l-g-h))))

(prove-lemma k-in-l8-11-imp (rewrite)
  (implies (and (molws n 1 g h)
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (lg n 1 g)
    (at 1 j 3)
    (blb n 1 g h k j)
    (union-at-n 1 k '(8 9 10 11)))
    (exist-hint-8-12-3-4 n lp gp h j))
    ((enable blb)
     (use (un8-11-then-un8-12))))

(prove-lemma m-lp9-12-k-in-l8-11 (rewrite)
  (implies (and (molws n 1 g h)
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (union-at-n lp k '(8 9 10 11 12)))
    (union-at-n 1 k '(8 9 10 11)))
    ((enable mrhoi union-at-n at)))

(prove-lemma k-in-lp9-12-imp (rewrite)
  (implies (and (molws n 1 g h)
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (lg n 1 g)
    (at 1 j 3)
    (blb n 1 g h k j)
    (union-at-n lp k '(8 9 10 11 12)))
    (exist-hint-8-12-3-4 n lp gp h j))
    ((use (k-in-l8-11-imp))
     (use (m-lp9-12-k-in-l8-11))))

(prove-lemma k-not-in-l7-imp (rewrite)
  (implies (and (molws n 1 g h)
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (lg n 1 g)
    (at 1 j 3)
    (not (at 1 k 7))
    (b0b n 1 h k j)
    (blb n 1 g h k j)
    (union-at-n lp k '(8 9 10 11 12)))
    (exist-hint-8-12-3-4 n lp gp h j))
    ((use (k-not-in-l7-then-lp9-12-or-15))
     (use (k-in-l5-imp))
     (use (k-in-lp9-12-imp))))

(prove-lemma lml-blb-i-eq-k-j-neq-k (rewrite)
  (implies (and (molws n 1 g h)
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (bld n 1 h k)
    (lg n 1 g)
    (at 1 j 3)
    (b0b n 1 h k j)
    (blb n 1 g h k j))
    (lml-blb-i-eq-k-j-neq-k))
    ((use (lml-blb-i-eq-k-j-neq-k))
     (use (j-neq-k-then-hp-eq-h))))

(blb n 1 g h (nth h k) j)
(union-at-n lp k '(8 9 10 11 12)))
(exist-hint-8-12-3-4 n lp gp h j))
((use (blb-k-in-l7-imp))
 (use (k-not-in-l7-imp))))

(prove-lemma ex-hint-lp-gp-h-leq-hp-j (rewrite)
  (implies (and (molws n 1 g h)
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (not (equal j k))
    (exist-hint-8-12-3-4 n lp gp h j))
    (not (lessp (exist-hint-8-12-3-4 n
      lp gp h j) (nth hp j))))
    ((use (ex-hint-lp-gp-h-leq-h-j))))

(prove-lemma j-neq-k-then-hp-eq-h (rewrite)
  (implies (and (molws n 1 g h)
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (not (equal j k))
    (exist-hint-8-12-3-4 n lp gp h j))
    (exist-hint-8-12-3-4 n lp gp h j))
    ((use (hint-wtn (h hp)
      (r (exist-hint-8-12-3-4 n lp gp h j))))
     (use (ex-hint-lp-gp-h-in-int-8-12-3-4))
     (use (ex-hint-lp-gp-h-leq-hp-j))))

(prove-lemma lm-blb-i-eq-k-j-neq-k (rewrite)
  (implies (and (molws n 1 g h)
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (bld n 1 h k)
    (not (equal j k))
    (lg n 1 g)
    (b0b n 1 h k j)
    (blb n 1 g h k j)
    (blb n 1 g h (nth h k) j)
    (at 1 j 3)
    (union-at-n lp k '(8 9 10 11 12)))
    (exist-hint-8-12-3-4 n lp gp hp j))
    ((use (lml-blb-i-eq-k-j-neq-k))
     (use (j-neq-k-then-hp-eq-h))))

(prove-lemma blb-i-eq-k-j-neq-k (rewrite)
  (implies (and (molws n 1 g h)
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (bld n 1 h k)
    (not (equal j k))
    (lg n 1 g)
    (b0b n 1 h k j)
    (blb n 1 g h k j)
    (blb n 1 g h (nth h k) j))
    (blb n lp gp hp k j))
    ((enable blb)
     (use (lm-blb-i-eq-k-j-neq-k))))

(prove-lemma blb-i-j-eq-k (rewrite)
  (implies (and (molws n 1 g h)
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (blb n 1 g h k k))
    (blb n lp gp hp k k))
    ((enable blb union-at-n at)))

;;;I wonder if (bld n 1 h i) is
;;;better than (bld n 1 h k).
(prove-lemma blb-i-eq-k (rewrite)
  (implies (and (molws n 1 g h)
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (bld n 1 h k)
    (lg n 1 g)
    (b0b n 1 h k j)
    (blb n 1 g h k j)
    (blb n 1 g h (nth h k) j))
    (blb n lp gp hp k j))
    ((use (blb-i-eq-k-j-neq-k))
     (use (blb-i-j-eq-k))))

;;;I wonder why the following two do not imply
;;;lm-blb-i-neq-k-j-eq-k.
;;;(prove-lemma k-not-in-l3 (rewrite)
;;;  (implies (and (molws n 1 g h)
;;;    (member i (nset n))
;;;    (member k (nset n))
;;;    (mrhoi n k 1 g h lp gp hp)
;;;    (not (equal i k))
;;;    (lg n 1 g)
;;;    (at lp k 3)
;;;    (not (at 1 k 3))
;;;    (union-at-n 1 i '(8 9 10 11 12))))

```

```

;;; (exist-hint-8-12-3-4 n lp gp hp k)))
;;;
;;; (prove-lemma k-in-13 (rewrite)
;;; (implies (and (molws n 1 g h)
;;; (member i (nset n))
;;; (member k (nset n))
;;; (mrhoi n k 1 g h lp gp hp)
;;; (at 1 k 3)
;;; (at lp k 3)
;;; (union-at-n 1 i '(8 9 10 11 12))))
;;; (exist-hint-8-12-3-4 n lp gp hp k)))
;;;
;;; (prove-lemma lm-blb-i-neq-k-j-eq-k (rewrite)
;;; (implies (and (molws n 1 g h)
;;; (member i (nset n))
;;; (member k (nset n))
;;; (mrhoi n k 1 g h lp gp hp)
;;; (not (equal i k))
;;; (lg n 1 g)
;;; (at lp k 3)
;;; (union-at-n 1 i '(8 9 10 11 12))))
;;; (exist-hint-8-12-3-4 n lp gp hp k)))

(prove-lemma ex-hint-leq-h-k (rewrite)
(implies (exist-hint-8-12-3-4 n 1 g h k)
(not (lessp (exist-hint-8-12-3-4 n 1 g h k)
(nth h k))))))

(prove-lemma h-k-leq-subl-ex-hint (rewrite)
(implies (and (exist-hint-8-12-3-4 n 1 g h k)
(not (equal (nth h k)
(exist-hint-8-12-3-4 n 1 g h k))))
(not (lessp (subl (exist-hint-8-12-3-4
n 1 g h k)) (nth h k))))
((enable at)
(use (ex-hint-leq-h-k))))

(prove-lemma ex-hint-neq-h-k (rewrite)
(implies (and (molws n 1 g h)
(member k (nset n))
(mrhoi n k 1 g h lp gp hp)
(exist-hint-8-12-3-4 n 1 g h k)
(at 1 k 3)
(at lp k 3)
(not (equal (nth h k)
(exist-hint-8-12-3-4 n 1 g h k))))
((use (ex-hint-not-in-g02))
(use (h-k-g02))))

(prove-lemma lm-hp-k-leq-ex-l-g-h (rewrite)
(implies (and (molws n 1 g h)
(member k (nset n))
(mrhoi n k 1 g h lp gp hp)
(at 1 k 3)
(at lp k 3)
(exist-hint-8-12-3-4 n 1 g h k)
(not (lessp (subl (exist-hint-8-12-3-4 n
1 g h k)) (nth h k))))
((use (h-k-leq-subl-ex-hint))
(use (ex-hint-neq-h-k))))

(prove-lemma ex-cond-13 (rewrite)
(implies (and (molws n 1 g h)
(member k (nset n))
(mrhoi n k 1 g h lp gp hp)
(at 1 k 3)
(exist-hint-8-12-3-4 n 1 g h k)
(not (lessp (subl (exist-hint-8-12-3-4 n
1 g h k)) (nth h k))))
(not (lessp (exist-hint-8-12-3-4 n 1 g h k)
(nth hp k))))
((use (cond-13 (i (exist-hint-8-12-3-4 n 1 g h k))))))

(prove-lemma hp-k-leq-ex-l-g-h (rewrite)
(implies (and (molws n 1 g h)
(member k (nset n))
(mrhoi n k 1 g h lp gp hp)
(at 1 k 3)
(at lp k 3)
(exist-hint-8-12-3-4 n 1 g h k)
(not (lessp (exist-hint-8-12-3-4 n 1 g h k)
(nth hp k))))
((use (hint-member (j k)))
(use (lm-hp-k-leq-ex-l-g-h))
(use (ex-cond-13))))

(prove-lemma ex-hint-neq-k-in-13 (rewrite)
(implies (and (at 1 k 3)
(union-at-n 1 (exist-hint-8-12-3-4 n 1 g h k)
'(8 9 10 11 12)))
(not (equal k (exist-hint-8-12-3-4 n 1 g h k))))
((enable at union-at-n) (use (ex-hint-in-18-12))))

;;;This is successfully proved
;;;by m-gp-same-g and m-lp-same-l.
;;;This is successfully proved by ex-hint-neq-k-imp,

```

```

;;;ex-hint-neq-k-in-13 and ex-hint-in-18-12.
(prove-lemma ex-hint-l-g-h-in-int-8-12-3-4 (rewrite)
(implies (and (molws n 1 g h)
(member k (nset n))
(mrhoi n k 1 g h lp gp hp)
(at 1 k 3)
(exist-hint-8-12-3-4 n 1 g h k)
(intersect-8-12-3-4-at-n
(exist-hint-8-12-3-4 n 1 g h k) lp gp))
((use (ex-hint-neq-k-imp))
(use (ex-hint-neq-k-in-13))))

(prove-lemma ex-l-g-h-k-in-13 (rewrite)
(implies (and (molws n 1 g h)
(member k (nset n))
(mrhoi n k 1 g h lp gp hp)
(at 1 k 3)
(at lp k 3)
(exist-hint-8-12-3-4 n 1 g h k)
(exist-hint-8-12-3-4 n lp gp hp k)
((use (hint-wtn (h hp) (j k)
(r (exist-hint-8-12-3-4 n 1 g h k))))
(use (ex-hint-l-g-h-in-int-8-12-3-4))
(use (hp-k-leq-ex-l-g-h))))

(prove-lemma lm-k-in-13 (rewrite)
(implies (and (molws n 1 g h)
(member i (nset n))
(member k (nset n))
(mrhoi n k 1 g h lp gp hp)
(at 1 k 3)
(at lp k 3)
(union-at-n 1 i '(8 9 10 11 12))
(blb n 1 g h i k))
(exist-hint-8-12-3-4 n lp gp hp k)
((enable blb)
(use (ex-l-g-h-k-in-13))))

(prove-lemma k-in-13 (rewrite)
(implies (and (molws n 1 g h)
(member i (nset n))
(member k (nset n))
(mrhoi n k 1 g h lp gp hp)
(not (equal i k))
(at 1 k 3)
(blb n 1 g h i k))
(blb n lp gp hp i k)
((enable blb)
(use (lm-k-in-13))))

(prove-lemma hp-k-leq-i (rewrite)
(implies (and (molws n 1 g h)
(member i (nset n))
(member k (nset n))
(mrhoi n k 1 g h lp gp hp)
(at 1 k 2)
(at lp k 3)
(not (lessp i (nth hp k))))
((enable mrhoi at)))

(prove-lemma blb-u-neq-k (rewrite)
(implies (and (member u (nset n))
(member k (nset n))
(lg n 1 g)
(at g u 4)
(at 1 k 2))
(not (equal k u)))
((use (u-if4))))

(prove-lemma lm-u-in-int-8-12-3-4 (rewrite)
(implies (and (molws n 1 g h)
(member u (nset n))
(member k (nset n))
(mrhoi n k 1 g h lp gp hp)
(lg n 1 g)
(at 1 k 2)
(at g u 4)
(union-at-n lp u '(8 9 10 11 12)))
(intersect-8-12-3-4-at-n u lp gp))
((enable intersect-8-12-3-4-at-n)
(use (blb-u-neq-k))))

(prove-lemma k-neq-u-in-lp8-12 (rewrite)
(implies (and (molws n 1 g h)
(member u (nset n))
(member k (nset n))
(mrhoi n k 1 g h lp gp hp)
(not (equal u k))
(lg n 1 g)
(at g u 4)
(union-at-n lp u '(8 9 10 11 12)))
((use (bla-if4))))

(prove-lemma lml-u-in-int-8-12-3-4 (rewrite)
(implies (and (molws n 1 g h)
(member u (nset n))
(member k (nset n))
(mrhoi n k 1 g h lp gp hp)

```

```

      (lg n 1 g)
      (at 1 k 2)
      (at g u 4))
    (union-at-n lp u '(8 9 10 11 12)))
  ((use (k-neq-u-in-lp8-12))))

(prove-lemma u-in-int-8-12-3-4 (rewrite)
  (implies (and (molws n 1 g h)
    (member u (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (lg n 1 g)
    (at 1 k 2)
    (at g u 4))
    (intersect-8-12-3-4-at-n u lp gp))
  ((use (lm-u-in-int-8-12-3-4))
  (use (lml-u-in-int-8-12-3-4))))

:::I wonder why the following does not trigger
:::molws-lp, molws-lp-gp.
::: (prove-lemma h-i-in-g34-imp (rewrite)
:::   (implies (and (molws n 1 g h)
:::     (member k (nset n))
:::     (mrhoi n k 1 g h lp gp hp)
:::     (member (nth h i) (nset n))
:::     (lg n 1 g)
:::     (at 1 k 2)
:::     (at lp k 3)
:::     (at g (nth h i) 4))
:::     (exist-hint-8-12-3-4 n lp gp hp k)))
:::although
::: (prove-lemma h-i-in-g34-imp (rewrite)
:::   (implies (and (member (nth h i) (nset n))
:::     (molws n 1 g h)
:::     (member k (nset n))
:::     (mrhoi n k 1 g h lp gp hp)
:::     (lg n 1 g)
:::     (at 1 k 2)
:::     (at lp k 3)
:::     (at g (nth h i) 4))
:::     (exist-hint-8-12-3-4 n lp gp hp k)))
:::does.

(prove-lemma h-i-in-g34-imp (rewrite)
  (implies (and (member (nth h i) (nset n))
    (molws n 1 g h)
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (lg n 1 g)
    (at 1 k 2)
    (at lp k 3)
    (at g (nth h i) 4))
    (exist-hint-8-12-3-4 n lp gp hp k))
  ((use (hint-wtn (h hp) (j k) (r (nth h i))))
  (use (hp-k-leq-i (i (nth h i))))
  (use (u-in-int-8-12-3-4 (u (nth h i))))))

(prove-lemma i-not-in-g34 (rewrite)
  (implies (and (not (union-at-n g i '(3 4)))
    (molws n 1 g h)
    (member i (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (lg n 1 g)
    (at 1 k 2)
    (at lp k 3)
    (blc n 1 g h i)
    (union-at-n 1 i '(8 9 10 11 12)))
    (exist-hint-8-12-3-4 n lp gp hp k))
  ((enable blc)
  (use (h-i-in-g34-imp))))

(prove-lemma i-in-int-8-12-3-4 (rewrite)
  (implies (and (union-at-n g i '(3 4))
    (molws n 1 g h)
    (member i (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (not (equal i k))
    (union-at-n 1 i '(8 9 10 11 12)))
    (intersect-8-12-3-4-at-n i lp gp))
  ((enable intersect-8-12-3-4-at-n)))

(prove-lemma i-in-g34 (rewrite)
  (implies (and (union-at-n g i '(3 4))
    (molws n 1 g h)
    (member i (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (not (equal i k))
    (at 1 k 2)
    (at lp k 3)
    (union-at-n 1 i '(8 9 10 11 12)))
    (exist-hint-8-12-3-4 n lp gp hp k))
  ((use (hint-wtn (h hp) (j k) (r i)))
  (use (i-in-int-8-12-3-4))
  (use (hp-k-leq-i))))

(prove-lemma k-in-12 (rewrite)
  (implies (and (molws n 1 g h)
    (member i (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (bld n 1 h k)
    (not (equal i k))
    (lg n 1 g)
    (at 1 k 2)
    (at lp k 3)
    (blc n 1 g h i)
    (union-at-n 1 i '(8 9 10 11 12)))
    (exist-hint-8-12-3-4 n lp gp hp k))
  ((use (i-not-in-g34))
  (use (i-in-g34))))

(prove-lemma lp3-then-13-or-12 (rewrite)
  (implies (and (molws n 1 g h)
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (not (at 1 k 3))
    (at lp k 3))
    (at 1 k 2))
  ((enable mrhoi at)))

(prove-lemma lm-k-not-in-13 (rewrite)
  (implies (and (molws n 1 g h)
    (member i (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (bld n 1 h k)
    (not (equal i k))
    (lg n 1 g)
    (not (at 1 k 3))
    (at lp k 3)
    (blc n 1 g h i)
    (union-at-n 1 i '(8 9 10 11 12)))
    (exist-hint-8-12-3-4 n lp gp hp k))
  ((use (k-in-12))
  (use (lp3-then-13-or-12))))

(prove-lemma k-not-in-13 (rewrite)
  (implies (and (molws n 1 g h)
    (member i (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (bld n 1 h k)
    (not (equal i k))
    (lg n 1 g)
    (not (at 1 k 3))
    (blb n 1 g h i k)
    (blc n 1 g h i))
    (blb n lp gp hp i k))
  ((enable blb)
  (use (lm-k-not-in-13))))

(prove-lemma blb-i-neq-k-j-eq-k (rewrite)
  (implies (and (molws n 1 g h)
    (member i (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (bld n 1 h k)
    (not (equal i k))
    (lg n 1 g)
    (blb n 1 g h i k)
    (blc n 1 g h i))
    (blb n lp gp hp i k))
  ((use (k-not-in-13))
  (use (k-in-13))))

(prove-lemma lm-i-neq-k-in-int-8-12-3-4 (rewrite)
  (implies (and (molws n 1 g h)
    (member i (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (not (equal i k))
    (lg n 1 g)
    (at g i 4))
    (intersect-8-12-3-4-at-n i lp gp))
  ((enable intersect-8-12-3-4-at-n)
  (use (k-neq-u-in-lp8-12 (u i)))))

(prove-lemma i-neq-k-in-int-8-12-3-4 (rewrite)
  (implies (and (molws n 1 g h)
    (member i (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (not (equal i k))
    (union-at-n 1 i '(8 9 10 11 12)))
    (lg n 1 g)
    (at 1 (exist-hint-8-12-3-4 n 1 g h j) 12)
    (b3a 1 g
      (exist-hint-8-12-3-4 n 1 g h j) i))
  ((enable b3a)
  (use (lm-i-neq-k-in-int-8-12-3-4))
  (use (un8-12-then-un5-12))))

```

```

(prove-lemma h-j-leq-i (rewrite)
  (implies (and (union-at-n 1 i '(8 9 10 11 12))
    (exist-hint-8-12-3-4 n 1 g h j)
    (at 1 (exist-hint-8-12-3-4 n 1 g h j) 12)
    (b2a 1 (exist-hint-8-12-3-4 n 1 g h j) i))
    (not (lessp i (nth h j)))))
(enable b2a)
(use (l12-then-un10-12
  (u (exist-hint-8-12-3-4 n 1 g h j))))
(use (un8-12-then-un5-12))
(use (ex-hint-1-g-h)))

(prove-lemma i-neq-k-ex-hint-in-112 (rewrite)
  (implies (and (molws n 1 g h)
    (member i (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (not (equal i k))
    (lg n 1 g)
    (union-at-n 1 i '(8 9 10 11 12))
    (exist-hint-8-12-3-4 n 1 g h j)
    (at 1 (exist-hint-8-12-3-4 n 1 g h j) 12)
    (b2a 1 (exist-hint-8-12-3-4 n 1 g h j) i)
    (b3a 1 g
      (exist-hint-8-12-3-4 n 1 g h j) i))
    (exist-hint-8-12-3-4 n lp gp h j))
  ((use (hint-wtn (r i)))
   (use (h-j-leq-i))
   (use (i-neq-k-in-int-8-12-3-4))))

(prove-lemma i-neq-k-ex-hint-not-in-112 (rewrite)
  (implies (and (molws n 1 g h)
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (lg n 1 g)
    (exist-hint-8-12-3-4 n 1 g h j)
    (not (at 1
      (exist-hint-8-12-3-4 n 1 g h j) 12)))
    (exist-hint-8-12-3-4 n lp gp h j))
  ((use (hint-wtn (r (exist-hint-8-12-3-4 n 1 g h j))))
   (use (ex-hint-not-in-112))
   (use (ex-hint-1-g-h))))

(prove-lemma lml-blb-i-j-neq-k (rewrite)
  (implies (and (molws n 1 g h)
    (member i (nset n))
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (not (equal i k))
    (lg n 1 g)
    (union-at-n 1 i '(8 9 10 11 12))
    (exist-hint-8-12-3-4 n 1 g h j)
    (b2a 1 (exist-hint-8-12-3-4 n 1 g h j) i)
    (b3a 1 g
      (exist-hint-8-12-3-4 n 1 g h j) i))
    (exist-hint-8-12-3-4 n lp gp h j))
  ((use (i-neq-k-ex-hint-in-112))
   (use (i-neq-k-ex-hint-not-in-112))))

(prove-lemma lm-blb-i-j-neq-k (rewrite)
  (implies (and (molws n 1 g h)
    (member i (nset n))
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (not (equal i k))
    (not (equal j k))
    (lg n 1 g)
    (union-at-n 1 i '(8 9 10 11 12))
    (b2a 1 (exist-hint-8-12-3-4 n 1 g h j) i)
    (b3a 1 g (exist-hint-8-12-3-4 n 1 g h j) i)
    (exist-hint-8-12-3-4 n 1 g h j))
    (exist-hint-8-12-3-4 n lp gp hp j))
  ((use (lml-blb-i-j-neq-k))
   (use (j-neq-k-then-hp-eq-h))))

(prove-lemma blb-i-j-neq-k (rewrite)
  (implies (and (molws n 1 g h)
    (member i (nset n))
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (not (equal i k))
    (not (equal j k))
    (lg n 1 g)
    (blb n 1 g h i j)
    (b2a 1 (exist-hint-8-12-3-4 n 1 g h j) i)
    (b3a 1 g
      (exist-hint-8-12-3-4 n 1 g h j) i))
    (exist-hint-8-12-3-4 n lp gp h j) i))
  ((enable blb)
   (use (lm-blb-i-j-neq-k))))

(prove-lemma blb-i-neq-k (rewrite)
  (implies (and (molws n 1 g h)
    (member i (nset n))
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (lg n 1 g)
    (union-at-n 1 k '(8 9 10 11 12))
    (b2a 1 (exist-hint-8-12-3-4 n 1 g h j) i)
    (b3a 1 g
      (exist-hint-8-12-3-4 n 1 g h j) i))
    (exist-hint-8-12-3-4 n lp gp h j) i))
  ((enable blb)
   (use (lm-blb-i-j-neq-k))))

```

```

(member k (nset n))
(mrhoi n k 1 g h lp gp hp)
(bld n 1 h k)
(not (equal i k))
(lg n 1 g)
(b0b n 1 h i j)
(blb n 1 g h i j)
(blc n 1 g h i)
(b2a 1 (exist-hint-8-12-3-4 n 1 g h j) i)
(b3a 1 g
  (exist-hint-8-12-3-4 n 1 g h j) i))
(blb n lp gp hp i j))
((use (blb-i-j-neq-k))
 (use (blb-i-neq-k-j-eq-k)))

;;;I wonder if (bld n 1 h i) and
;;;(bld n 1 h j) are better than
;;;(bld n 1 h k).
;;;(blb n 1 g h (nth h i) j) and (blb n 1 g h (nth h k) j).
;;;What about (b2a 1 (exist-hint-8-12-3-4 n 1 g h j) i)
;;;(b3a 1 g (exist-hint-8-12-3-4 n 1 g h j) i)) ?

(prove-lemma mrho-preserves-blb ()
  (implies (and (molws n 1 g h)
    (member i (nset n))
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (bld n 1 h k)
    (lg n 1 g)
    (b0b n 1 h i j)
    (blbnlgh (nth h k) j)
    (blb n 1 g h i j)
    (blc n 1 g h i)
    (b2a 1 (exist-hint-8-12-3-4 n 1 g h j) i)
    (b3a 1 g
      (exist-hint-8-12-3-4 n 1 g h j) i))
    (blb n lp gp hp i j))
  ((use (blb-i-neq-k) (blb-i-eq-k))))

;;;;;;;;; b l c ;;;;;;;;;;

;;;;;;;;;common in mole and atom.
(prove-lemma not-g34-then-not-g4 (rewrite)
  (implies (not (union-at-n g i '(3 4)))
    (not (at g i 4)))
  ((enable union-at-n at)))

(prove-lemma contra-if4 (rewrite)
  (implies (and (member j (nset n))
    (lg n 1 g)
    (at g j 4))
    (union-at-n 1 j '(9 10 11 12)))
  ((enable lg lg-at-n lg-3-at-n union-at-n at nset)))
;;;;;;;;;common in mole and atom end.

(prove-lemma lp8-not-15-then-17 (rewrite)
  (implies (and (molws n 1 g h)
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (not (at 1 k 5))
    (at lp k 8))
    (at 1 k 7))
  ((enable at mrhoi)))

(prove-lemma lp8-not-g34-then-k-in-17 (rewrite)
  (implies (and (molws n 1 g h)
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (lg n 1 g)
    (at lp k 8)
    (union-at-n lp k '(8 9 10 11 12))
    (not (union-at-n gp k '(3 4))))
    (at 1 k 7))
  ((use (l8-not-15-then-17))
   (use (lp8-then-k-in-g34))))

(prove-lemma lm-k-in-17 (rewrite)
  (implies (and (molws n 1 g h)
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (lg n 1 g)
    (lg n lp gp)
    (union-at-n lp k '(8 9 10 11 12))
    (not (union-at-n gp k '(3 4))))
    (at 1 k 7))
  ((use (un8-12-then-l8-or-l9-12))
   (use (lp9-12-then-k-in-g34))
   (use (lp8-not-g34-then-k-in-17))))

(prove-lemma k-in-17 (rewrite)
  (implies (and (molws n 1 g h)
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (lg n 1 g)
    (union-at-n lp k '(8 9 10 11 12))

```

```

(not (union-at-n gp k '(3 4)))
(at 1 k 7))
((use (lm-k-in-17))
 (use (mrho-preserves-lg)))

(prove-lemma h-k-cond-17 (rewrite)
 (implies (and (molws n 1 g h)
 (member k (nset n))
 (mrhoi n k 1 g h lp gp hp)
 (at 1 k 7)
 (union-at-n lp k '(8 9 10 11 12)))
 (equal (nth hp k) (nth h k)))
 (enable at union-at-n mrhoi)))

(prove-lemma lm-h-k-g4 (rewrite)
 (implies (and (molws n 1 g h)
 (member k (nset n))
 (mrhoi n k 1 g h lp gp hp)
 (bld n 1 h k)
 (lg n 1 g)
 (at 1 k 7)
 (union-at-n lp k '(8 9 10 11 12)))
 (and (member (nth hp k) (nset n))
 (at g (nth hp k) 4)))
 (enable bld)
 (use (h-k-cond-17))
 (use (k-in-17))
 (use (cond-17)))

(prove-lemma h-k-g4 (rewrite)
 (implies (and (molws n 1 g h)
 (member k (nset n))
 (mrhoi n k 1 g h lp gp hp)
 (bld n 1 h k)
 (lg n 1 g)
 (union-at-n lp k '(8 9 10 11 12))
 (not (union-at-n gp k '(3 4)))
 (and (member (nth hp k) (nset n))
 (at g (nth hp k) 4)))
 ((use (lm-h-k-g4))
 (use (k-in-17))
 (use (cond-17)))

(prove-lemma lml-i-eq-k-then-h-k-neq-k (rewrite)
 (implies (and (molws n 1 g h)
 (member k (nset n))
 (mrhoi n k 1 g h lp gp hp)
 (bld n 1 h k)
 (lg n 1 g)
 (at 1 k 7)
 (at g (nth hp k) 4))
 (not (at hp k k)))
 (enable at bld union-at-n)
 (use (contra-if4 (j (nth hp k))))))

;;:Need k-in-17.
(prove-lemma lm-i-eq-k-then-h-k-neq-k (rewrite)
 (implies (and (molws n 1 g h)
 (member k (nset n))
 (mrhoi n k 1 g h lp gp hp)
 (bld n 1 h k)
 (lg n 1 g)
 (at 1 k 7)
 (union-at-n lp k '(8 9 10 11 12))
 (not (union-at-n gp k '(3 4)))
 (not (at hp k k)))
 ((use (lml-i-eq-k-then-h-k-neq-k))
 (use (h-k-g4))))

(prove-lemma i-eq-k-then-h-k-neq-k (rewrite)
 (implies (and (molws n 1 g h)
 (member k (nset n))
 (mrhoi n k 1 g h lp gp hp)
 (bld n 1 h k)
 (lg n 1 g)
 (union-at-n lp k '(8 9 10 11 12))
 (not (union-at-n gp k '(3 4)))
 (not (at hp k k)))
 ((use (h-k-g4))
 (use (lm-i-eq-k-then-h-k-neq-k))
 (use (k-in-17))))

(prove-lemma blc-i-eq-k-hp-k-neq-k (rewrite)
 (implies (and (molws n 1 g h)
 (member k (nset n))
 (mrhoi n k 1 g h lp gp hp)
 (bld n 1 h k)
 (not (at hp k k))
 (lg n 1 g)
 (union-at-n lp k '(8 9 10 11 12))
 (not (union-at-n gp k '(3 4)))
 (and (member (nth hp k) (nset n))
 (at gp (nth hp k) 4)))
 (enable at)
 (use (h-k-g4))))

(prove-lemma blc-i-eq-k (rewrite)
 (implies (and (molws n 1 g h)

```

```

(member k (nset n))
(mrhoi n k 1 g h lp gp hp)
(bld n 1 h k)
(lg n 1 g)
(blc n lp gp hp k))
(enable blc)
(use (blc-i-eq-k-hp-k-neq-k))
(use (i-eq-k-then-h-k-neq-k)))

(prove-lemma l9-11-then-in-lp9-12 (rewrite)
 (implies (and (molws n 1 g h)
 (member k (nset n))
 (mrhoi n k 1 g h lp gp hp)
 (not (at 1 k 12))
 (union-at-n 1 k '(9 10 11 12)))
 (union-at-n lp k '(9 10 11 12)))
 (enable union-at-n at mrhoi)))

(prove-lemma k-in-lp9-12 (rewrite)
 (implies (and (molws n 1 g h)
 (member k (nset n))
 (mrhoi n k 1 g h lp gp hp)
 (lg n 1 g)
 (at g k 4)
 (not (at 1 k 12)))
 (union-at-n lp k '(9 10 11 12)))
 ((use (contra-if4 (j k)))
 (use (l9-11-then-in-lp9-12))))

(prove-lemma lm-k-not-in-l12-imp (rewrite)
 (implies (and (molws n 1 g h)
 (member k (nset n))
 (mrhoi n k 1 g h lp gp hp)
 (lg n 1 g)
 (lg n lp gp)
 (at g k 4)
 (not (at 1 k 12)))
 (at gp k 4))
 ((disable mrho-preserves-lg)
 (use (k-in-lp9-12))
 fuse (if4 (j k) (1 lp) (g gp))))

(prove-lemma k-not-in-l12-imp (rewrite)
 (implies (and (molws n 1 g h)
 (member k (nset n))
 (mrhoi n k 1 g h lp gp hp)
 (lg n 1 g)
 (at g k 4)
 (not (at 1 k 12)))
 (at gp k 4))
 ((use (lm-k-not-in-l12-imp))
 (use (mrho-preserves-lg))))

(prove-lemma k-not-in-l12 (rewrite)
 (implies (and (molws n 1 g h)
 (member i (nset n))
 (member k (nset n))
 (mrhoi n k 1 g h lp gp hp)
 (b3a 1 g k i)
 (union-at-n 1 i '(8 9 10 11 12))
 (not (union-at-n g i '(3 4)))
 (not (at 1 k 12)))
 (enable b3a)
 (use (un8-12-then-un5-12))
 (use (not-g34-then-not-g4))))

(prove-lemma lml-blc-i-neq-k-h-i-eq-k (rewrite)
 (implies (and (molws n 1 g h)
 (member i (nset n))
 (member k (nset n))
 (mrhoi n k 1 g h lp gp hp)
 (equal (nth h i) k)
 (lg n 1 g)
 (blc n 1 g h i)
 (b3a 1 g (nth h i) i)
 (union-at-n 1 i '(8 9 10 11 12))
 (not (union-at-n g i '(3 4)))
 (at gp k 4))
 (enable blc)
 (use (k-not-in-l12))
 (use (k-not-in-l12-imp))))

(prove-lemma lm-blc-i-neq-k-h-i-eq-k (rewrite)
 (implies (and (molws n 1 g h)
 (member i (nset n))
 (member k (nset n))
 (mrhoi n k 1 g h lp gp hp)
 (at h i k)
 (lg n 1 g)
 (blc n 1 g h i)
 (b3a 1 g (nth h i) i)
 (union-at-n 1 i '(8 9 10 11 12))
 (not (union-at-n g i '(3 4)))
 (and (member (nth h i) (nset n))
 (at gp (nth h i) 4)))
 (enable at)
 (use (lml-blc-i-neq-k-h-i-eq-k))))

```

```

(prove-lemma b3a-h-rholemma (rewrite)
  (implies (and (molws n 1 g h)
    (member i (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (not (equal i k))
    (b3a 1 g (nth h i) i))
    (b3a 1 g (nth hp i) i)))

::m-l-same-lp and m-gp-same-g are used.
(prove-lemma blc-i-neq-k-h-i-eq-k (rewrite)
  (implies (and (molws n 1 g h)
    (member i (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (not (equal i k))
    (at h i k)
    (lg n 1 g)
    (blc n 1 g h i)
    (b3a 1 g (nth h i) i))
    (b1c n lp gp hp i))
  ((enable blc)
  (use (lm-blc-i-neq-k-h-i-eq-k))
  (use (b3a-h-rholemma))))

(prove-lemma lm-blc-i-h-i-neq-k (rewrite)
  (implies (and (molws n 1 g h)
    (member i (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (not (at h i k))
    (blc n 1 g h i)
    (union-at-n 1 i '(8 9 10 11 12))
    (not (union-at-n g i '(3 4))))
    (and (member (nth h i) (nset n))
    (at gp (nth h i) 4)))
  ((enable blc at)
  (use (g-mrholemma (j (nth h i))))))

(prove-lemma blc-i-h-i-neq-k (rewrite)
  (implies (and (molws n 1 g h)
    (member i (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (not (equal i k))
    (not (at h i k))
    (blc n 1 g h i))
    (b1c n lp gp hp i))
  ((enable blc)
  (use (lm-blc-i-h-i-neq-k))))

(prove-lemma blc-i-neq-k (rewrite)
  (implies (and (molws n 1 g h)
    (member i (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (not (equal i k))
    (lg n 1 g)
    (blc n 1 g h i)
    (b3a 1 g (nth h i) i))
    (b1c n lp gp hp i))
  ((use (blc-i-h-i-neq-k))
  (use (blc-i-neq-k-h-i-eq-k))))

(prove-lemma mrho-preserves-b1c ()
  (implies (and (molws n 1 g h)
    (member i (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (bld n 1 h k)
    (lg n 1 g)
    (blc n 1 g h i)
    (b3a 1 g (nth h i) i))
    (b1c n lp gp hp i))
  ((use (blc-i-neq-k))
  (use (blc-i-eq-k))))

::: bld :::

(prove-lemma remainder-quotient (rewrite)
  (equal (remainder x (add1 x))
    (fix x)))

(prove-lemma lml-member-remainder (rewrite)
  (implies (not (lessp x n))
    (not (member (add1 x) (nset (sub1 n)))))
  ((enable nset)))

(prove-lemma lm-member-remainder (rewrite)
  (implies (member (add1 x) (nset (sub1 n)))
    (member (add1 (remainder x n)) (nset (sub1 n))))
  ((enable nset)))

(prove-lemma member-remainder (rewrite)
  (implies (member j (nset n))
    (member (add1 (remainder (sub1 j) n)) (nset n))))

```

```

((enable nset)))

(prove-lemma one-nset (rewrite)
  (implies (not (zerop n))
    (member 1 (nset n)))
  ((enable nset)))

(prove-lemma lm-bld-i-eq-k (rewrite)
  (implies (and (listp 1)
    (listp h)
    (numberp n)
    (numberp (nth h k))
    (equal (length 1) n)
    (equal (length h) n)
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (bld n 1 h k))
    (bld n lp hp k))
  ((enable mrhoi bld at)
  (use (member-remainder (j (nth h k))))))

(prove-lemma bld-i-eq-k (rewrite)
  (implies (and (molws n 1 g h)
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (bld n 1 h k))
    (bld n lp hp k))
  ((disable bld)
  (enable lm-bld-i-eq-k)
  (use (lm-bld-i-eq-k))))

(prove-lemma bld-neq-k (rewrite)
  (implies (and (molws n 1 g h)
    (member i (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (not (equal i k))
    (bld n 1 h i))
    (bld n lp hp i))
  ((enable bld)))

(prove-lemma mrhoi-preserves-bld (rewrite)
  (implies (and (molws n 1 g h)
    (member i (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (bld n 1 h i))
    (bld n lp hp i))
  ((disable bld)
  (use (bld-i-neq-k))
  (use (bld-i-eq-k))))

```

```

***** b2a *****
:* i-eq-k-j-neq-k

(prove-lemma j-lt-h-k (rewrite)
  (implies (and (molws n 1 g h)
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (lessp j k)
    (at 1 k 9)
    (union-at-n lp k '(10 11 12)))
    (lessp j (nth h k)))
    ((enable mrhoi union-at-n at))))

(prove-lemma lm-case-k-in-19 (rewrite)
  (implies (and (molws n 1 g h)
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (not (equal j k))
    (lessp j k)
    (b2b 1 h k j)
    (at 1 k 9)
    (lessp j (nth h k))
    (union-at-n lp k '(10 11 12)))
    (not (union-at-n 1 j '(5 6 7 8 9 10 11 12))))
    ((enable b2b)))

;;;In the rewrite rule a set of hypotheses is replaced
;;;by another set of formulas. Thus if in a proof
;;;intended beforehand there is a formula belonging to
;;;more than one set of hypotheses which are expected to
;;;be replaced, Bmp is very likely to be unsuccessful.
;;;If j is not equal to k and the k's entry in 1 is
;;;between 10 and 12, then the j's entry in lp is not
;;;between 5 and 12.
(prove-lemma case-k-in-19 (rewrite)
  (implies (and (molws n 1 g h)
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (not (equal j k))
    (lessp j k)
    (b2b 1 h k j)
    (at 1 k 9)
    (union-at-n lp k '(10 11 12)))
    (not (union-at-n lp j '(5 6 7 8 9 10 11 12))))
    ((use (lm-case-k-in-19) (j-lt-h-k))))

;;;need un10-11-then-un10-12.
(prove-lemma case-k-in-110-11 (rewrite)
  (implies (and (molws n 1 g h)
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (not (equal j k))
    (lessp j k)
    (b2a 1 k j)
    (union-at-n 1 k '(10 11))
    (not (union-at-n lp j '(5 6 7 8 9 10 11 12))))
    ((enable b2a)))

(prove-lemma k-in-110-11-or-19 (rewrite)
  (implies (and (molws n 1 g h)
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (union-at-n lp k '(10 11 12))
    (not (union-at-n 1 k '(10 11))))
    (at 1 k 9))
    ((enable mrhoi union-at-n at)))

(prove-lemma lm-b2a-i-eq-k-j-neq-k (rewrite)
  (implies (and (molws n 1 g h)
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (not (equal j k))
    (lessp j k)
    (lg n 1 g)
    (b2a 1 k j)
    (b2b 1 h k j)
    (union-at-n lp k '(10 11 12)))
    (not (union-at-n lp j '(5 6 7 8 9 10 11 12))))
    ((use (k-in-110-11-or-19))
    (use (case-k-in-110-11))
    (use (case-k-in-19))))

;;;I proved
;;;(prove-lemma lm-b2a-i-eq-k-j-neq-k (rewrite)
;;;  (implies (and (molws n 1 g h)
;;;    (member j (nset n))
;;;    (member k (nset n))
;;;    (mrhoi n k 1 g h lp gp hp)
;;;    (not (equal j k))
;;;    (lessp j k)
;;;    (lg n 1 g)
;;;    (b2a 1 k j)
(b2b 1 h k j)

```

```

(prove-lemma m-k-not-in-lp5-12 (rewrite)
  (implies (and (molws n 1 g h)
    (member i (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (bla 1 i k)
    (union-at-n 1 i '(10 11 12)))
    (not (union-at-n 1 k '(5 6 7 8 9 10 11 12))))
  ((enable bla)
    (use (unl0-12-then-un8-12) (m-k-not-in-14))))

(prove-lemma lm-b2a-i-neq-k-j-eq-k (rewrite)
  (implies (and (molws n 1 g h)
    (member i (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (not (equal i k))
    (lessp k i)
    (bla 1 i k)
    (b2a 1 i k)
    (union-at-n lp i '(10 11 12)))
    (not (union-at-n lp k '(5 6 7 8 9 10 11 12))))
  ((enable b2a) (use (m-k-not-in-lp5-12))))

(prove-lemma b2a-i-neq-k-j-eq-k (rewrite)
  (implies (and (molws n 1 g h)
    (member i (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (not (equal i k))
    (lessp k i)
    (bla 1 i k)
    (b2a 1 i k))
    (b2a lp i k))
  ((enable b2a) (use (lm-b2a-i-neq-k-j-eq-k))))

:* i-j-neq-k-neq-k

(prove-lemma b2a-i-j-neq-k (rewrite)
  (implies (and (molws n 1 g h)
    (member i (nset n))
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (not (equal i k))
    (not (equal j k))
    (lessp j i)
    (b2a 1 i j))
    (b2a lp i j))
  ((enable b2a)))

(prove-lemma b2a-i-neq-k (rewrite)
  (implies (and (molws n 1 g h)
    (member i (nset n))
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (not (equal i k))
    (lessp j i)
    (bla 1 i j)
    (b2a 1 i j)
    (b2b 1 h i j))
    (b2a lp i j))
  ((use (b2a-i-j-neq-k) (b2a-i-neq-k-j-eq-k))))

(prove-lemma b2a-i-eq-k (rewrite)
  (implies (and (molws n 1 g h)
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (lessp j k)
    (lg n 1 g)
    (b2a 1 k j)
    (b2b 1 h k j))
    (b2a lp k j))
  ((use (b2a-i-eq-k-j-neq-k))))

(prove-lemma mrho-preserves-b2a ()
  (implies (and (molws n 1 g h)
    (member i (nset n))
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (lessp j i)
    (lg n 1 g)
    (bla 1 i j)
    (b2a 1 i j)
    (b2b 1 h i j))
    (b2a lp i j))
  ((use (b2a-i-neq-k) (b2a-i-eq-k))))

::: b2b :::
:::Common in atom and mole.

```

```

(prove-lemma l9-then-un8-12 (rewrite)
  (implies (at 1 1 9)
    (union-at-n 1 1 i '(8 9 10 11 12)))
  ((enable at union-at-n)))
:::Common in atom and mole end.

(prove-lemma lg-nth-h-k (rewrite)
  (implies (and (molws n 1 g h)
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (lg n 1 g)
    (at h k j)
    (union-at-n g (nth h k) '(0 1)))
    (not (union-at-n 1 j '(5 6 7 8 9 10 11 12))))
  ((enable at) (use (if1))))

(prove-lemma l9-g01 (rewrite)
  (implies (and (molws n 1 g h)
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (at h k j)
    (at 1 k 9)
    (at lp k 9))
    (union-at-n g (nth h k) '(0 1)))
  ((enable mrhoi at)))

prove-lemma l9-nth-h-k-eq-j (rewrite)
  (implies (and (at h k j)
    (molws n 1 g h)
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (lg n 1 g)
    (at 1 k 9)
    (at lp k 9))
    (not (union-at-n 1 j '(5 6 7 8 9 10 11 12))))
  ((use (l9-g01) (lg-nth-h-k))))

(prove-lemma lm-j-not-in-15-12 (rewrite)
  (implies (and (molws n 1 g h)
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (lessp j k)
    (lg n 1 g)
    (b2b 1 h k j)
    (at 1 k 9)
    (at lp k 9)
    (lessp (sub1 j) (nth h k)))
    (not (union-at-n 1 j '(5 6 7 8 9 10 11 12))))
  ((enable b2b)
    (use (nth-k-lt-j-or-eq-j) (l9-nth-h-k-eq-j))))

(prove-lemma cond-19 (rewrite)
  (implies (and (molws n 1 g h)
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (at 1 k 9)
    (lessp j (nth hp k)))
    (lessp (sub1 j) (nth h k)))
  ((enable mrhoi at)))

(prove-lemma j-not-in-15-12 (rewrite)
  (implies (and (molws n 1 g h)
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (lessp j k)
    (lg n 1 g)
    (b2b 1 h k j)
    (at 1 k 9)
    (at lp k 9)
    (lessp j (nth hp k)))
    (not (union-at-n 1 j '(5 6 7 8 9 10 11 12))))
  ((use (lm-j-not-in-15-12) (cond-19))))

(prove-lemma k-in-19 (rewrite)
  (implies (and (molws n 1 g h)
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (at lp k 9)
    (lessp j (nth hp k)))
    (at 1 k 9))
  ((enable mrhoi at)))

:::The order of the hints is crucial.
(prove-lemma lm-b2b-i-eq-k-j-neq-k (rewrite)
  (implies (and (molws n 1 g h)
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (not (equal j k))
    (lessp j k)
    (lg n 1 g)

```

```

      (b2b 1 h k j)
      (at lp k 9)
      (lessp j (nth hp k)))
      (not (union-at-n lp j '(5 6 7 8 9 10 11 12))))
      ((use (j-not-in-15-12) (k-in-19))))

(prove-lemma b2b-i-eq-k-j-neq-k (rewrite)
  (implies (and (molws n 1 g h)
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (not (equal j k))
    (lessp j k)
    (lg n 1 g)
    (b2b 1 h k j))
    (b2b lp hp k j))
    ((enable b2b) (use (lm-b2b-i-eq-k-j-neq-k)))))

(prove-lemma b2b-i-eq-k (rewrite)
  (implies (and (molws n 1 g h)
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (lessp j k)
    (lg n 1 g)
    (b2b 1 h k j))
    (b2b lp hp k j))
    ((use (b2b-i-eq-k-j-neq-k)))))

(prove-lemma not-k-in-15-12-imp (rewrite)
  (implies (and (molws n 1 g h)
    (member i (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (bla 1 i k)
    (at 1 i 9)
    (not (union-at-n 1 k '(5 6 7 8 9 10 11 12))))
    (not (union-at-n lp k '(5 6 7 8 9 10 11 12))))
    ((enable bla)
      (use (19-then-un8-12) (m-k-not-in-14)))))

;;;The order of hypotheses is crucial.
(prove-lemma lm-b2b-i-neq-k-j-eq-k (rewrite)
  (implies (and (molws n 1 g h)
    (member i (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (not (equal i k))
    (lessp k i)
    (bla 1 i k)
    (b2b 1 h i k)
    (at 1 i 9)
    (lessp k (nth h i)))
    (not (union-at-n lp k '(5 6 7 8 9 10 11 12))))
    ((enable b2b) (use (not-k-in-15-12-imp)))))

(prove-lemma b2b-i-neq-k-j-eq-k (rewrite)
  (implies (and (molws n 1 g h)
    (member i (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (not (equal i k))
    (lessp k i)
    (bla 1 i k)
    (b2b 1 h i k))
    (b2b lp hp i k))
    ((enable b2b) (use (lm-b2b-i-neq-k-j-eq-k)))))

;;;The position of (member k (nset n)) is
;;;crucial to trigger rhollemmas.
(prove-lemma b2b-i-j-neq-k (rewrite)
  (implies (and (molws n 1 g h)
    (member i (nset n))
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (not (equal i k))
    (not (equal j k))
    (lessp j i)
    (b2b 1 h i j))
    (b2b lp hp i j))
    ((enable b2b)))))

(prove-lemma b2b-i-neq-k (rewrite)
  (implies (and (molws n 1 g h)
    (member i (nset n))
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (not (equal i k))
    (lessp j i)
    (bla 1 i j)
    (b2b 1 h i j))
    (b2b lp hp i j))
    ((use (b2b-i-j-neq-k) (b2b-i-neq-k-j-eq-k)))))

(prove-lemma mrho-preserves-b2b ()
  (implies (and (molws n 1 g h)
    (member i (nset n))
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (not (equal i k))
    (lessp j i)
    (bla 1 i j)
    (b2b 1 h i j))
    (b2b lp hp i j))
    ((use (b2b-i-j-neq-k) (b2b-i-neq-k-j-eq-k)))))

```

```

***** b3a *****

; * i-neq-k-j-eq-k

(prove-lemma lm-b3a-k-in-19-11 (rewrite)
  (implies (and (molws n 1 g h)
    (member i (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (lg n 1 g)
    (b3a 1 g i k)
    (at 1 i 12)
    (union-at-n 1 k '(5 6 7 8 9 10 11)))
    (union-at-n 1 k '(9 10 11)))
    ((enable b3a)
      (use (un5-11-then-un5-12) (k-in-15-11-g4-then-19-11))))

(prove-lemma b3a-k-in-19-11 (rewrite)
  (implies (and (molws n 1 g h)
    (member i (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (lg n 1 g)
    (bla 1 i k)
    (b3a 1 g i k)
    (at 1 i 12)
    (union-at-n lp k '(5 6 7 8 9 10 11 12)))
    (union-at-n 1 k '(9 10 11)))
    ((enable b1a)
      (use (lm-b3a-k-in-19-11) (l12-then-un8-12) (m-k-in-15-11))))

(prove-lemma m-k-in-lp9-12 (rewrite)
  (implies (and (molws n 1 g h)
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (union-at-n 1 k '(9 10 11)))
    (union-at-n lp k '(9 10 11 12)))
    ((enable union-at-n at mrhoi)))

(prove-lemma lm-b3a-i-neq-k-j-eq-k (rewrite)
  (implies (and (molws n 1 g h)
    (member i (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (lg n 1 g)
    (lg n lp gp)
    (bla 1 i k)
    (b3a 1 g i k)
    (at 1 i 12)
    (union-at-n lp k '(5 6 7 8 9 10 11 12)))
    (at gp k 4))
    ((disable mrho-preserves-lg)
      (use (b3a-k-in-19-11) (m-k-in-lp9-12))
      (use (if4 (j k) (1 lp) (g gp))))))

(prove-lemma b3a-i-neq-k-j-eq-k (rewrite)
  (implies (and (molws n 1 g h)
    (member i (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (not (equal i k))
    (lg n 1 g)
    (bla 1 i k)
    (b3a 1 g i k)
    (b3a lp gp i k))
    ((enable b3a)
      (use (lm-b3a-i-neq-k-j-eq-k))
      (use (mrho-preserves-lg))))

; * i-eq-k-j-neq-k
(prove-lemma cond-lp12 (rewrite)
  (implies (and (molws n 1 g h)
    (member k (nset n))
    (member j (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (at lp k 12)
    (at 1 k 11))
    (lessp j (nth h k)))
    ((enable mrhoi at)))

(prove-lemma b3a-j-in-15-12 (rewrite)
  (implies (and (molws n 1 g h)
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (lg n 1 g)
    (b3b 1 g h k j)
    (at 1 k 11)
    (at lp k 12)
    (union-at-n 1 j '(5 6 7 8 9 10 11 12)))
    (at g j 4))
    ((enable b3b) (use (cond-lp12))))

(prove-lemma m-k-in-111 (rewrite)
  (implies (and (molws n 1 g h)
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (at lp k 12))

***** b3b *****

(at 1 k 11))
((enable mrhoi at)))

(prove-lemma lm-b3a-i-eq-k-j-neq-k (rewrite)
  (implies (and (molws n 1 g h)
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (not (equal j k))
    (lg n 1 g)
    (b3b 1 g h k j)
    (at lp k 12)
    (union-at-n lp j '(5 6 7 8 9 10 11 12)))
    (at g j 4))
    ((use (b3a-j-in-15-12) (m-k-in-111))))

(prove-lemma b3a-i-eq-k-j-neq-k (rewrite)
  (implies (and (molws n 1 g h)
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (not (equal j k))
    (lg n 1 g)
    (b3b 1 g h k j))
    (b3a lp gp k j))
    ((enable b3a) (use (lm-b3a-i-eq-k-j-neq-k))))

; * i-j-neq-k
(prove-lemma b3a-i-j-neq-k (rewrite)
  (implies (and (molws n 1 g h)
    (member i (nset n))
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (not (equal i k))
    (not (equal j k))
    (b3a 1 g i j))
    (b3a lp gp i j))
    ((enable b3a)))

(prove-lemma b3a-i-neq-k (rewrite)
  (implies (and (molws n 1 g h)
    (member i (nset n))
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (not (equal i k))
    (lg n 1 g)
    (bla 1 i j)
    (b3a 1 g i j))
    (b3a lp gp i j))
    ((use (b3a-i-j-neq-k) (b3a-i-neq-k-j-eq-k))))

; * i-j-eq-k
(prove-lemma b3a-i-j-eq-k (rewrite)
  (implies (and (molws n 1 g h)
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (lg n 1 g)
    (b3a 1 g k k)
    (b3b 1 g h k k))
    (b3a lp gp k k))
    ((enable b3a)
      (use (if4 (j k) (1 lp) (g gp)))
      (use (l12-then-un9-12))))

(prove-lemma b3a-i-eq-k (rewrite)
  (implies (and (molws n 1 g h)
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (lg n 1 g)
    (b3a 1 g k j)
    (b3b 1 g h k j))
    (b3a lp gp k j))
    ((use (b3a-i-eq-k-j-neq-k) (b3a-i-j-eq-k))))

(prove-lemma mrho-preserves-b3a {}
  (implies (and (molws n 1 g h)
    (member i (nset n))
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (lg n 1 g)
    (bla 1 i j)
    (b3a 1 g i j)
    (b3b 1 g h i j))
    (b3a lp gp i j))
    ((use (b3a-i-neq-k) (b3a-i-eq-k))))

***** b3b *****

*****common in atom and mole.
(prove-lemma l10-then-un10-12 (rewrite)
  (implies (at 1 k 10)

```

```

(union-at-n 1 k '(10 11 12)))
((enable at union-at-n)))

(prove-lemma 111-then-un9-12 (rewrite)
  (implies (at lp k 11)
    (union-at-n lp k '(9 10 11 12)))
  ((enable union-at-n at)))

(prove-lemma 111-then-un8-12 (rewrite)
  (implies (at 1 i 11)
    (union-at-n 1 i '(8 9 10 11 12)))
  ((enable union-at-n at)))
:::common in atom and mole end.

:* i-neq-k-j-eq-k

(prove-lemma 1m-b3b-k-in-19-11 (rewrite)
  (implies (and (molws n 1 g h)
    (member i (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (lg n 1 g)
    (b3b 1 g h i k)
    (at 1 i 11)
    (lessp k (nth h i))
    (union-at-n 1 k '(5 6 7 8 9 10 11)))
    (union-at-n 1 k '(9 10 11)))
  ((enable b3b)
    (use (un5-11-then-un5-12))
    (use (k-in-15-11-g4-then-19-11))))

(prove-lemma b3b-k-in-19-11 (rewrite)
  (implies (and (molws n 1 g h)
    (member i (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (bla 1 i k)
    (b3b 1 g h i k)
    (lg n 1 g)
    (at 1 i 11)
    (lessp k (nth h i))
    (union-at-n lp k '(5 6 7 8 9 10 11 12)))
    (union-at-n 1 k '(9 10 11)))
  ((enable bla)
    (use (1m-b3b-k-in-19-11) (111-then-un8-12) (m-k-in-15-11))))

(prove-lemma 1m-b3b-i-neq-k-j-eq-k (rewrite)
  (implies (and (molws n 1 g h)
    (member i (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (lg n 1 g)
    (lg n lp gp)
    (bla 1 i k)
    (b3b 1 g h i k)
    (at 1 i 11)
    (lessp k (nth h i))
    (union-at-n lp k '(5 6 7 8 9 10 11 12)))
    (at gp k 4))
  ((disable mrho-preserves-lg)
    (use (b3b-k-in-19-11) (m-k-in-19-12))
    (use (if4 (j k) (1 lp) (g gp))))))

(prove-lemma b3b-i-neq-k-j-eq-k (rewrite)
  (implies (and (molws n 1 g h)
    (member i (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (not (equal i k))
    (lg n 1 g)
    (bla 1 i k)
    (b3b 1 g h i k)
    (b3b lp gp hp i k))
    (enable b3b)
    (use (1m-b3b-i-neq-k-j-eq-k))
    (use (mrho-preserves-lg))))

:* i-eq-k-j-neq-k

(prove-lemma j-in-g4 (rewrite)
  (implies (and (molws n 1 g h)
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (lg n 1 g)
    (at h k j)
    (not (union-at-n g (nth h k) '(2 3)))
    (union-at-n 1 j '(5 6 7 8 9 10 11 12)))
    (at g j 4))
  ((enable at)
    (use (if4) (15-12-eq-15-8-or-19-12) (if3))))

(prove-lemma 111-g14 (rewrite)
  (implies (and (molws n 1 g h)
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (at h k j)
    (at 1 k 11)
    (at lp k 11)
    (union-at-n 1 j '(5 6 7 8 9 10 11 12)))
    (at g j 4))
  ((use (111-g14) (j-in-g4))))

(prove-lemma 1m-j-in-15-12 (rewrite)
  (implies (and (molws n 1 g h)
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (lg n 1 g)
    (b3b 1 g h k j)
    (at 1 k 11)
    (at lp k 11)
    (lessp (sub1 j) (nth h k))
    (union-at-n 1 j '(5 6 7 8 9 10 11 12)))
    (at g j 4))
  ((enable b3b)
    (use (nth-k-1t-j-or-eq-j))
    (use (111-nth-h-k-eq-j))))

(prove-lemma cond-111 (rewrite)
  (implies (and (molws n 1 g h)
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (at 1 k 11)
    (lessp j (nth hp k))
    (lessp (sub1 j) (nth h k)))
    (enable mrhoi at)))

(prove-lemma j-in-15-12 (rewrite)
  (implies (and (molws n 1 g h)
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (not (equal j k))
    (lg n 1 g)
    (b3b 1 g h k j)
    (at 1 k 11)
    (at lp k 11)
    (lessp j (nth hp k))
    (union-at-n 1 j '(5 6 7 8 9 10 11 12)))
    (at g j 4))
  ((use (1m-j-in-15-12) (cond-111))))

(prove-lemma j-leq-addlk-then-k-not-in-110 (rewrite)
  (implies (and (molws n 1 g h)
    (member k (nset n))
    (member j (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (b2a 1 k j)
    (union-at-n 1 j '(5 6 7 8 9 10 11 12))
    (not (equal j k))
    (lessp j (addl k))
    (not (at 1 k 10)))
    (enable b2a)
    (use (110-then-un10-12))))

(prove-lemma not-j-leq-addlk-then-k-not-in-110 (rewrite)
  (implies (and (molws n 1 g h)
    (member k (nset n))
    (member j (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (at lp k 11)
    (lessp j (nth hp k))
    (not (lessp j (addl k))))
    (not (at 1 k 10)))
  ((enable mrhoi at)))

:::The order of (member k (nset n)) and
::: (member j (nset n)) are switched deliberately.
(prove-lemma k-not-in-110 (rewrite)
  (implies (and (molws n 1 g h)
    (member k (nset n))
    (member j (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (not (equal j k))
    (b2a 1 k j)
    (union-at-n 1 j '(5 6 7 8 9 10 11 12))
    (at lp k 11)
    (lessp j (nth hp k)))
    (not (at 1 k 10)))
  ((use (not-j-leq-addlk-then-k-not-in-110))
    (use (j-leq-addlk-then-k-not-in-110))))

```

```

(prove-lemma lp11-then-111-or-110 (rewrite)
  (implies (and (molws n 1 g h)
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (at lp k 11)
    (not (at 1 k 10)))
    (at 1 k 11))
  ((enable mrhoi at)))

;;When the order of (member j (nset n)) and
;;(member k (nset n)) is switched, the order of
;;hints must be switched, in order to make the proof
;;successful.
(prove-lemma b3b-k-in-111 (rewrite)
  (implies (and (molws n 1 g h)
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (not (equal j k))
    (b2a 1 k j)
    (union-at-n 1 j '(5 6 7 8 9 10 11 12))
    (at lp k 11)
    (lessp j (nth hp k)))
    (at 1 k 11))
  ((use (k-not-in-110))
   (use (lp11-then-111-or-110))))

;;When the order of (member j (nset n)) and
;;(member k (nset n)) is switched then the order of
;;hints must be switched in order to make the proof
;;successful.
(prove-lemma lm-b3b-i-eq-k-j-neq-k (rewrite)
  (implies (and (molws n 1 g h)
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (not (equal j k))
    (lg n 1 g)
    (b2a 1 k j)
    (b3b 1 g h k j)
    (at lp k 11)
    (lessp j (nth hp k))
    (union-at-n 1 j '(5 6 7 8 9 10 11 12)))
    (at g j 4))
  ((use (b3b-k-in-111) (j-in-15-12))))

(prove-lemma b3b-i-eq-k-j-neq-k (rewrite)
  (implies (and (molws n 1 g h)
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (not (equal j k))
    (lg n 1 g)
    (b3b 1 g h k j)
    (b2a 1 k j))
    (b3b lp gp hp k j))
  ((enable b3b) (use (lm-b3b-i-eq-k-j-neq-k))))

(prove-lemma b3b-i-j-neq-k (rewrite)
  (implies (and (molws n 1 g h)
    (member i (nset n))
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (not (equal i k))
    (not (equal j k))
    (b3b 1 g h i j))
    (b3b lp gp hp i j))
  ((enable b3b)))

(prove-lemma b3b-i-neq-k (rewrite)
  (implies (and (molws n 1 g h)
    (member i (nset n))
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (not (equal i k))
    (lg n 1 g)
    (b1a 1 i j)
    (b3b 1 g h i j))
    (b3b lp gp hp i j))
  ((use (b3b-i-j-neq-k) (b3b-i-neq-k-j-eq-k))))

(prove-lemma b3b-i-j-eq-k (rewrite)
  (implies (and (molws n 1 g h)
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (lg n 1 g)
    (b3b 1 g h k k))
    (b3b lp gp hp k k))
  ((enable b3b)
   (use (if4 (j k) (1 lp) (g gp)) (I11-then-un9-12))))

(prove-lemma b3b-i-eq-k (rewrite)
  (implies (and (molws n 1 g h)
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)

```

```

(lg n 1 g)
(b1a 1 k j)
(b3b 1 g h k j)
(b2a 1 k j))
(b3b lp gp hp k j))
((use (b3b-i-eq-k-j-neq-k) (b3b-i-j-eq-k))))

(prove-lemma mrho-preserves-b3b ()
  (implies (and (molws n 1 g h)
    (member i (nset n))
    (member j (nset n))
    (member k (nset n))
    (mrhoi n k 1 g h lp gp hp)
    (lg n 1 g)
    (b1a 1 i j)
    (b3b 1 g h i j)
    (b2a 1 i j))
    (b3b lp gp hp i j))
  ((use (b3b-i-neq-k) (b3b-i-eq-k))))

```