```
;; Koenig's tree lemma.

;; We use the formulation that says that any finitely
;; branching tree which is infinite has an infinite branch.

;; More precisely, we consider trees whose nodes are finite sequences
;; of natural numbers, such that each node's successors are a list
;; obtained by tacking on some natural number from 1 to the number of
;; successors.  For convenience, I'll keep each node as a list of
;; numbers where the root is the last element in the list.
```

EVENT: Start with the initial **nqthm** theory.

```
;; All this initial stuff is just to get the CONSTRAIN below accepted.
```

DEFINITION:
ones (n)
= **if** $n \simeq 0$ **then nil**
else cons (1, ones ($n - 1$)) **endif**

DEFINITION:
all-ones (s)
= **if** listp (s) **then** (car (s) = 1) $\wedge$ all-ones (cdr (s))
else s = **nil endif**

DEFINITION:
length (s)
= **if** listp (s) **then** $1 +$ length (cdr (s))
else 0 **endif**

DEFINITION:
subseq ($s1$, $s2$)
= **if** $s1 = s2$ **then t**
elseif $s2 \simeq$ **nil then f**
else subseq ($s1$, cdr ($s2$)) **endif**

THEOREM: subseq-all-ones
(all-ones ($s1$) $\wedge$ subseq ($s2$, $s1$)) $\rightarrow$ all-ones ($s2$)

DEFINITION:
plistp (s)
= **if** listp (s) **then** plistp (cdr (s))
else s = **nil endif**

THEOREM: plistp-all-ones
all-ones $(s) \rightarrow$ plistp (s)

THEOREM: all-ones-ones
all-ones (ones (n))

THEOREM: ones-is-injective
$((i \in \mathbf{N}) \land (j \in \mathbf{N}) \land (i \neq j)) \rightarrow (\text{ones}(i) \neq \text{ones}(j))$

CONSERVATIVE AXIOM: koenig-intro
node-p (**nil**)
$\land$ (truep (node-p (s)) $\lor$ falsep (node-p (s)))
$\land$ (node-p (s)
$\rightarrow$ (node-p (cons (n, s)) $= ((0 < n) \land (\text{succard}(s) \not< n))$)))
$\land$ ((node-p (*s1*) $\land$ subseq $(s, s1)$) $\rightarrow$ node-p (s))
$\land$ node-p (s-n (n))
$\land$ $(((i \in \mathbf{N}) \land (j \in \mathbf{N}) \land (i \neq j)) \rightarrow (\text{s-n}(i) \neq \text{s-n}(j)))$
$\land$ (($\neg$ plistp (s)) $\rightarrow$ ($\neg$ node-p (s)))

Simultaneously, we introduce the new function symbols *node-p*, *succard*, and *s-n*.

```
;; We want to define a function s-height which returns an element of a given height.
;; The next several events culminate in the following lemma:

;; (prove-lemma length-s-height (rewrite)
;;   (equal (length (s-height n)) (fix n)))
```

DEFINITION:
succ-aux (s, n)
= **if** $n \simeq 0$ **then nil**
else cons (cons (n, s), succ-aux $(s, n - 1)$) **endif**

DEFINITION: successors (s) = succ-aux $(s$, succard $(s))$

DEFINITION:
successors-list (ss)
= **if** listp (ss)
then append (successors (car (ss)), successors-list (cdr (ss)))
else nil endif

DEFINITION:
level (n)
= **if** $n \simeq 0$ **then** list (**nil**)
else successors-list (level $(n - 1)$) **endif**

Definition:
init-tree (n)
$=$ **if** $n \simeq 0$ **then** list (**nil**)
else append (level (n), init-tree $(n - 1)$) **endif**

Definition:
remove1 $(a,\ x)$
$=$ **if** listp (x)
then if $a =$ car (x) **then** cdr (x)
else cons (car (x), remove1 $(a,$ cdr $(x)))$ **endif**
else x **endif**

Theorem: length-remove1
$(a \in x) \rightarrow$ (length (remove1 $(a,\ x)) <$ length $(x))$

Definition:
first-non-member-index $(i,\ x)$
$=$ **if** s-n $(i) \in x$ **then** first-non-member-index $(1 + i,$ remove1 (s-n $(i),\ x))$
else i **endif**

Definition:
nthcdr $(n,\ s)$
$=$ **if** $n \simeq 0$ **then** s
else nthcdr $(n - 1,$ cdr $(s))$ **endif**

Definition:
s-height (n)
$=$ nthcdr (length (s-n (first-non-member-index (0, init-tree (n)))) $- n$,
s-n (first-non-member-index (0, init-tree (n))))

Theorem: nthcdr-subseq
(length $(s) \not< n) \rightarrow$ subseq (nthcdr $(n,\ s),\ s)$

Theorem: node-p-nthcdr
(node-p $(s) \wedge$ (length $(s) \not< n)) \rightarrow$ node-p (nthcdr $(n,\ s)$)

Theorem: lessp-difference-1
$(x < (x - n)) =$ **f**

Theorem: node-p-s-height
node-p (s-height (n))

Theorem: length-nthcdr
length (nthcdr $(n,\ s)) =$ (length $(s) - n)$

Theorem: first-non-member-index-lessp
first-non-member-index $(i,\ x) \not< i$

THEOREM: s-n-first-non-member-index-not-equal
$(i \in \mathbf{N})$
$\rightarrow \quad (\text{s-n}\,(\text{first-non-member-index}\,(1 + i,\ \text{remove1}\,(\text{s-n}\,(i),\ x))) \neq \text{s-n}\,(i))$

THEOREM: member-remove1
$(a \neq b) \rightarrow ((a \in \text{remove1}\,(b,\ x)) = (a \in x))$

THEOREM: s-n-first-non-member-index
$(i \in \mathbf{N}) \rightarrow (\text{s-n}\,(\text{first-non-member-index}\,(i,\ x)) \notin x)$

THEOREM: member-append
$(a \in \text{append}\,(x,\ y)) = ((a \in x) \vee (a \in y))$

THEOREM: member-cons-succ-aux
$(\text{cons}\,(z,\ v) \in \text{succ-aux}\,(v,\ n)) = ((0 < z) \wedge (n \not< z))$

THEOREM: node-p-cons-lemma
$(\neg\ \text{node-p}\,(s)) \rightarrow (\neg\ \text{node-p}\,(\text{cons}\,(n,\ s)))$

THEOREM: node-p-cons
$\text{node-p}\,(\text{cons}\,(n,\ s)) = (\text{node-p}\,(s) \wedge (0 < n) \wedge (\text{succard}\,(s) \not< n))$

DEFINITION:
$\text{all-length-n}\,(ss,\ n)$
$= \quad$ **if** $\text{listp}\,(ss)$ **then** $(\text{length}\,(\text{car}\,(ss)) = n)$
$\wedge \quad \text{all-length-n}\,(\text{cdr}\,(ss),\ n)$
else t endif

THEOREM: all-length-n-append
$\text{all-length-n}\,(\text{append}\,(ss1,\ ss2),\ n)$
$= \quad (\text{all-length-n}\,(ss1,\ n) \wedge \text{all-length-n}\,(ss2,\ n))$

THEOREM: all-length-n-succ-aux
$(\text{length}\,(s) = n) \rightarrow \text{all-length-n}\,(\text{succ-aux}\,(s,\ k),\ 1 + n)$

THEOREM: all-length-n-successors-list
$\text{all-length-n}\,(ss,\ n) \rightarrow \text{all-length-n}\,(\text{successors-list}\,(ss),\ 1 + n)$

THEOREM: length-0
$(\text{length}\,(s) = 0) = (s \simeq \mathbf{nil})$

DEFINITION:
$\text{member-level-induction}\,(s,\ n)$
$= \quad$ **if** $n \simeq 0$ **then t**
else $\text{member-level-induction}\,(\text{cdr}\,(s),\ n - 1)$ **endif**

THEOREM: succ-aux-listp
$(\neg \text{listp}(s)) \rightarrow (s \notin \text{succ-aux}(z, n))$

THEOREM: successors-list-listp
$(\neg \text{listp}(s)) \rightarrow (s \notin \text{successors-list}(ss))$

THEOREM: member-succ-aux
$(s \in \text{succ-aux}(x, n)) \rightarrow (\text{cdr}(s) = x)$

THEOREM: member-successors-list-successors-list-witness
$(s \in \text{successors-list}(ss))$
$= \quad ((\text{cdr}(s) \in ss) \wedge (s \in \text{successors}(\text{cdr}(s))))$

THEOREM: member-level
$((n \in \mathbf{N}) \wedge \text{node-p}(s)) \rightarrow ((s \in \text{level}(n)) = (\text{length}(s) = n))$

THEOREM: member-init-tree
$\text{node-p}(s) \rightarrow ((s \in \text{init-tree}(n)) = (n \not< \text{length}(s)))$

THEOREM: length-s-non-member-index
$(i \in \mathbf{N}) \rightarrow (n < \text{length}(\text{s-n}(\text{first-non-member-index}(i, \text{init-tree}(n)))))$

THEOREM: length-s-height
$\text{length}(\text{s-height}(n)) = \text{fix}(n)$

EVENT: Disable s-height.

```
;; End of s-height excursion.

;; Our goal:
#|
(prove-lemma konig-tree-lemma nil
  (and (node-p (k n))
       (implies (not (lessp j i))
(subseq (k i) (k j)))
       (equal (length (k n)) (fix n))))
|#
```

DEFINITION:
$\text{inf}(s)$
$\leftrightarrow \quad \forall\, \textit{big-h}\ \exists\, \textit{big-s}\ (\text{subseq}(s, \textit{big-s})$
$\wedge \quad \text{node-p}(\textit{big-s})$
$\wedge \quad (\textit{big-h} < \text{length}(\textit{big-s})))$

```
;; The following three events were generated mechanically.  They are
;; useful especially for applying the Skolem axioms for INF inside the
;; proof-checker, via the macro command SK*.
```

EVENT: Disable inf.

THEOREM: inf-suff
(subseq (s, *big-s*) $\wedge$ node-p (*big-s*) $\wedge$ (big-h (s) $<$ length (*big-s*)))
$\rightarrow$ inf (s)

THEOREM: inf-necc
($\neg$ (subseq (s, big-s (*big-h*, s))
$\wedge$ node-p (big-s (*big-h*, s))
$\wedge$ (*big-h* $<$ length (big-s (*big-h*, s)))))
$\rightarrow$ ($\neg$ inf (s))

DEFINITION:
next (s, *max*)
= **if** *max* $\simeq$ 0 **then** cons (0, s)
elseif inf (cons (*max*, s)) **then** cons (*max*, s)
else next (s, *max* $-$ 1) **endif**

```
;; We want to show that NEXT gives us a successor with infinitely many
;; successors.

#| INF-IMPLIES-INF-NEXT:
(implies (and (node-p s)
      (inf s))
 (inf (next s (succard s))))
|#

;; Note that if some successor of s has infinitely many successors, so
;; does (NEXT S (SUCCARD S)).  This is the lemma
;; INF-CONS-IMPLIES-INF-NEXT below.  But first note:
```

THEOREM: inf-implies-node-p
inf (s) $\rightarrow$ node-p (s)

THEOREM: not-inf-zerop
($i \simeq 0$) $\rightarrow$ ($\neg$ inf (cons (i, s)))

THEOREM: inf-cons-implies-inf-next
(node-p (s) $\wedge$ inf (cons (i, s)) $\wedge$ ($n \not< i$)) $\rightarrow$ inf (next (s, n))

```
;; Our goal now is to apply this lemma by proving that
;; (inf (cons i s)) for some i <= (succard s).
```

DEFINITION:
all-big-h $(s,\ n)$
$=$ **if** $n \simeq 0$ **then** $1 + \text{length}(s)$
else big-h (cons $(n,\ s)$) + all-big-h $(s,\ n - 1)$ **endif**

THEOREM: all-big-h-length
length (s) < all-big-h $(s,\ n)$

THEOREM: all-big-h-lessp
$((0 < i) \wedge (n \not< i))$
$\rightarrow$ ((big-h (cons $(i,\ s)$) < all-big-h $(s,\ n)$) = **t**)

```
;; Here's a function which tells us which way s first branches on its
;; way to extending to s1.
```

DEFINITION:
first-branch $(s,\ s1)$
$=$ **if** $s = \text{cdr}(s1)$ **then** car $(s1)$
elseif $s1 \simeq$ **nil** **then** 0
else first-branch $(s,\ \text{cdr}(s1))$ **endif**

THEOREM: subseq-cons-first-branch
(subseq $(s,\ x) \wedge (s \neq x)$) $\rightarrow$ subseq (cons (first-branch $(s,\ x)$, s), x)

THEOREM: length-non-equal
(length (x) < length (y)) $\rightarrow$ $((x = y) = \mathbf{f})$

THEOREM: first-branch-ok-for-succard
(subseq $(s,\ big\text{-}s)$ $\wedge$ node-p $(big\text{-}s)$ $\wedge$ $(s \neq big\text{-}s)$)
$\rightarrow$ ((first-branch $(s,\ big\text{-}s) \in \mathbf{N}$)
$\wedge$ (0 < first-branch $(s,\ big\text{-}s)$))
$\wedge$ (succard $(s) \not<$ first-branch $(s,\ big\text{-}s)$)))

THEOREM: all-big-h-lessp-linear
$((0 < i) \wedge$ (succard $(s) \not< i$))
$\rightarrow$ (big-h (cons $(i,\ s)$) < all-big-h $(s$, succard (s)))

EVENT: Disable all-big-h-lessp.

THEOREM: inf-implies-inf-next
(node-p (s) $\wedge$ inf (s)) $\rightarrow$ inf (next $(s$, succard (s)))

DEFINITION:
$k(n)$
$= \quad$ **if** $n \simeq 0$ **then nil**
else next (k $(n-1)$, succard (k $(n-1)$)) **endif**

THEOREM: subseq-nil
subseq (**nil**, x) = plistp (x)

THEOREM: node-p-implies-plistp
node-p (s) $\rightarrow$ plistp (s)

THEOREM: inf-nil
inf (**nil**)

EVENT: Disable node-p-implies-plistp.

THEOREM: konig-tree-lemma-1
inf (k (n))

THEOREM: length-next
inf (x) $\rightarrow$ (length (next (s, n)) = (1 + length (s)))

THEOREM: konig-tree-lemma-2
length (k (n)) = fix (n)

THEOREM: subseq-next
subseq ($s1$, $s2$) $\rightarrow$ subseq ($s1$, next ($s2$, n))

THEOREM: konig-tree-lemma-3
($j \not< i$) $\rightarrow$ subseq (k (i), k (j))

THEOREM: konig-tree-lemma
node-p (k (n))
$\wedge$ (($j \not< i$) $\rightarrow$ subseq (k (i), k (j)))
$\wedge$ (length (k (n)) = fix (n))

```
;; or, if one prefers:
```

THEOREM: konig-tree-lemma-again
($n \in \mathbf{N}$)
$\rightarrow$ (node-p (k (n))
$\wedge$ (($j \not< i$) $\rightarrow$ subseq (k (i), k (j)))
$\wedge$ (length (k (n)) = n))

Index