EVENT: Start with the library `"subset"`.

DEFINITION:
newpairs (*pair*, *pairs*)
= **if** ¬ listp (*pairs*) **then nil**
elseif cdr (*pair*) = caar (*pairs*)
then cons (cons (car (*pair*), cdar (*pairs*)), newpairs (*pair*, cdr (*pairs*)))
else newpairs (*pair*, cdr (*pairs*)) **endif**

DEFINITION:
all-newpairs (*pairs1*, *pairs2*)
= **if** ¬ listp (*pairs1*) **then nil**
else append (newpairs (car (*pairs1*), *pairs2*),
all-newpairs (cdr (*pairs1*), *pairs2*)) **endif**

DEFINITION:
all-cars (*pairs*)
= **if** listp (*pairs*) **then** cons (caar (*pairs*), all-cars (cdr (*pairs*)))
else nil endif

DEFINITION:
all-cdrs (*pairs*)
= **if** listp (*pairs*) **then** cons (cdar (*pairs*), all-cdrs (cdr (*pairs*)))
else nil endif

DEFINITION:
prod1 (*a*, *l*)
= **if** listp (*l*) **then** cons (cons (*a*, car (*l*)), prod1 (*a*, cdr (*l*)))
else nil endif

DEFINITION:
prod (*l*, *m*)
= **if** listp (*l*) **then** append (prod1 (car (*l*), *m*), prod (cdr (*l*), *m*))
else nil endif

DEFINITION:
field (*pairs*) = prod (all-cars (*pairs*), all-cdrs (*pairs*))

DEFINITION:
all-pairsp (*pairs*)
= **if** ¬ listp (*pairs*) **then t**
else listp (car (*pairs*)) ∧ all-pairsp (cdr (*pairs*)) **endif**

THEOREM: newpairs-prod1-subsetp
subsetp (newpairs (*pair*, *pairs*), prod1 (car (*pair*), all-cdrs (*pairs*)))

THEOREM: all-newpairs-field-subsetp-1
subsetp (all-newpairs (*pairs1*, *pairs2*), prod (all-cars (*pairs1*), all-cdrs (*pairs2*)))

THEOREM: all-newpairs-field-subsetp
subsetp (all-newpairs (*pairs*, *pairs*), field (*pairs*))

DEFINITION:
close1 (*pairs*) = (make-set (all-newpairs (*pairs*, *pairs*)) $\cup$ *pairs*)

```
; My goal is to show that
; (SUBSETP PAIRS (FIELD PAIRS))
; and
; (IMPLIES (SUBSETP X (FIELD PAIRS))
;          (SUBSETP (CLOSE1 X) (FIELD PAIRS))) .
; For then we can prove the following lemma, which will
; show that the following definition should
; be accepted, where CARD is cardinality:

;(IMPLIES (AND (ALL-PAIRSP PAIRS)
;       (NOT (EQUAL PAIRS (CLOSE1 PAIRS))))
;  (EQUAL (LESSP (DIFFERENCE (CARD (FIELD (CLOSE1 PAIRS)))
;    (CARD (CLOSE1 PAIRS)))
;        (DIFFERENCE (CARD (FIELD PAIRS))
;    (CARD PAIRS)))
; T))
```

THEOREM: cdr-all-cars-subsetp
subsetp (all-cars (cdr (*pairs*)), all-cars (*pairs*))

THEOREM: cdr-all-cdrs-subsetp
subsetp (all-cdrs (cdr (*pairs*)), all-cdrs (*pairs*))

THEOREM: prod-subsetp-1
subsetp (x, y) $\rightarrow$ subsetp (prod (x, z), prod (y, z))

THEOREM: prod1-subsetp-2
subsetp (y, z) $\rightarrow$ subsetp (prod1 (a, y), prod1 (a, z))

THEOREM: prod-subsetp-2
subsetp (y, z) $\rightarrow$ subsetp (prod (x, y), prod (x, z))

THEOREM: field-subsetp
all-pairsp (*pairs*) $\rightarrow$ subsetp (*pairs*, field (*pairs*))

THEOREM: member-all-cars
($a \in y$) $\rightarrow$ (car (a) $\in$ all-cars (y))

THEOREM: subsetp-all-cars
$\text{subsetp}(x,\ y) \rightarrow \text{subsetp}(\text{all-cars}(x),\ \text{all-cars}(y))$

THEOREM: member-all-cdrs
$(a \in y) \rightarrow (\text{cdr}(a) \in \text{all-cdrs}(y))$

THEOREM: subsetp-all-cdrs
$\text{subsetp}(x,\ y) \rightarrow \text{subsetp}(\text{all-cdrs}(x),\ \text{all-cdrs}(y))$

THEOREM: field-preserves-subsetp
$\text{subsetp}(x,\ y) \rightarrow \text{subsetp}(\text{field}(x),\ \text{field}(y))$

THEOREM: subsetp-all-cars-prod
$\text{subsetp}(\text{all-cars}(\text{prod}(x,\ y)),\ x)$

THEOREM: subsetp-all-cdrs-prod
$\text{subsetp}(\text{all-cdrs}(\text{prod}(x,\ y)),\ y)$

THEOREM: prod-subsetp
$(\text{subsetp}(x1,\ x2) \land \text{subsetp}(y1,\ y2)) \rightarrow \text{subsetp}(\text{prod}(x1,\ y1),\ \text{prod}(x2,\ y2))$

THEOREM: subsetp-field-twice
$\text{subsetp}(\text{field}(\text{field}(l)),\ \text{field}(l))$

THEOREM: subsetp-close1-field
$\text{subsetp}(x,\ \text{field}(pairs)) \rightarrow \text{subsetp}(\text{close1}(x),\ \text{field}(pairs))$

DEFINITION:
card (x)
= **if** listp (x)
then **if** car $(x) \in$ cdr (x) **then** card (cdr (x))
else $1 +$ card (cdr (x)) **endif**
else 0 **endif**

THEOREM: card-not-0
$(x \in y) \rightarrow (\text{card}(y) \neq 0)$

DEFINITION:
remove $(x,\ l)$
= **if** listp (l)
then **if** $x =$ car (l) **then** remove $(x,$ cdr $(l))$
else cons (car (l), remove $(x,$ cdr $(l)))$ **endif**
else l **endif**

THEOREM: remove-non-member
$(a \notin l) \rightarrow (\text{remove}(a,\ l) = l)$

THEOREM: remove-preserves-non-member
$(a \notin l) \rightarrow (a \notin \text{remove}(x, l))$

THEOREM: remove-subsetp
$\text{subsetp}(\text{remove}(a, l), l)$

THEOREM: remove-preserves-member
$((b \in l) \wedge (a \neq b)) \rightarrow (b \in \text{remove}(a, l))$

```
; Interestingly, I couldn't get the following at first, but in the
; course of performing a manual proof I found the rewrite lemmas
; (added above) that I needed.  By the way, as things are now the
; induction hint below is necessary.
```

THEOREM: remove-card
$(a \in l) \rightarrow (\text{card}(\text{remove}(a, l)) = (\text{card}(l) - 1))$

DEFINITION:
length (lst)
= **if** listp (lst) **then** $1 + \text{length}(\text{cdr}(lst))$
else 0 **endif**

THEOREM: length-remove
$(\text{length}(\text{remove}(a, x)) \neq \text{length}(x)) \rightarrow (\text{length}(\text{remove}(a, x)) < \text{length}(x))$

THEOREM: card-subsetp-ind-help
$\text{listp}(x) \rightarrow (\text{length}(\text{remove}(\text{car}(x), x)) < \text{length}(x))$

DEFINITION:
card-subsetp-ind (x, y)
= **if** listp (x) **then** card-subsetp-ind (remove (car (x), x), remove (car (x), y))
else t endif

THEOREM: remove-preserves-subsetp
$\text{subsetp}(x, y) \rightarrow \text{subsetp}(\text{remove}(a, x), \text{remove}(a, y))$

```
; Interestingly, I came to proving the following lemma by
; trying to do a hand proof of the one after it and
; realizing that the former was conceptually simpler.
; I'm still surprised that the theorem-prover handled the
; former but not (directly) the latter.
```

THEOREM: card-subsetp
$\text{subsetp}(x, y) \rightarrow (\text{card}(x) \leq \text{card}(y))$

THEOREM: card-subsetp-rewrite
$(\text{subsetp}\,(x,\ y) \land (\text{card}\,(x) \neq \text{card}\,(y))) \rightarrow ((\text{card}\,(x) < \text{card}\,(y)) = \mathbf{t})$

```
; Recall that we proved
; SUBSETP-CLOSE1-FIELD :
; (IMPLIES (SUBSETP X (FIELD PAIRS))
;    (SUBSETP (CLOSE1 X) (FIELD PAIRS)))
```

THEOREM: field-of-close1-is-subsetp
$\text{all-pairsp}\,(x) \rightarrow \text{subsetp}\,(\text{field}\,(\text{close1}\,(x)),\ \text{field}\,(x))$

THEOREM: field-of-close1-is-superset
$\text{subsetp}\,(\text{field}\,(x),\ \text{field}\,(\text{close1}\,(x)))$

THEOREM: close1-preserves-card-of-field
$\text{all-pairsp}\,(x) \rightarrow (\text{card}\,(\text{field}\,(\text{close1}\,(x))) = \text{card}\,(\text{field}\,(x)))$

THEOREM: union-card-lessp
$((x \cup y) \neq y) \rightarrow (\text{card}\,(y) < \text{card}\,(x \cup y))$

THEOREM: trans-accept-1
$(\text{all-pairsp}\,(\mathit{pairs}) \land (\mathit{pairs} \neq \text{close1}\,(\mathit{pairs})))$
$\rightarrow \quad ((\text{card}\,(\mathit{pairs}) < \text{card}\,(\text{close1}\,(\mathit{pairs}))) = \mathbf{t})$

THEOREM: card-field-lessp
$(\text{all-pairsp}\,(x) \land (\text{card}\,(x) \neq \text{card}\,(\text{field}\,(x))))$
$\rightarrow \quad ((\text{card}\,(x) < \text{card}\,(\text{field}\,(x))) = \mathbf{t})$

THEOREM: union-preserves-all-pairsp
$(\text{all-pairsp}\,(x) \land \text{all-pairsp}\,(y)) \rightarrow \text{all-pairsp}\,(x \cup y)$

THEOREM: append-preserves-all-pairsp
$(\text{all-pairsp}\,(x) \land \text{all-pairsp}\,(y)) \rightarrow \text{all-pairsp}\,(\text{append}\,(x,\ y))$

THEOREM: newpairs-preserves-all-pairsp
$(\text{all-pairsp}\,(\mathit{pairs1}) \land \text{all-pairsp}\,(\mathit{pairs2}))$
$\rightarrow \quad \text{all-pairsp}\,(\text{newpairs}\,(\mathit{pairs1},\ \mathit{pairs2}))$

THEOREM: all-newpairs-preserves-all-pairsp
$(\text{all-pairsp}\,(\mathit{pairs1}) \land \text{all-pairsp}\,(\mathit{pairs2}))$
$\rightarrow \quad \text{all-pairsp}\,(\text{all-newpairs}\,(\mathit{pairs1},\ \mathit{pairs2}))$

THEOREM: make-set-preserves-all-pairsp
$\text{all-pairsp}\,(\mathit{pairs}) \rightarrow \text{all-pairsp}\,(\text{make-set}\,(\mathit{pairs}))$

THEOREM: close-1-preserves-all-pairsp
all-pairsp $(pairs) \rightarrow$ all-pairsp (close1 $(pairs)$)

THEOREM: lessp-difference-for-trans-accept
$((c < a) \land (c < b)) \rightarrow (((a - b) < (a - c)) = \mathbf{t})$

THEOREM: card-field-not-lessp
all-pairsp $(x) \rightarrow$ ((card (field (x)) $<$ card (x)) $= \mathbf{f}$)

THEOREM: trans-accept
(all-pairsp $(pairs) \land (pairs \neq$ close1 $(pairs)$))
$\rightarrow$ (((card (field (close1 $(pairs)$)) $-$ card (close1 $(pairs)$))
$<$ (card (field $(pairs)$) $-$ card $(pairs)$))
$=$ $\mathbf{t}$)

EVENT: Disable field.

EVENT: Disable close1.

```
; It's unfortunate that the following is needed.
```

EVENT: Disable close1-preserves-card-of-field.

DEFINITION:
trans $(pairs)$
$=$ **if** all-pairsp $(pairs)$
then if $pairs =$ close1 $(pairs)$ **then** $pairs$
else trans (close1 $(pairs)$) **endif**
else $pairs$ **endif**

THEOREM: all-newpairs-closes
((cons $(x, y) \in pairs1$) $\land$ (cons $(y, z) \in pairs2$))
$\rightarrow$ (cons $(x, z) \in$ all-newpairs $(pairs1, pairs2)$)

THEOREM: close1-closes
(all-pairsp $(pairs) \land$ (cons $(x, y) \in pairs$) $\land$ (cons $(y, z) \in pairs$))
$\rightarrow$ (cons $(x, z) \in$ close1 $(pairs)$)

THEOREM: trans-closes
(all-pairsp $(pairs) \land$ (cons $(x, y) \in pairs$) $\land$ (cons $(y, z) \in pairs$))
$\rightarrow$ (cons $(x, z) \in$ trans $(pairs)$)

```
; Finally, we prove that TRANS gives the least transitive set
; containing the original pairs.
```

DEFINITION:
transp (l) = (all-pairsp (l) $\wedge$ subsetp (all-newpairs (l, l), l))

```
;; ok to here 1/04/87 09:49:47
```

THEOREM: member-newpairs
(all-pairsp (l) $\wedge$ $(b \in l)$ $\wedge$ (cdr (a) = car (b)))
$\rightarrow$ (cons (car (a), cdr (b)) $\in$ newpairs (a, l))

THEOREM: newpairs-preserves-subsetp
(all-pairsp (l) $\wedge$ all-pairsp (m) $\wedge$ subsetp (l, m))
$\rightarrow$ subsetp (newpairs (a, l), newpairs (a, m))

THEOREM: all-newpairs-preserves-subsetp-2
(all-pairsp (l) $\wedge$ all-pairsp $(m1)$ $\wedge$ all-pairsp $(m2)$ $\wedge$ subsetp $(m1, m2)$)
$\rightarrow$ subsetp (all-newpairs $(l, m1)$, all-newpairs $(l, m2)$)

THEOREM: member-newpairs-all-newpairs
$(a \in l)$ $\rightarrow$ subsetp (newpairs (a, m), all-newpairs (l, m))

THEOREM: all-newpairs-preserves-subsetp-1
(all-pairsp $(l1)$ $\wedge$ all-pairsp $(l2)$ $\wedge$ all-pairsp (m) $\wedge$ subsetp $(l1, l2)$)
$\rightarrow$ subsetp (all-newpairs $(l1, m)$, all-newpairs $(l2, m)$)

THEOREM: all-newpairs-preserves-subsetp-both
(all-pairsp $(l1)$
$\wedge$ all-pairsp $(l2)$
$\wedge$ all-pairsp $(m1)$
$\wedge$ all-pairsp $(m2)$
$\wedge$ subsetp $(l1, l2)$
$\wedge$ subsetp $(m1, m2)$)
$\rightarrow$ subsetp (all-newpairs $(l1, m1)$, all-newpairs $(l2, m2)$)

THEOREM: close1-preserves-subsetp
(all-pairsp (l) $\wedge$ all-pairsp (m) $\wedge$ subsetp (l, m))
$\rightarrow$ subsetp (close1 (l), close1 (m))

THEOREM: trans-is-least-transp
(all-pairsp $(pairs)$ $\wedge$ subsetp $(pairs, l)$ $\wedge$ transp (l))
$\rightarrow$ subsetp (trans $(pairs)$, l)

```
; Now let's show that the transitive closure of a transitive set is itself.
; In particular, (TRANS (TRANS PAIRS)) = (TRANS PAIRS) for a list PAIRS of
; pairs.  Interestingly, though, the latter result is easily proved by
; a trivial induction on trans:
```

THEOREM: trans-is-idempotent
$\text{trans}\,(\text{trans}\,(\mathit{pairs})) = \text{trans}\,(\mathit{pairs})$

THEOREM: close1-of-transp
$\text{transp}\,(\mathit{pairs}) \rightarrow (\text{close1}\,(\mathit{pairs}) = \mathit{pairs})$

THEOREM: trans-of-transp
$\text{transp}\,(\mathit{pairs}) \rightarrow (\text{trans}\,(\mathit{pairs}) = \mathit{pairs})$

THEOREM: trans-closes-again
$$
\begin{array}{ll}
\multicolumn{2}{l}{(\text{all-pairsp}\,(\mathit{pairs})} \\
\ \wedge & (\text{cons}\,(x,\ y) \in \text{trans}\,(\mathit{pairs})) \\
\ \wedge & (\text{cons}\,(y,\ z) \in \text{trans}\,(\mathit{pairs}))) \\
\rightarrow & (\text{cons}\,(x,\ z) \in \text{trans}\,(\mathit{pairs}))
\end{array}
$$

Index