

Combinatorics on Words formalized
Binary codes that do not preserve primitivity

Štěpán Holub
Martin Raška

April 18, 2024

Funded by the Czech Science Foundation grant GAČR 20-20621S.

Contents

0.1	Lemmas for covered x square	2
0.1.1	Two particular cases	2
0.1.2	Main cases	4
0.2	Square interpretation	9
0.2.1	Locale: interpretation	10
0.2.2	Locale with additional parameters	16
0.2.3	Back to the main locale	19
0.2.4	Locale: Extendable interpretation	22
0.3	General primitivity not preserving codes	26
0.4	Covered uniform square	29
0.4.1	Primitivity (non)preserving uniform binary codes	33
0.5	The main theorem	34
0.5.1	Imprimitive words with single y	34
0.5.2	Conjugate words	36
0.5.3	Square factor of the longer word and both words primitive (was all _{assms})	36
0.5.4	Obtaining primitivity with two squares (refining)	41
0.5.5	Obtaining the square of the longer word (gluing)	43
0.6	Examples	46
0.7	Primitivity non-preserving binary code	47
0.7.1	The target theorem	48
0.8	Upper bound of the power exponent in the canonical imprimitivity witness	50
0.8.1	Optimality of the exponent upper bound	54
0.9	Characterization of binary primitivity preserving morphisms given by a pair of words	54
0.9.1	Code equation for <i>bin-prim</i> predicate	57
0.10	Characterization of binary imprimitivity codes	58
	References	59

`theory Binary-Square-Interpretation`

imports
Combinatorics-Words.Submonoids
Combinatorics-Words.Equations-Basic
begin

0.1 Lemmas for covered x square

This section explores various variants of the situation when $x \cdot x$ is covered with $x \cdot y \textcircled{^k} u \cdot v \cdot y \textcircled{^l} x$, with $y = u \cdot v$, and the displayed dots being synchronized.

0.1.1 Two particular cases

lemma *pref-suf-pers-short*: **assumes** $x \leq_p v \cdot x$ **and** $|v \cdot u| < |x|$ **and** $x \leq_s r \cdot u \cdot v \cdot u$ **and** $r \in \langle \{u, v\} \rangle$

- $x \cdot x$ is covered by $(p \cdot u \cdot v \cdot u) \cdot v \cdot x$, the displayed dots being synchronized
- That is, the condition on the first x in $x \cdot y \textcircled{^k} u \cdot v \cdot y \textcircled{^l} x$ is relaxed

shows $u \cdot v = v \cdot u$

proof (*rule nemp-comm*)

have $v \cdot u <_s x$

using *suf-prod-long-less*[*OF* - $\langle |v \cdot u| < |x| \rangle$, *of* $r \cdot u$, *unfolded rassoc*, *OF* $\langle x \leq_s r \cdot u \cdot v \cdot u \rangle$].

assume $u \neq \varepsilon$ **and** $v \neq \varepsilon$

obtain q **where** $x = q \cdot v \cdot u$ **and** $q \neq \varepsilon$

using $\langle v \cdot u <_s x \rangle$ **by** (*auto simp add: suffix-def*)

hence $q \leq_s r \cdot u$

using $\langle x \leq_s r \cdot u \cdot v \cdot u \rangle$ **by** (*auto simp add: suffix-def*)

from *suf-trans*[*OF* *primroot-suf this*]

have $\varrho q \leq_s r \cdot u$.

have $q \cdot v = v \cdot q$

using *pref-marker*[*OF* $\langle x \leq_p v \cdot x \rangle$, *of* q] $\langle x = q \cdot v \cdot u \rangle$ **by** *simp*

from *suf-marker-per-root*[*OF* $\langle x \leq_p v \cdot x \rangle$, *of* q u , *unfolded rassoc* $\langle x = q \cdot v \cdot u \rangle$]

have $u <_p v \cdot u$

using $\langle v \neq \varepsilon \rangle$ **by** *blast*

from *per-root-primroot*[*OF* *this*]

comm-primroots'[*OF* $\langle q \neq \varepsilon \rangle$ $\langle v \neq \varepsilon \rangle$ $\langle q \cdot v = v \cdot q \rangle$]

have $u \leq_p \varrho q \cdot u$

by *force*

from *gen-prim*[*OF* $\langle r \in \langle \{u, v\} \rangle \rangle$, *unfolded*]

have $r \in \langle \{u, \varrho q\} \rangle$

unfolding $\langle \varrho q = \varrho v \rangle$.

from *two-elem-root-suf-comm*[*OF* $\langle u \leq_p \varrho q \cdot u \rangle$ $\langle \varrho q \leq_s r \cdot u \rangle$ *this*]

show $u \cdot v = v \cdot u$

using *comm-primroot-conv*[*of* - v , *folded* $\langle \varrho q = \varrho v \rangle$] **by** *blast*

qed

lemma *pref-suf-pers-large-overlap*:

assumes

$p \leq p \cdot x$ and $s \leq s \cdot x$ and $p \leq p \cdot r \cdot p$ and $s \leq s \cdot r \cdot s$ and $|x| + |r| \leq |p| + |s|$

shows $x \cdot r = r \cdot x$

using *assms*

proof (*cases* $r = \varepsilon$)

assume $r \neq \varepsilon$ **hence** $r \neq \varepsilon$ **by** *blast*

have $|s| \leq |x|$

using $\langle s \leq s \cdot x \rangle$ **unfolding** *suffix-def* **by** *force*

have $|p| \leq |x|$

using $\langle p \leq p \cdot x \rangle$ **by** (*force simp add: prefix-def*)

have $|r| \leq |p|$

using $\langle |x| + |r| \leq |p| + |s| \rangle$ $\langle |s| \leq |x| \rangle$ **unfolding** *lenmorph* **by** *linarith*

have $|r| \leq |s|$

using $\langle |x| + |r| \leq |p| + |s| \rangle$ $\langle |p| \leq |x| \rangle$ **unfolding** *lenmorph* **by** *linarith*

obtain $p1 \cdot ov \cdot s1$ **where** $p1 \cdot ov \cdot s1 = x$ and $p1 \cdot ov = p$ and $ov \cdot s1 = s$

using *pref-suf-overlapE*[*OF* $\langle p \leq p \cdot x \rangle$ $\langle s \leq s \cdot x \rangle$] **using** $\langle |x| + |r| \leq |p| + |s| \rangle$

by *auto*

have $|r| \leq |ov|$

using $\langle |x| + |r| \leq |p| + |s| \rangle$ [*folded* $\langle p1 \cdot ov \cdot s1 = x \rangle$ $\langle p1 \cdot ov = p \rangle$ $\langle ov \cdot s1 = s \rangle$]

unfolding *lenmorph* **by** *force*

have $r \leq p \cdot p$

using $\langle |r| \leq |p| \rangle$ [*unfolded swap-len*] *pref-prod-long*[*OF* $\langle p \leq p \cdot r \cdot p \rangle$] **by** *blast*

hence $r \leq p \cdot x$

using $\langle p \leq p \cdot x \rangle$ **by** *auto*

have $r \leq s \cdot s$

using $\langle |r| \leq |s| \rangle$ [*unfolded swap-len*] *pref-prod-long*[*reversed*, *OF* $\langle s \leq s \cdot r \cdot s \rangle$]

by *blast*

hence $r \leq s \cdot x$

using $\langle s \leq s \cdot x \rangle$ **by** *auto*

obtain k **where** $p \leq p \cdot r^{\textcircled{a}} k$ $0 < k$

using *per-root-powE*[*OF* *per-rootI*[*OF* $\langle p \leq p \cdot r \cdot p \rangle$ $\langle r \neq \varepsilon \rangle$]] *sprefD1* **by** *metis*

hence $p1 \cdot ov \leq f \cdot r^{\textcircled{a}} k$

unfolding $\langle p1 \cdot ov = p \rangle$ **by** *blast*

obtain l **where** $s \leq s \cdot r^{\textcircled{a}} l$ $0 < l$

using *per-root-powE*[*reversed*, *OF* *per-rootI*[*reversed*, *OF* $\langle s \leq s \cdot r \cdot s \rangle$ $\langle r \neq \varepsilon \rangle$]]

ssufD1 **by** *metis*

hence $ov \cdot s1 \leq f \cdot r^{\textcircled{a}} l$

unfolding $\langle ov \cdot s1 = s \rangle$ **by** *blast*

from *per-glue-facs*[*OF* $\langle p1 \cdot ov \leq f \cdot r^{\textcircled{a}} k \rangle$ $\langle ov \cdot s1 \leq f \cdot r^{\textcircled{a}} l \rangle$ $\langle |r| \leq |ov| \rangle$, *unfolded* $\langle p1 \cdot ov \cdot s1 = x \rangle$]

obtain m **where** $x \leq f \cdot r^{\textcircled{a}} m$.

show $x \cdot r = r \cdot x$

using *root-suf-comm*[*OF*

pref-prod-root[*OF* *marker-fac-pref*[*OF* $\langle x \leq f \cdot r^{\textcircled{a}} m \rangle$ $\langle r \leq p \cdot x \rangle$]]

suffix-appendI[*OF* $\langle r \leq s \cdot x \rangle$]]..

qed *simp*

0.1.2 Main cases

locale *pref-suf-pers* =

fixes $x\ u\ v\ k\ m$

assumes

$x\text{-pref}$: $x \leq_p (v \cdot (u \cdot v)^{\textcircled{m}k}) \cdot x \text{---} \leq_p x (p \cdot x)$ and $\leq_p p (q \cdot p)$ where $q = v \cdot u$
and

$x\text{-suf}$: $x \leq_s x \cdot (u \cdot v)^{\textcircled{m}} \cdot u \text{---} \leq_s x (s \cdot x)$ and $\leq_s s (q' \cdot s)$ where $q' = u \cdot v$
and $k\text{-pos}$: $0 < k$ and $m\text{-pos}$: $0 < m$

begin

lemma *pref-suf-commute-all-commutes*:

assumes $|u \cdot v| \leq |x|$ and $u \cdot v = v \cdot u$

shows *commutes* $\{u, v, x\}$

using *assms*

proof (cases $u \cdot v = \varepsilon$)

let $?p = (v \cdot (u \cdot v)^{\textcircled{m}k})$

let $?s = (u \cdot v)^{\textcircled{m}} \cdot u$

note $x\text{-pref}\ x\text{-suf}$

assume $u \cdot v \neq \varepsilon$

have $?p \neq \varepsilon$ and $?s \neq \varepsilon$ and $v \cdot u \neq \varepsilon$

using $\langle u \cdot v \neq \varepsilon \rangle\ m\text{-pos}\ k\text{-pos}$ by *auto*

obtain r where $u \in r^*$ and $v \in r^*$ and *primitive* r

using $\langle u \cdot v = v \cdot u \rangle\ \text{comm-primrootE}$ by *metis*

hence $r \neq \varepsilon$

by *force*

have $?p \in r^*$ and $?s \in r^*$ and $v \cdot u \in r^*$ and $u \cdot v \in r^*$

using $\langle u \in r^* \rangle\ \langle v \in r^* \rangle$

by (*simp-all add: add-roots root-pow-root*)

have $x \leq_p r \cdot x$

using $\langle ?p \in r^* \rangle\ \langle x \leq_p ?p \cdot x \rangle\ \langle ?p \neq \varepsilon \rangle$ by *blast*

have $v \cdot u \leq_s x$

using *ruler-le*[*reversed*, *OF* - - $\langle |u \cdot v| \leq |x| \rangle$][*unfolded swap-len*[*of* u]],

of $(x \cdot (u \cdot v)^{\textcircled{m}} (m-1) \cdot u) \cdot v \cdot u$, *OF* *triv-suf*, *unfolded rassoc*, *OF* $\langle x \leq_s$
 $x \cdot ?s \rangle$][*unfolded pow-pos*][*OF* $m\text{-pos}$] *rassoc*]].

have $r \leq_s v \cdot u$

using $\langle v \cdot u \neq \varepsilon \rangle\ \langle v \cdot u \in r^* \rangle\ \text{per-root-suf}$ by *blast*

have $r \leq_s r \cdot x$

using *suf-trans*[*OF* $\langle r \leq_s v \cdot u \rangle\ \langle v \cdot u \leq_s x \rangle$, *THEN* *suffix-appendI*] by *blast*

have $x \cdot r = r \cdot x$

using *root-suf-comm*[*OF* $\langle x \leq_p r \cdot x \rangle\ \langle r \leq_s r \cdot x \rangle$, *symmetric*].

hence $x \in r^*$

by (*simp add: <primitive r> prim-comm-root*)

thus *commutes* $\{u, v, x\}$

using $\langle u \in r^* \rangle\ \langle v \in r^* \rangle\ \text{commutesI-root}$ [*of* $\{u, v, x\}$] by *blast*

qed *simp*

lemma *no-overlap*:

assumes

len: $|v \cdot (u \cdot v)^{\textcircled{k}}| + |(u \cdot v)^{\textcircled{m}} \cdot u| \leq |x|$ (**is** $|?p| + |?s| \leq |x|$) **and**
 $0 < k \ 0 < m$

shows *commutes* $\{u, v, x\}$

using *assms*

proof (*cases* $u \cdot v = \varepsilon$)

note *x-pref x-suf*

assume $u \cdot v \neq \varepsilon$

have $?p \neq \varepsilon$ **and** $?s \neq \varepsilon$

using $\langle u \cdot v \neq \varepsilon \rangle$ *m-pos k-pos* **by** *force+*

from *per-lemma-pref-suf*[*OF per-rootI*[*OF* $\langle x \leq p \ ?p \cdot x \ \langle ?p \neq \varepsilon \rangle$] *per-rootI*[*reversed*,
 $OF \ \langle x \leq s \ x \ ?s \ \langle ?s \neq \varepsilon \rangle \ \langle |?p| + |?s| \leq |x| \rangle$]

obtain $r \ s \ kp \ ks \ mw$ **where** $?p = (r \cdot s)^{\textcircled{k}} kp$ **and** $?s = (s \cdot r)^{\textcircled{m}} ks$ **and** $x = (r \cdot s)^{\textcircled{m}} mw \cdot r$ **and** *primitive* $(r \cdot s)$.

hence $\varrho \ ?p = r \cdot s$

using $\langle v \cdot (u \cdot v)^{\textcircled{k}} \neq \varepsilon \rangle$ *comm-primroots nemp-pow-nemp pow-comm prim-self-root* **by** *metis*

moreover **have** $\varrho \ ?s = s \cdot r$

using *primroot-unique*[*OF* $\langle ?s \neq \varepsilon \rangle - \langle ?s = (s \cdot r)^{\textcircled{m}} ks \rangle$] *prim-conjug*[*OF* $\langle \text{primitive } (r \cdot s) \rangle$] **by** *blast*

ultimately **have** $\varrho \ ?p \sim \varrho \ ?s$

by *force*

from *conj-pers-conj-comm*[*OF this k-pos m-pos*]

have $u \cdot v = v \cdot u$.

from *pref-suf-commute-all-commutes*[*OF - this*]

show *commutes* $\{u, v, x\}$

using *len* **by** *auto*

qed *simp*

lemma *no-overlap'*:

assumes

len: $|v \cdot (u \cdot v)^{\textcircled{k}}| + |(u \cdot v)^{\textcircled{m}} \cdot u| \leq |x|$ (**is** $|?p| + |?s| \leq |x|$)
and $0 < k \ 0 < m$

shows $u \cdot v = v \cdot u$

by (*rule commutesE*[*of* $\{u, v, x\}$], *simp-all add: no-overlap*[*OF assms*])

lemma *short-overlap*:

assumes

len1: $|x| < |v \cdot (u \cdot v)^{\textcircled{k}}| + |(u \cdot v)^{\textcircled{m}} \cdot u|$ (**is** $|x| < |?p| + |?s|$) **and**

len2: $|v \cdot (u \cdot v)^{\textcircled{k}}| + |(u \cdot v)^{\textcircled{m}} \cdot u| \leq |x| + |u|$ (**is** $|?p| + |?s| \leq |x| + |u|$)

shows *commutes* $\{u, v, x\}$

proof (*rule pref-suf-commute-all-commutes*)

show $|u \cdot v| \leq |x|$

using *len2* **unfolding** *pow-pos*[*OF k-pos*] *lenmorph* **by** *simp*

next

note *x-pref x-suf*

— obtain the overlap

have $|?p| \leq |x|$
using *len2 unfolding lenmorph by linarith*
hence $?p \leq_p x$
using $\langle x \leq_p ?p \cdot x \rangle$ *pref-prod-long by blast*

have $|?s| \leq |x|$
using *len2 unfolding pow-pos[OF k-pos] pow-len lenmorph by auto*
hence $?s \leq_s x$
using *suf-prod-long[OF $\langle x \leq_s x \cdot ?s \rangle$] by blast*

from *pref-suf-overlapE[OF $\langle ?p \leq_p x \rangle \langle ?s \leq_s x \rangle$ less-imp-le[OF len1]]*
obtain $p1 \text{ } ov \text{ } s1$ **where** $p1 \cdot ov \cdot s1 = x$ **and** $p1 \cdot ov = ?p$ **and** $ov \cdot s1 = ?s$.

from *len1[folded this]*
have $ov \neq \varepsilon$
by *fastforce*

have $|ov| \leq |u|$
using *len2[folded $\langle p1 \cdot ov \cdot s1 = x \rangle \langle p1 \cdot ov = ?p \rangle \langle ov \cdot s1 = ?s \rangle$] unfolding lenmorph by auto*

then obtain s' **where** $ov \cdot s' = u$ **and** $s' \cdot v \cdot (u \cdot v)^{\textcircled{m-1}} \cdot u = s1$
using *eqdE[OF $\langle ov \cdot s1 = ?s \rangle$] unfolded pow-pos[OF m-pos] rassoc] by auto*

— obtain the left complement

from *eqdE[reversed, of $p1 \text{ } ov \text{ } v \cdot (u \cdot v)^{\textcircled{k-1}} \text{ } u \cdot v$, unfolded rassoc, OF $\langle p1 \cdot ov = ?p \rangle$] unfolded pow-pos'[OF k-pos]] $\langle |ov| \leq |u| \rangle$*
have $v \cdot (u \cdot v)^{\textcircled{k-1}} \leq_p p1$
unfolding *lenmorph by (auto simp add: prefix-def)*

then obtain q **where** $v \cdot (u \cdot v)^{\textcircled{k-1}} \cdot q = p1$
by *(force simp add: prefix-def)*

— main proof using the lemma $\llbracket ?u \cdot ?v \cdot ?v \cdot ?u \cdot ?p = ?q \cdot ?u \cdot ?v \cdot ?u; \leq_s ?p ?u; \leq_s ?q ?w; ?w \in \langle \{ ?u, ?v \} \rangle \rrbracket \implies ?v \cdot ?u = ?u \cdot ?v$

show $u \cdot v = v \cdot u$
proof *(rule sym, rule uvu-suf-uvvu)*
show $s' \leq_s u$
using $\langle ov \cdot s' = u \rangle \langle ov \neq \varepsilon \rangle$ **by** *blast*
show $u \cdot v \cdot v \cdot u \cdot s' = q \cdot u \cdot v \cdot u$ — the main fact: the overlap situation
proof—
have $u \cdot v \cdot u \leq_p ?s$
unfolding *pow-pos[OF m-pos] rassoc pref-cancel-conv shift-pow by blast*
hence $p1 \cdot u \cdot v \cdot u \leq_p x$
unfolding $\langle p1 \cdot ov \cdot s1 = x \rangle$ *[symmetric] $\langle ov \cdot s1 = ?s \rangle$ pref-cancel-conv.*
hence $v \cdot (u \cdot v)^{\textcircled{k-1}} \cdot q \cdot u \cdot v \cdot ov \leq_p x$

using $\langle v \cdot (u \cdot v)^{\textcircled{k-1}} \cdot q = p1 \rangle \langle ov \cdot s' = u \rangle$ **by** (*force simp add: prefix-def*)

have $v \cdot u \leq_p x$
using $\langle ?p \leq_p x \rangle$ [*unfolded pow-pos[OF k-pos]*] **by** (*auto simp add: prefix-def*)
have $|?p \cdot v \cdot u| \leq |x|$
using *len2 unfolding pow-pos[OF m-pos] lenmorph by force*
hence $?p \cdot v \cdot u \leq_p x$
using $\langle x \leq_p ?p \cdot x \rangle \langle v \cdot u \leq_p x \rangle$ *pref-prod-longer by blast*
hence $v \cdot (u \cdot v)^{\textcircled{k-1}} \cdot u \cdot v \cdot v \cdot u \leq_p x$
unfolding *pow-pos'[OF k-pos] rassoc.*

have $|v \cdot (u \cdot v)^{\textcircled{k-1}} \cdot u \cdot v \cdot v \cdot u| = |v \cdot (u \cdot v)^{\textcircled{k-1}} \cdot q \cdot u \cdot v \cdot ov|$
using *lenarg[OF $\langle p1 \cdot ov = ?p \rangle$ [folded $\langle v \cdot (u \cdot v)^{\textcircled{k-1}} \cdot q = p1 \rangle$, unfolded pow-pos[OF k-pos] rassoc cancel]]*
by force

from *ruler-eq-len[OF $\langle v \cdot (u \cdot v)^{\textcircled{k-1}} \cdot u \cdot v \cdot v \cdot u \leq_p x \rangle \langle v \cdot (u \cdot v)^{\textcircled{k-1}} \cdot q \cdot u \cdot v \cdot ov \leq_p x \rangle$ this, unfolded cancel]*
have $u \cdot v \cdot v \cdot u = q \cdot u \cdot v \cdot ov.$

thus $u \cdot v \cdot v \cdot u \cdot s' = q \cdot u \cdot v \cdot u$
using $\langle ov \cdot s' = u \rangle$ **by auto**

qed

show $q \leq_s v \cdot u$

proof (*rule ruler-le[reversed]*)

show $q \leq_s x$

proof (*rule suf-trans*)

show $p1 \leq_s x$

using $\langle p1 \cdot ov \cdot s1 = x \rangle$ [*unfolded $\langle ov \cdot s1 = ?s \rangle \langle x \leq_s x \cdot ?s \rangle$ same-suffix-suffix*

by blast

show $q \leq_s p1$

using $\langle v \cdot (u \cdot v)^{\textcircled{k-1}} \cdot q = p1 \rangle$ **by auto**

qed

show $v \cdot u \leq_s x$

using $\langle ?s \leq_s x \rangle$ [*unfolded pow-pos'[OF m-pos] rassoc] suf-extD by metis*

show $|q| \leq |v \cdot u|$

using *lenarg[OF $\langle u \cdot v \cdot v \cdot u \cdot s' = q \cdot u \cdot v \cdot u \rangle$ lenarg[OF $\langle ov \cdot s' = u \rangle$]*

by force

qed

qed auto

qed

lemma *medium-overlap:*

assumes

*len1: $|x| + |u| < |v \cdot (u \cdot v)^{\textcircled{k}}| + |(u \cdot v)^{\textcircled{m}} \cdot u|$ (**is** $|x| + |u| < |?p| + |?s|$)*

and

*len2: $|v \cdot (u \cdot v)^{\textcircled{k}}| + |(u \cdot v)^{\textcircled{m}} \cdot u| < |x| + |u \cdot v|$ (**is** $|?p| + |?s| < |x| + |u \cdot v|$)*

shows $\text{commutes } \{u, v, x\}$
proof (*rule pref-suf-commute-all-commutes*)
show $|u \cdot v| \leq |x|$
using *len2 unfolding pow-pos[OF k-pos]* **by force**
next
note $x\text{-pref } x\text{-suf}$
have $|?p| \leq |x|$
using *len2 unfolding pow-pos[OF m-pos]* **by auto**
hence $?p \leq_p x$
using $\langle x \leq_p ?p \cdot x \rangle$ *pref-prod-long* **by blast**
hence $v \cdot (u \cdot v)^{\textcircled{k-1}} \cdot u \cdot v \cdot v \leq_p ?p \cdot x$
using $\langle x \leq_p ?p \cdot x \rangle$ *unfolding pow-pos'[OF k-pos]* *rassoc* **by** (*auto simp add: prefix-def*)

have $|?s| \leq |x|$
using *len2 unfolding pow-pos[OF k-pos] pow-len lenmorph* **by auto**
hence $?s \leq_s x$
using *suf-prod-long[OF \langle x \leq_s x \cdot ?s \rangle]* **by blast**
then obtain p' **where** $p' \cdot u \cdot v \leq_p x$ **and** $p' \cdot ?s = x$
unfolding *pow-pos[OF m-pos]* **by** (*auto simp add: suffix-def*)

have $|p' \cdot u \cdot v| \leq |?p \cdot v|$
using *len1[folded \langle p' \cdot ?s = x \rangle]* **by force**

have $|v \cdot (u \cdot v)^{\textcircled{k-1}}| < |p'|$
using *len2[folded \langle p' \cdot ?s = x \rangle]* *unfolding pow-pos'[OF k-pos]* **by force**

from *less-imp-le[OF this]*
obtain p **where** $v \cdot (u \cdot v)^{\textcircled{k-1}} \cdot p = p'$
using *ruler-le[OF \langle ?p \leq_p x \rangle \langle p' \cdot u \cdot v \leq_p x \rangle,*
unfolded pow-pos'[OF k-pos] lassoc, THEN pref-cancel-right, THEN pref-cancel-right]
unfolding *lenmorph* **by** (*auto simp add: prefix-def*)

have $|p| \leq |v|$
using $\langle v \cdot (u \cdot v)^{\textcircled{k-1}} \cdot p = p' \rangle \langle |p' \cdot u \cdot v| \leq |?p \cdot v| \rangle$ *unfolding pow-pos'[OF k-pos]* **by force**

show $u \cdot v = v \cdot u$
proof (*rule uv-fac-uvv*)
show $p \cdot u \cdot v \leq_p u \cdot v \cdot v$
proof (*rule pref-cancel[of v \cdot (u \cdot v)^{\textcircled{k-1}}], rule ruler-le*)
show $(v \cdot (u \cdot v)^{\textcircled{k-1}}) \cdot p \cdot u \cdot v \leq_p ?p \cdot x$
unfolding *lassoc \langle v \cdot (u \cdot v)^{\textcircled{k-1}} \cdot p = p' \rangle* [*unfolded lassoc*]
using $\langle p' \cdot u \cdot v \leq_p x \rangle \langle x \leq_p ?p \cdot x \rangle$ *unfolding pow-pos'[OF k-pos]* **by force**
show $(v \cdot (u \cdot v)^{\textcircled{k-1}}) \cdot u \cdot v \cdot v \leq_p (v \cdot (u \cdot v)^{\textcircled{k}}) \cdot x$
unfolding *pow-pos'[OF k-pos] rassoc*
using $\langle v \cdot (u \cdot v)^{\textcircled{k}} \leq_p x \rangle$ **by** (*auto simp add: prefix-def*)
show $|v \cdot (u \cdot v)^{\textcircled{k-1}} \cdot p \cdot u \cdot v| \leq |(v \cdot (u \cdot v)^{\textcircled{k-1}}) \cdot u \cdot v \cdot v|$
using $\langle v \cdot (u \cdot v)^{\textcircled{k-1}} \cdot p = p' \rangle \langle |p' \cdot u \cdot v| \leq |?p \cdot v| \rangle$ *unfolding*

pow-pos'[*OF k-pos*] **by force**
qed

have $p \leq_s x$
using $\langle p' \cdot ?s = x \rangle$ [*folded* $\langle v \cdot (u \cdot v)^{\otimes(k-1)} \cdot p = p' \rangle$] $\langle x \leq_s x \cdot ?s \rangle$ *suf-cancel*
suf-extD **by metis**

from *ruler-le*[*reversed, OF this* $\langle ?s \leq_s x \rangle$, *unfolded pow-pos'*[*OF m-pos*] *rassoc*]
show $p \leq_s (u \cdot v)^{\otimes(m-1)} \cdot u \cdot v \cdot u$
using $\langle |p| \leq |v| \rangle$ **unfolding** *lenmorph* **by auto**

show $(u \cdot v)^{\otimes(m-1)} \cdot u \cdot v \cdot u \in \langle \{u, v\} \rangle$
by (*simp add: gen-in hull-closed power-in*)

show $p \neq \varepsilon$
using $\langle |v \cdot (u \cdot v)^{\otimes(k-1)}| < |p'| \rangle$ $\langle v \cdot (u \cdot v)^{\otimes(k-1)} \cdot p = p' \rangle$ **by force**
qed
qed

thm
no-overlap
short-overlap
medium-overlap

end

thm
pref-suf-pers.no-overlap
pref-suf-pers.short-overlap
pref-suf-pers.medium-overlap
pref-suf-pers.large-overlap

0.2 Square interpretation

In this section fundamental description is given of (the only) possible $\{x, y\}$ -interpretation of the square $x \cdot x$, where $|y| \leq |x|$. The proof is divided into several locales.

lemma *cover-not-disjoint*:

shows *primitive* $(\mathbf{a \cdot b \cdot a \cdot b \cdot a \cdot b \cdot a})$ (**is primitive** $?x$) **and**

primitive $(\mathbf{a \cdot b})$ (**is primitive** $?y$) **and**

$(\mathbf{a \cdot b \cdot a \cdot b \cdot a \cdot b \cdot a}) \cdot (\mathbf{a \cdot b}) \neq (\mathbf{a \cdot b}) \cdot (\mathbf{a \cdot b \cdot a \cdot b \cdot a \cdot b \cdot a})$

(is $?x \cdot ?y \neq ?y \cdot ?x$) **and**

$\varepsilon (\mathbf{a \cdot b \cdot a \cdot b \cdot a \cdot b \cdot a}) \cdot (\mathbf{a \cdot b \cdot a \cdot b \cdot a \cdot b \cdot a}) (\mathbf{b \cdot a \cdot b \cdot a}) \sim_{\mathcal{I}} [(\mathbf{a \cdot b \cdot a \cdot b \cdot a \cdot b \cdot a}), (\mathbf{a \cdot b}), (\mathbf{a \cdot b}), (\mathbf{a \cdot b \cdot a \cdot b \cdot a \cdot b \cdot a})]$

(is $\varepsilon ?x \cdot ?x ?s \sim_{\mathcal{I}} [?x, ?y, ?y, ?x]$)

unfolding *factor-interpretation-def*

by *primitivity-inspection+ force*

0.2.1 Locale: interpretation

locale *square-interp* =

— The basic set of assumptions
 — The goal is to arrive at $ws = [x] \cdot [y]^{\textcircled{a}} k \cdot [x]$ including the description of the interpretation in terms of the first and the second occurrence of x in the interpreted square.

fixes $x y p s ws$

assumes

non-comm: $x \cdot y \neq y \cdot x$ **and**

prim-x: *primitive* x **and**

y-le-x: $|y| \leq |x|$ **and**

ws-lists: $ws \in \text{lists } \{x, y\}$ **and**

nconjug: $\neg x \sim y$ **and**

disj-interp: $p [x, x] s \sim_{\mathcal{D}} ws$

begin

lemma *interp*: $p (x \cdot x) s \sim_{\mathcal{I}} ws$

using *disj-interpD*[*OF disj-interp*] **by** *force*

lemma *disjoint*: $p1 \leq p [x, x] \implies p2 \leq p ws \implies p \cdot \text{concat } p1 \neq \text{concat } p2$

using *disj-interpD1*[*OF disj-interp*].

interpretation *binary-code* $x y$

using *non-comm* **by** *unfold-locales*

lemmas *interpret-concat* = *fac-interpD*(3)[*OF interp*]

lemma *p-nemp*: $p \neq \varepsilon$

using *disjoint*[*of* $\varepsilon \varepsilon$] **by** *auto*

lemma *s-nemp*: $s \neq \varepsilon$

using *disjoint*[*of* $[x, x] ws$] *interpret-concat* **by** *force*

lemma *x-root*: $\rho x = x$

using *prim-x* **by** *blast*

lemma *ws-nemp*: $ws \neq \varepsilon$

using *bin-fst-nemp* *fac-interp-nemp* *interp* **by** *blast*

lemma *hd-ws-lists*: $hd ws \in \{x, y\}$

using *lists-hd-in-set* *ws-lists* *ws-nemp* **by** *auto*

lemma *last-ws-lists*: $last ws \in \{x, y\}$

using *lists-hd-in-set*[*reversed*, *OF ws-nemp ws-lists*].

lemma *kE*: **obtains** k **where** $[hd ws] \cdot [y]^{\textcircled{a}} k \cdot [last ws] = ws$

proof—

from *list.collapse*[*OF ws-nemp*] *hd-word*
obtain *ws'* **where** $ws = [hd\ ws] \cdot ws'$
by *metis*
hence $|hd\ ws| \leq |x|$
using *two-elem-cases*[*OF lists-hd-in-set*[*OF ws-nemp ws-lists*]] *y-le-x* **by** *blast*
hence $|x| \leq |concat\ ws'|$
using *lenarg*[*OF interpret-concat, unfolded lenmorph*]
unfolding *concat.simps emp-simps arg-cong*[*OF* $\langle ws = [hd\ ws] \cdot ws' \rangle$, *of* $\lambda x.$
 $|concat\ x|$, *unfolded concat-morph lenmorph*]
by *linarith*
hence $ws' \neq \varepsilon$
using *nemp-len*[*OF bin-fst-nemp*] **by** *fastforce*
then obtain *mid-ws* **where** $ws' = mid-ws \cdot [last\ ws]$
using $\langle ws = [hd\ ws] \cdot ws' \rangle$ *append-butlast-last-id last-appendR* **by** *metis*
note $\langle ws = [hd\ ws] \cdot ws' \rangle$ [*unfolded this*]
fac-interpD[*OF interp*]
obtain *p'* **where** [*symmetric*]: $p \cdot p' = hd\ ws$ **and** $p' \neq \varepsilon$
using *spref-exE*[*OF* $\langle p <_p\ hd\ ws \rangle$].
obtain *s'* **where** [*symmetric*]: $s' \cdot s = last\ ws$ **and** $s' \neq \varepsilon$
using *spref-exE*[*reversed, OF* $\langle s <_s\ last\ ws \rangle$].
have $p' \cdot concat\ mid-ws \cdot s' = x \cdot x$
using $\langle ws = [hd\ ws] \cdot mid-ws \cdot [last\ ws] \rangle$ [*unfolded* $\langle hd\ ws = p \cdot p' \rangle$ $\langle last\ ws =$
 $s' \cdot s \rangle$]
 $\langle p \cdot (x \cdot x) \cdot s = concat\ ws \rangle$ **by** *simp*
note *over = prim-overlap-sqE*[*OF prim-x, folded this*]
have $mid-ws \in lists\ \{x, y\}$
using $\langle ws = [hd\ ws] \cdot ws' \rangle$ $\langle ws' = mid-ws \cdot [last\ ws] \rangle$ *append-in-lists-conv ws-lists*
by *metis*
have $x \notin set\ mid-ws$
proof
assume $x \in set\ mid-ws$
then obtain *r q* **where** $concat\ mid-ws = r \cdot x \cdot q$
using *concat.simps(2) concat-morph in-set-conv-decomp-first* **by** *metis*
have $(p' \cdot r) \cdot x \cdot (q \cdot s') = x \cdot x$
using $\langle p' \cdot concat\ mid-ws \cdot s' = x \cdot x \rangle$ [*unfolded* $\langle concat\ mid-ws = r \cdot x \cdot q \rangle$]
unfolding *rassoc*.
from *prim-overlap-sqE*[*OF prim-x this*]
show *False*
using $\langle p' \neq \varepsilon \rangle$ $\langle s' \neq \varepsilon \rangle$ **by** *blast*
qed
hence $mid-ws \in lists\ \{y\}$
using $\langle mid-ws \in lists\ \{x, y\} \rangle$ **by** *force*
from *that sing-lists-exp*[*OF this*]
show *thesis*
using $\langle ws = [hd\ ws] \cdot mid-ws \cdot [last\ ws] \rangle$ **by** *metis*
qed

lemma *l-mE*: **obtains** $m\ u\ v\ l$ **where** $(hd\ ws) \cdot y^{\textcircled{m}} \cdot m \cdot u = p \cdot x$ **and** $v \cdot y^{\textcircled{l}} \cdot (last\ ws) = x \cdot s$ **and**

$u \cdot v = y \ u \neq \varepsilon \ v \neq \varepsilon$ **and** $x \cdot (v \cdot u) \neq (v \cdot u) \cdot x$
proof–
note *fac-interpD*[*OF interp*]
obtain k **where** $[hd \ ws] \cdot [y]^{\textcircled{a}}k \cdot [last \ ws] = ws$
using kE .
from *arg-cong*[*OF this, of concat, folded interpret-concat, unfolded concat-morph*
rassoc concat-sing' concat-sing-pow]
have $hd \ ws \cdot y^{\textcircled{a}}k \cdot last \ ws = p \cdot x \cdot x \cdot s$.
have $|hd \ ws| \leq |p \cdot x|$
unfolding *lenmorph* **by** (*rule two-elem-cases*[*OF hd-ws-lists*])
(use dual-order.trans[*OF le-add2 y-le-x*] *le-add2*[*of |x|*] **in fast**)**+**
from *eqd*[*OF - this*]
obtain ya **where** $hd \ ws \cdot ya = p \cdot x$
using $\langle hd \ ws \cdot y^{\textcircled{a}}k \cdot last \ ws = p \cdot x \cdot x \cdot s \rangle$ **by** *auto*
have $|last \ ws| \leq |x|$
unfolding *lenmorph* **using** *dual-order.trans last-ws-lists y-le-x* **by** *auto*
hence $|last \ ws| < |x \cdot s|$
unfolding *lenmorph* **using** *nemp-len*[*OF s-nemp*] **by** *linarith*
from *eqd*[*reversed, OF - less-imp-le*[*OF this*]]
obtain yb **where** $yb \cdot (last \ ws) = x \cdot s$
using $\langle (hd \ ws) \cdot y^{\textcircled{a}}k \cdot (last \ ws) = p \cdot x \cdot x \cdot s \rangle$ *rassoc* **by** *metis*
hence $yb \neq \varepsilon$
using *s-nemp* $\langle |last \ ws| < |x \cdot s| \rangle$ **by** *force*
have $ya \cdot yb = y^{\textcircled{a}}k$
using $\langle (hd \ ws) \cdot y^{\textcircled{a}}k \cdot (last \ ws) = p \cdot x \cdot x \cdot s \rangle$ [*folded* $\langle yb \cdot (last \ ws) = x \cdot s \rangle$,
unfolded lassoc cancel-right, folded $\langle (hd \ ws) \cdot ya = p \cdot x \rangle$, *unfolded rassoc cancel,*
symmetric].
from *pref-mod-pow'*[*OF sprefI*[*OF prefI*[*OF this*]], *folded this*]
obtain $m \ u$ **where** $m < k$ **and** $u <_p y$ **and** $y^{\textcircled{a}}m \cdot u = ya$
using $\langle yb \neq \varepsilon \rangle$ **by** *blast*
have $y^{\textcircled{a}}m \cdot u \cdot (u^{-1} > y) \cdot y^{\textcircled{a}}(k - m - 1) = y^{\textcircled{a}}m \cdot y \cdot y^{\textcircled{a}}(k - m - 1)$
using $\langle u <_p y \rangle$ **by** (*auto simp add: prefix-def*)
also have $\dots = y^{\textcircled{a}}(m + 1 + (k - m - 1))$
using *rassoc add-exps pow-1* **by** *metis*
also have $\dots = y^{\textcircled{a}}k$
using $\langle m < k \rangle$ **by** *auto*
finally obtain $l \ v$ **where** $u \cdot v = y$ **and** $y^{\textcircled{a}}m \cdot u \cdot v \cdot y^{\textcircled{a}}l = y^{\textcircled{a}}k$
using $\langle u <_p y \rangle$ *lq-pref* **by** *blast*
have $concat \ ([hd \ ws] \cdot [y]^{\textcircled{a}}m) = hd \ ws \cdot y^{\textcircled{a}}m$
by *simp*
have $v \neq \varepsilon$
using $\langle u <_p y \rangle \ \langle u \cdot v = y \rangle$ **by** *force*
have $[hd \ ws] \cdot [y]^{\textcircled{a}}m \leq_p ws$
using $\langle [hd \ ws] \cdot [y]^{\textcircled{a}}k \cdot [last \ ws] = ws \rangle$ [*folded pop-pow*[*OF less-imp-le*[*OF* $\langle m < k \rangle$]]] **by** *fastforce*
from *disjoint*[*OF - this, of [x], unfolded* $\langle concat \ ([hd \ ws] \cdot [y]^{\textcircled{a}}m) = hd \ ws \cdot y^{\textcircled{a}}m \rangle$]
have $u \neq \varepsilon$
using $\langle (hd \ ws) \cdot ya = p \cdot x \rangle$ [*folded* $\langle y^{\textcircled{a}}m \cdot u = ya \rangle$] *s-nemp* **by** *force*

have $x \cdot (v \cdot u) \neq (v \cdot u) \cdot x$
proof
assume $x \cdot v \cdot u = (v \cdot u) \cdot x$
from *comm-primroots'*[*OF bin-fst-nemp suf-nemp*[*OF* $\langle u \neq \varepsilon \rangle$] *this, unfolded*
x-root]
have $x = \varrho (v \cdot u)$.
thus *False*
using $\langle u \cdot v = y \rangle$ *nconjug y-le-x*
using *conjugI' nle-le pref-same-len primroot-emp primroot-len-le primroot-pref*
swap-len **by** *metis*
qed
with *that*[*of* $m \ u \ u^{-1} \rangle y \ l$, *OF* $\langle hd \ ws \cdot ya = p \cdot x \rangle$ [*folded* $\langle y^{\textcircled{m}} \cdot u = ya \rangle$],
folded $\langle yb \cdot last \ ws = x \cdot s \rangle \langle u \cdot v = y \rangle$,
unfolded *lq-triv lassoc cancel-right*, *OF* $\langle u \neq \varepsilon \rangle \langle v \neq \varepsilon \rangle$ *this*[*unfolded lassoc*]
show *thesis*
using $\langle y^{\textcircled{m}} \cdot u \cdot v \cdot y^{\textcircled{l}} = y^{\textcircled{k}} \rangle$ [*folded* $\langle ya \cdot yb = y^{\textcircled{k}} \rangle \langle y^{\textcircled{m}} \cdot u =$
 $ya \rangle$, *unfolded* *rassoc cancel*, *folded* $\langle u \cdot v = y \rangle$] **by** *blast*
qed

lemma *last-ws: last ws = x*
proof(*rule ccontr*)
assume $last \ ws \neq x$
hence $last \ ws = y$
using *last-ws-lists* **by** *blast*
obtain $l \ m \ u \ v$ **where** $(hd \ ws) \cdot y^{\textcircled{m}} \cdot u = p \cdot x$ **and** $v \cdot y^{\textcircled{l}} \cdot (last \ ws) = x \cdot s$ **and**
 $u \cdot v = y$ **and** $u \neq \varepsilon$ **and** $v \neq \varepsilon$ **and** $x \cdot v \cdot u \neq (v \cdot u) \cdot x$
using *l-mE* **by** *metis*
note $y \cdot le \cdot x$ [*folded* $\langle u \cdot v = y \rangle$, *unfolded* *swap-len*[*of* u]]

from $\langle v \cdot y^{\textcircled{l}} \cdot (last \ ws) = x \cdot s \rangle$ [*unfolded* $\langle last \ ws = y \rangle$, *folded* $\langle u \cdot v = y \rangle$]
have $x \leq_p (v \cdot u)^{\textcircled{Suc \ l}} \cdot v$
unfolding *pow-Suc'* *rassoc* **using** *append-eq-appendI prefix-def shift-pow* **by**
metis
moreover **have** $(v \cdot u)^{\textcircled{Suc \ l}} \cdot v \leq_p (v \cdot u) \cdot (v \cdot u)^{\textcircled{Suc \ l}} \cdot v$
unfolding *lassoc pow-comm*[*symmetric*] **using** *rassoc* **by** *blast*
ultimately **have** $x \leq_p (v \cdot u) \cdot x$
using *pref-keeps-per-root* **by** *blast*

thus *False*
proof (*cases* $m = 0$)
assume $m \neq 0$
have $v \cdot u \leq_s x$
using $\langle (hd \ ws) \cdot y^{\textcircled{m}} \cdot u = p \cdot x \rangle$ [*folded* $\langle u \cdot v = y \rangle$, *unfolded* *pow-pos'*[*OF* $\langle m \neq$
 $0 \rangle$][*unfolded* *neq0-conv*]] *rassoc*
suf-extD[*THEN* *suf-prod-long*[*OF* $\langle |v \cdot u| \leq |x| \rangle$], *of* $p \ hd \ ws \cdot (u \cdot v)^{\textcircled{m-1}}$
 $\cdot u$, *unfolded* *rassoc*] **by** *simp*
have [*symmetric*]: $(v \cdot u) \cdot x = x \cdot (v \cdot u)$
using *root-suf-comm'*[*OF* $\langle x \leq_p (v \cdot u) \cdot x \rangle \langle (v \cdot u) \leq_s x \rangle$].
thus *False*

```

    using  $\langle x \cdot v \cdot u \neq (v \cdot u) \cdot x \rangle$  by blast
next
assume  $m = 0$ 
thus False
proof (cases  $hd\ ws = y$ )
  assume  $hd\ ws = y$ 
  have  $p \cdot (x \cdot x) \cdot s = y^{\textcircled{m}} \text{Suc} (\text{Suc} (\text{Suc} (m+l)))$ 
    unfolding  $rassoc \langle v \cdot y^{\textcircled{l}} \cdot (last\ ws) = x \cdot s \rangle$  [unfolded  $\langle last\ ws = y \rangle$ ,
symmetric] power-Suc2
    unfolding  $lassoc \langle (hd\ ws) \cdot y^{\textcircled{m}} \cdot u = p \cdot x \rangle$  [unfolded  $\langle hd\ ws = y \rangle$ , symmetric]
     $\langle u \cdot v = y \rangle$  [symmetric]
    by comparison
  have  $\varrho\ x \sim \varrho\ y$ 
proof (rule fac-two-conjug-primroot')
  show  $x \neq \varepsilon$  and  $y \neq \varepsilon$  using bin-fst-nemp bin-snd-nemp.
  show  $x \cdot x \leq_f y^{\textcircled{m}} \text{Suc} (\text{Suc} (\text{Suc} (m+l)))$ 
    using facI[of  $x \cdot x\ p\ s$ , unfolded  $\langle p \cdot (x \cdot x) \cdot s = y^{\textcircled{m}} \text{Suc} (\text{Suc} (\text{Suc} (m+l))) \rangle$ ].
  show  $x \cdot x \leq_f x^{\textcircled{2}}$ 
    unfolding pow-two by blast
  show  $|x| + |y| \leq |x \cdot x|$ 
    using y-le-x unfolding lenmorph by auto
qed
thus False
  unfolding x-root using nconjug y-le-x
  by (metis conjug-len long-pref primroot-pref)
next
assume  $hd\ ws \neq y$ 
hence  $hd\ ws = x$ 
  using hd-ws-lists by auto

  have  $x \leq_s x \cdot u$ 
    using  $\langle (hd\ ws) \cdot y^{\textcircled{m}} \cdot u = p \cdot x \rangle$  [unfolded  $\langle m = 0 \rangle \langle hd\ ws = x \rangle$  pow-zero
emp-simps]
    by (simp add: suffix-def)
  have  $v \cdot u \leq_p x$ 
    using  $\langle x \leq_p (v \cdot u) \cdot x \rangle$  y-le-x [folded  $\langle u \cdot v = y \rangle$ , unfolded swap-len [of  $u$ ]]
    pref-prod-long by blast
  hence  $|v \cdot u| < |x|$ 
    using nconjug conjugI[OF -  $\langle u \cdot v = y \rangle$ , of  $x$ ]  $\langle |v \cdot u| \leq |x| \rangle$ 
    le-neq-implies-less pref-same-len by blast
  have  $u \cdot v = v \cdot u$ 
proof (rule pref-suf-pers-short [reversed])
  from  $\langle x \leq_p (v \cdot u)^{\textcircled{m}} \text{Suc}\ l \cdot v \rangle$ 
  show  $x \leq_p ((v \cdot u) \cdot v) \cdot (u \cdot v)^{\textcircled{l}}$ 
    by comparison
  show  $(u \cdot v)^{\textcircled{l}} \in \langle \{v, u\} \rangle$ 
    by blast
qed fact+
from pref-extD[OF  $\langle v \cdot u \leq_p x \rangle$  [folded  $\langle u \cdot v = v \cdot u \rangle$ ]]

```

```

have  $x \cdot u = u \cdot x$ 
  using  $\langle x \leq_s x \cdot u \rangle$  suf-root-pref-comm by blast
with comm-trans[OF this  $\langle u \cdot v = v \cdot u \rangle$  [symmetric]  $\langle u \neq \varepsilon \rangle$ ]
have  $x \cdot (v \cdot u) = (v \cdot u) \cdot x$ 
  using comm-prod by blast
thus False
  using  $\langle x \cdot v \cdot u \neq (v \cdot u) \cdot x \rangle$  by blast
qed
qed
qed

lemma rev-square-interp:
  square-interp (rev  $x$ ) (rev  $y$ ) (rev  $s$ ) (rev  $p$ ) (rev (map rev  $ws$ ))
proof (unfold-locales)
  show rev (map rev  $ws$ )  $\in$  lists {rev  $x$ , rev  $y$ }
    using ws-lists by force
  show  $|$ rev  $y$  $| \leq |$ rev  $x$  $|$ 
    using y-le-x by simp
  show  $\neg$  (rev  $x$ )  $\sim$  (rev  $y$ )
    by (simp add: conjug-rev-conv nconjug)
  show primitive (rev  $x$ )
    using prim-x
    by (simp-all add: prim-rev-iff)
  show (rev  $s$ ) [rev  $x$ , rev  $x$ ] (rev  $p$ )  $\sim_{\mathcal{D}}$  (rev (map rev  $ws$ ))
proof
  show (rev  $s$ ) (concat [rev  $x$ , rev  $x$ ] (rev  $p$ ))  $\sim_{\mathcal{I}}$  rev (map rev  $ws$ )
    using interp rev-fac-interp by fastforce
  show  $\bigwedge p1\ p2. p1 \leq_p$  [rev  $x$ , rev  $x$ ]  $\implies p2 \leq_p$  rev (map rev  $ws$ )  $\implies$  rev  $s \cdot$ 
concat  $p1 \neq$  concat  $p2$ 
proof
  fix  $p1'\ p2'$  assume  $p1' \leq_p$  [rev  $x$ , rev  $x$ ] and  $p2' \leq_p$  rev (map rev  $ws$ ) and
rev  $s \cdot$  concat  $p1' =$  concat  $p2'$ 
  obtain  $p1\ p2$  where  $p1' \cdot p1 =$  [rev  $x$ , rev  $x$ ] and  $p2' \cdot p2 =$  rev (map rev  $ws$ )
    using  $\langle p1' \leq_p$  [rev  $x$ , rev  $x$ ]  $\rangle$   $\langle p2' \leq_p$  rev (map rev  $ws$ )  $\rangle$  by (auto simp add:
prefix-def)
  hence rev  $s \cdot$  (concat  $p1' \cdot$  concat  $p1$ )  $\cdot$  rev  $p =$  concat  $p2' \cdot$  concat  $p2$ 
    unfolding concat-morph[symmetric] using  $\langle$ (rev  $s$ ) (concat[rev  $x$ , rev  $x$ ]) (rev
 $p$ )  $\sim_{\mathcal{I}}$  rev (map rev  $ws$ ) $\rangle$ 
    fac-interpD(3) by force
  from this[unfolded lassoc, folded  $\langle$ rev  $s \cdot$  concat  $p1' =$  concat  $p2'$  $\rangle$ , unfolded
rassoc cancel]
  have concat  $p1 \cdot$  rev  $p =$  concat  $p2$ .
  hence  $p \cdot$  (concat (rev (map rev  $p1$ ))) = concat (rev (map rev  $p2$ ))
    using rev-append rev-concat rev-map rev-rev-ident by metis
  have rev (map rev  $p1$ )  $\leq_p$  [ $x$ ,  $x$ ]
    using arg-cong[OF  $\langle p1' \cdot p1 =$  [rev  $x$ , rev  $x$ ] $\rangle$ , of  $\lambda x. \text{rev} (\text{map } \text{rev } x)$ ,
unfolded map-append rev-append]
    by fastforce
  have rev (map rev  $p2$ )  $\leq_p$   $ws$ 

```


using *arg-cong*[*OF* $\langle p2'.p2 = \text{rev} (\text{map rev ws}) \rangle$, of $\lambda x. \text{rev} (\text{map rev } x)$,
unfolded map-append rev-append rev-map
rev-rev-ident map-rev-involution, folded rev-map] **by** *blast*
from *disjoint*[*OF* $\langle \text{rev} (\text{map rev } p1) \leq p [x,x] \rangle \langle \text{rev} (\text{map rev } p2) \leq p ws \rangle$]
show *False*
using $\langle p \cdot (\text{concat} (\text{rev} (\text{map rev } p1))) = \text{concat} (\text{rev} (\text{map rev } p2)) \rangle$ **by**
blast
qed
qed
show $\text{rev } x \cdot \text{rev } y \neq \text{rev } y \cdot \text{rev } x$
using *non-comm unfolding comm-rev-iff*.
qed

lemma *hd-ws*: $\text{hd } ws = x$
using *square-interp.last-ws*[*reversed, OF rev-square-interp*]
unfolding *hd-map*[*OF ws-nemp*]
by *simp*

lemma *p-pref*: $p < p x$
using *fac-interpD(1) hd-ws interp* **by** *auto*

lemma *s-suf*: $s < s x$
using *fac-interpD(2) last-ws interp* **by** *auto*

end

0.2.2 Locale with additional parameters

locale *square-interp-plus* = *square-interp* +
fixes $l m u v$
assumes *fst-x*: $x \cdot y^{\textcircled{m}} \cdot u = p \cdot x$ **and**
snd-x: $v \cdot y^{\textcircled{l}} \cdot x = x \cdot s$ **and**
uv-y: $u \cdot v = y$ **and**
u-nemp: $u \neq \varepsilon$ **and** *v-nemp*: $v \neq \varepsilon$ **and**
vu-x-non-comm: $x \cdot (v \cdot u) \neq (v \cdot u) \cdot x$
begin

interpretation *binary-code* $x y$
using *non-comm* **by** *unfold-locales*

lemma *rev-square-interp-plus*: $\text{square-interp-plus} (\text{rev } x) (\text{rev } y) (\text{rev } s) (\text{rev } p)$
 $(\text{rev} (\text{map rev } ws)) m l (\text{rev } v) (\text{rev } u)$

proof–

note *fac-interpD*[*OF interp, unfolded hd-ws last-ws*]
note $\langle s < s x \rangle$ [*unfolded strict-suffix-to-prefix*]
note $\langle p < p x \rangle$ [*unfolded spref-rev-suf-iff*]

interpret *i*: $\text{square-interp} (\text{rev } x) (\text{rev } y) (\text{rev } s) (\text{rev } p) (\text{rev} (\text{map rev } ws))$

using *rev-square-interp.*
show *?thesis*
by *standard*
(simp-all del: rev-append add: rev-pow[symmetric] rev-append[symmetric],
simp-all add: fst-x snd-x uv-y v-nemp u-nemp vu-x-non-comm[symmetric],
unfolded rassoc])
qed

Exactly one of the exponents is zero: impossible.

Uses lemma $\llbracket \leq_p ?x (?v \cdot ?x); |?v \cdot ?u| < |?x|; \leq_s ?x (?r \cdot ?u \cdot ?v \cdot ?u); ?r \in \langle \{?u, ?v\} \rangle \rrbracket \implies ?u \cdot ?v = ?v \cdot ?u$ and exploits the symmetric interpretation.

lemma *fst-exp-zero: assumes $m = 0$ and $0 < l$ shows False*

proof *(rule notE[OF vu-x-non-comm])*

note *y-le-x[folded uv-y, unfolded swap-len[of u]]*

have $x \leq_p (v \cdot (u \cdot v)^{\textcircled{a}} l) \cdot x$

unfolding *rassoc using snd-x[folded uv-y] by blast*

have $v \cdot (u \cdot v)^{\textcircled{a}} l \neq \varepsilon$

using *v-nemp by force*

obtain *exp where $x \leq_p (v \cdot (u \cdot v)^{\textcircled{a}} l)^{\textcircled{a}} \text{exp}$ $0 < \text{exp}$*

using *per-root-powE[OF per-rootI[OF $\langle x \leq_p (v \cdot (u \cdot v)^{\textcircled{a}} l) \cdot x \rangle \langle v \cdot (u \cdot v)^{\textcircled{a}} l \neq \varepsilon \rangle$, of thesis] by blast*

have $x \leq_s x \cdot u$

using *fst-x[unfolded $\langle m = 0 \rangle$ pow-zero emp-simps] by (simp add: suffix-def)*

have $((v \cdot u) \cdot v) \cdot ((u \cdot v)^{\textcircled{a}}(l-1)) \cdot (v \cdot (u \cdot v)^{\textcircled{a}} l)^{\textcircled{a}}(\text{exp}-1) = (v \cdot (u \cdot v)^{\textcircled{a}} l)^{\textcircled{a}} \text{exp}$

(is $((v \cdot u) \cdot v) \cdot ?\text{suf} = (v \cdot (u \cdot v)^{\textcircled{a}} l)^{\textcircled{a}} \text{exp}$)

using $\langle 0 < l \rangle \langle 0 < \text{exp} \rangle$ **by** *comparison*

have $v \cdot u \leq_p x$

using *pref-prod-longer[OF $\langle x \leq_p (v \cdot (u \cdot v)^{\textcircled{a}} l) \cdot x \rangle$ [unfolded rassoc] - $\langle |v \cdot u| \leq |x| \rangle$]*

unfolding *pow-pos[OF $\langle 0 < l \rangle$] rassoc by blast*

hence $|v \cdot u| < |x|$

using *nconjug conjugI[OF - uv-y, of x] $\langle |v \cdot u| \leq |x| \rangle$*

le-neq-implies-less pref-same-len by blast

have $u \cdot v = v \cdot u$

proof *(rule pref-suf-pers-short[reversed])*

show $x \leq_p ((v \cdot u) \cdot v) \cdot ?\text{suf}$

unfolding $\langle ((v \cdot u) \cdot v) \cdot ?\text{suf} = (v \cdot (u \cdot v)^{\textcircled{a}} l)^{\textcircled{a}} \text{exp} \rangle$ **by** *fact*

show $((u \cdot v)^{\textcircled{a}}(l-1)) \cdot (v \cdot (u \cdot v)^{\textcircled{a}} l)^{\textcircled{a}}(\text{exp}-1) \in \langle \{v, u\} \rangle$

by *(simp add: gen-in hull-closed power-in)*

qed *fact+*

show $x \cdot v \cdot u = (v \cdot u) \cdot x$

using *root-suf-comm[OF - $\langle x \leq_s x \cdot u \rangle$] pref-keeps-per-root comm-trans[OF $\langle u \cdot v = v \cdot u \rangle$ [symmetric] - u-nemp, symmetric] $\langle v \cdot u \leq_p x \rangle$ comm-prod prefI*

by *metis*

qed

lemma *snd-exp-zero*: **assumes** $0 < m$ **and** $l = 0$ **shows** *False*
using *square-interp-plus.fst-exp-zero*[*OF rev-square-interp-plus, reversed,*
rotated, OF assms].

Both exponents positive: impossible

lemma *both-exps-pos*: **assumes** $0 < m$ **and** $0 < l$ **shows** *False*

proof–

note *fac-interpD*[*OF interp, unfolded hd-ws last-ws*]
have $|p| \leq |x|$ **and** $|s| \leq |x|$
using *pref-len*[*OF sprefD1*[*OF <p <p x>*]] *suf-len*[*OF ssufD1*[*OF <s <s x>*]].

have $x \leq_p (v \cdot (u \cdot v)^{\textcircled{m}}) \cdot x$
(is $x \leq_p$ *?pref* $\cdot x$)
using *snd-x*[*folded uv-y*] **by** *force*
moreover **have** $x \leq_s x \cdot ((u \cdot v)^{\textcircled{m}} \cdot u)$
(is $x \leq_s$ $x \cdot$ *?suf*)
using *fst-x*[*folded uv-y*] **by** *force*

ultimately interpret *pref-suf-pers* $x u v l m$
using $\langle 0 < l \rangle \langle 0 < m \rangle$ **by** *unfold-locale*

have *?pref* $\leq_p x$
using *snd-x*[*folded uv-y rassoc, symmetric*] *eqd*[*reversed, OF - <|s| ≤ |x|>*] **by**
blast

have *?suf* $\leq_s x$
using *fst-x*[*folded uv-y, symmetric*] *eqd*[*OF - <|p| ≤ |x|>*] **by** *blast*

have *in-particular*: *commutes* $\{u, v, x\} \implies x \cdot (v \cdot u) = (v \cdot u) \cdot x$
unfolding *commutes-def* **by** (*rule comm-prod*) *blast+*

— Case analysis based on (slightly modified) lemmas for covered x square.

note *no-overlap-comm* = *no-overlap*[*THEN in-particular*] **and**
short-overlap-comm = *short-overlap*[*THEN in-particular*] **and**
medium-overlap-comm = *medium-overlap*[*THEN in-particular*] **and**
large-overlap-conjug = *pref-suf-pers-large-overlap*[*OF <?pref ≤ p x> <?suf ≤ s x>*,
of v · u]

consider

(*no-overlap*) $|?pref| + |?suf| \leq |x|$ |
(*short-overlap*) $|x| < |?pref| + |?suf| \wedge |?pref| + |?suf| \leq |x| + |u|$ |
(*medium-overlap*) $|x| + |u| < |?pref| + |?suf| \wedge |?pref| + |?suf| < |x| + |u \cdot$
 $v|$ |

(*large-overlap*) $|x| + |v \cdot u| \leq |?pref| + |?suf|$

unfolding *swap-len*[*of v*] **by** *linarith*

thus *False*

proof (*cases*)

case *no-overlap*

```

then show False
  using no-overlap-comm vu-x-non-comm  $\langle 0 < l \rangle \langle 0 < m \rangle$  by blast
next
case short-overlap
then show False
  using short-overlap-comm vu-x-non-comm by blast
next
case medium-overlap
then show False
  using medium-overlap-comm vu-x-non-comm by blast
next
case large-overlap
show False
  thm large-overlap-conjug nconjug
proof (rule notE[OF vu-x-non-comm], rule large-overlap-conjug[OF -- large-overlap])
  have  $(u \cdot v)^{\textcircled{a}} (l-1) \leq_p (u \cdot v)^{\textcircled{a}} \text{Suc } (l-1)$ 
    using pref-pow-ext by blast
  thus  $v \cdot (u \cdot v)^{\textcircled{a}} l \leq_p (v \cdot u) \cdot v \cdot (u \cdot v)^{\textcircled{a}} l$ 
    unfolding pow-pos[OF  $\langle 0 < l \rangle$ ] pow-Suc assoc pref-cancel-conv.
  show  $(u \cdot v)^{\textcircled{a}} m \cdot u \leq_s ((u \cdot v)^{\textcircled{a}} m \cdot u) \cdot v \cdot u$ 
    by comparison
qed
qed
qed

thm suf-cancel-conv

end

```

0.2.3 Back to the main locale

context *square-interp*

begin

definition *u where* $u = x^{-1} \triangleright (p \cdot x)$

definition *v where* $v = (x \cdot s)^{\langle -1 \rangle} x$

lemma *cover-xyx*: $ws = [x, y, x]$ **and** *vu-x-non-comm*: $x \cdot (v \cdot u) \neq (v \cdot u) \cdot x$ **and**
uv-y: $u \cdot v = y$ **and**

px-xu: $p \cdot x = x \cdot u$ **and** *vx-xs*: $v \cdot x = x \cdot s$ **and** *u-nemp*: $u \neq \varepsilon$ **and** *v-nemp*:
 $v \neq \varepsilon$

proof–

obtain *k where* *ws*: $[x] \cdot [y]^{\textcircled{a}} k \cdot [x] = ws$

using *kE*[*unfolded hd-ws last-ws*].

obtain *m u' v' l where* $x \cdot y^{\textcircled{a}} m \cdot u' = p \cdot x$ **and** $v' \cdot y^{\textcircled{a}} l \cdot x = x \cdot s$ **and**
 $u' \cdot v' = y$

and $u' \neq \varepsilon$ **and** $v' \neq \varepsilon$ **and** $x \cdot v' \cdot u' \neq (v' \cdot u') \cdot x$

using *l-mE*[*unfolded hd-ws last-ws*].

then interpret *square-interp-plus* $x\ y\ p\ s\ ws\ l\ m\ u'\ v'$
by (*unfold-locales*)
have $m = 0$ **and** $l = 0$ **and** $y \neq \varepsilon$
using *both-exps-pos snd-exp-zero fst-exp-zero* $\langle u' \cdot v' = y \rangle \langle u' \neq \varepsilon \rangle$ **by** *blast+*
have $u' = u$
unfolding *u-def*
using *conjug-lq*[*OF fst-x*[*unfolded* $\langle m = 0 \rangle$ *pow-zero emp-simps, symmetric*]].
have $v' = v$
unfolding *v-def*
using *conjug-lq*[*reversed, OF snd-x*[*unfolded* $\langle l = 0 \rangle$ *pow-zero emp-simps, symmetric*]].
have $x \cdot y^{\textcircled{m}} \cdot (u' \cdot v') \cdot y^{\textcircled{l}} \cdot x = \text{concat } ws$
unfolding *interpret-concat*[*symmetric*] **using** *fst-x snd-x* **by** *force*
from *this*[*folded ws, unfolded* $\langle u' \cdot v' = y \rangle \langle m = 0 \rangle \langle l = 0 \rangle$ *pow-zero emp-simps*]
have $k = 1$
unfolding *eq-pow-exp*[*OF* $\langle y \neq \varepsilon \rangle$, *of k 1, symmetric*] *pow-1 concat-morph*
concat-pow
by *simp*
from *ws*[*unfolded this pow-1*]
show $ws = [x, y, x]$ **by** *simp*
show $u \cdot v = y$
unfolding $\langle u' = u \rangle$ [*symmetric*] $\langle v' = v \rangle$ [*symmetric*] **by** *fact+*
show $p \cdot x = x \cdot u$
using $\langle x \cdot y^{\textcircled{m}} \cdot m \cdot u' = p \cdot x \rangle$ [*unfolded* $\langle m = 0 \rangle \langle u' = u \rangle$ *pow-zero emp-simps, symmetric*].
show $v \cdot x = x \cdot s$
using $\langle v' \cdot y^{\textcircled{l}} \cdot l \cdot x = x \cdot s \rangle$ [*unfolded* $\langle l = 0 \rangle \langle v' = v \rangle$ *pow-zero emp-simps*].
show $x \cdot (v \cdot u) \neq (v \cdot u) \cdot x$
using $\langle x \cdot v' \cdot u' \neq (v' \cdot u') \cdot x \rangle$ [*unfolded* $\langle u' = u \rangle \langle v' = v \rangle$].
show $u \neq \varepsilon$ **and** $v \neq \varepsilon$
using $\langle u' \neq \varepsilon \rangle \langle v' \neq \varepsilon \rangle$ **unfolding** $\langle u' = u \rangle \langle v' = v \rangle$.
qed

lemma *cover*: $x \cdot y \cdot x = p \cdot x \cdot x \cdot s$
using *interpret-concat cover-xyx* **by** *auto*

lemma *conjug-facs*: $q\ u \sim q\ v$

proof–

note *sufI*[*OF px-xu*]
have $u \neq \varepsilon$
using *p-nemp px-xu* **by** *force*
obtain *expu* **where** $x < s\ u^{\textcircled{m}}\ expu\ 0 < expu$
using *per-root-powE*[*reversed, OF per-rootI*[*reversed, OF* $\langle x \leq s\ x \cdot u \rangle \langle u \neq \varepsilon \rangle$]].
hence $x \leq f\ u^{\textcircled{m}}\ expu$
using *ssufD1* **by** *blast*

note *prefI*[*OF vx-xs*[*symmetric*]]
have $v \neq \varepsilon$

using *s-nemp vx-xs* **by force**
obtain *expv* **where** $x < p \ v^{\textcircled{a}} \text{expv} \ 0 < \text{expv}$
using *per-root-powE*[*OF per-rootI*[*OF* $\langle x \leq p \ v \cdot x \rangle \langle v \neq \varepsilon \rangle$]].
hence $x \leq f \ v^{\textcircled{a}} \text{expv}$ **by blast**

show $\varrho \ u \sim \varrho \ v$
proof(*rule fac-two-conjug-primroot'*[*OF* $\langle x \leq f \ u^{\textcircled{a}} \text{expu} \rangle \langle x \leq f \ v^{\textcircled{a}} \text{expv} \rangle \langle u \neq \varepsilon \rangle$
 $\langle v \neq \varepsilon \rangle$])
show $|u| + |v| \leq |x|$
using *y-le-x*[*folded uv-y, unfolded lenmorph*] **by fastforce**
qed
qed

term *square-interp.v*

— We have a detailed information about all words

lemma *bin-sq-interpE*: **obtains** $r \ t \ m \ k \ l$
where $(t \cdot r)^{\textcircled{a}} k = u$ **and** $(r \cdot t)^{\textcircled{a}} l = v$ **and**
 $(r \cdot t)^{\textcircled{a}} m \cdot r = x$ **and** $(t \cdot r)^{\textcircled{a}} k \cdot (r \cdot t)^{\textcircled{a}} l = y$
and $(r \cdot t)^{\textcircled{a}} k = p$ **and** $(t \cdot r)^{\textcircled{a}} l = s$ **and** $r \cdot t \neq t \cdot r$ **and**
 $0 < k$ **and** $0 < m$ **and** $0 < l$

proof–

obtain $r \ t \ k \ m$ **where** $(r \cdot t)^{\textcircled{a}} k = p$ **and** $(t \cdot r)^{\textcircled{a}} k = u$ **and** $(r \cdot t)^{\textcircled{a}} m \cdot r = x$
and
 $t \neq \varepsilon$ **and** $0 < k$ **and** *primitive* $(r \cdot t)$
using *conjug-eq-primrootE*[*OF px-xu p-nemp*].
have $t \cdot r = \varrho \ u$
using *prim-conjug*[*OF* $\langle \text{primitive} \ (r \cdot t) \rangle$, *THEN primroot-unique*[*OF u-nemp*],
OF conjugI' $\langle (t \cdot r)^{\textcircled{a}} k = u \rangle$ [*symmetric*]].

have $0 < m$

proof (*rule ccontr*)

assume $\neg 0 < m$

hence $x = r$ **using** $\langle (r \cdot t)^{\textcircled{a}} m \cdot r = x \rangle$ **by simp**

show *False*

using $\langle 0 < k \rangle \langle (r \cdot t)^{\textcircled{a}} k = p \rangle \langle x = r \rangle$ *comp-pows-pref-zero p-pref* **by blast**

qed

from $\langle (r \cdot t)^{\textcircled{a}} m \cdot r = x \rangle$ [*unfolded pow-pos*[*OF* $\langle 0 < m \rangle$]]

have $r \cdot t \leq p \ x$

by auto

have $r \cdot t = \varrho \ v$

proof (*rule ruler-eq-len*[*of* $\varrho \ v \ x \ r \cdot t$, *symmetric*])

have $|\varrho \ v| \leq |x|$

unfolding *conjug-len*[*OF conjug-facs, symmetric*] $\langle t \cdot r = \varrho \ u \rangle$ [*symmetric*]

unfolding $\langle (r \cdot t)^{\textcircled{a}} m \cdot r = x \rangle$ [*symmetric*] *pow-pos*[*OF* $\langle 0 < m \rangle$]

```

    lenmorph pow-len by auto
  from ruler-le[OF - - this, of v · x]
  show  $\varrho v \leq p x$ 
    using vx-xs prefI prefix-prefix primroot-pref v-nemp by metis
  show  $r \cdot t \leq p x$  by fact
  show  $|\varrho v| = |r \cdot t|$ 
    unfolding conjug-len[OF conjug-facs, symmetric, folded  $\langle t \cdot r = \varrho v \rangle$ ] lenmorph
  by simp
  qed

  then obtain l where  $(r \cdot t)^{\textcircled{a}} l = v$  and  $0 < l$ 
    using primroot-expE v-nemp by metis

  have  $(t \cdot r)^{\textcircled{a}} l = s$ 
    using vx-xs[folded  $\langle (r \cdot t)^{\textcircled{a}} m \cdot r = x \rangle$   $\langle (r \cdot t)^{\textcircled{a}} l = v \rangle$ , unfolded lassoc
  pows-comm[of - - m],
    unfolded rassoc cancel, unfolded shift-pow cancel].

  have  $r \cdot t \neq t \cdot r$ 
  proof
    assume  $r \cdot t = t \cdot r$ 
    hence aux:  $r \cdot (t \cdot r)^{\textcircled{a}} e = (t \cdot r)^{\textcircled{a}} e \cdot r$  for e
      by comparison
    have  $x \cdot (v \cdot u) = (v \cdot u) \cdot x$ 
      unfolding  $\langle (t \cdot r)^{\textcircled{a}} k = u \rangle$ [symmetric]  $\langle (r \cdot t)^{\textcircled{a}} l = v \rangle$ [symmetric]
      unfolding  $\langle (r \cdot t)^{\textcircled{a}} m \cdot r = x \rangle$ [symmetric] add-exps[symmetric]  $\langle r \cdot t = t \cdot$ 
  r> aux rassoc
      unfolding lassoc cancel-right add-exps[symmetric]
      by (simp add: add commute)
    thus False
      using vu-x-non-comm by blast
  qed

  show thesis
    using that[OF  $\langle (t \cdot r)^{\textcircled{a}} k = u \rangle$   $\langle (r \cdot t)^{\textcircled{a}} l = v \rangle$   $\langle (r \cdot t)^{\textcircled{a}} m \cdot r = x \rangle$ 
      uv-y[folded  $\langle (t \cdot r)^{\textcircled{a}} k = u \rangle$   $\langle (r \cdot t)^{\textcircled{a}} l = v \rangle$ ]  $\langle (r \cdot t)^{\textcircled{a}} k = p \rangle$   $\langle (t \cdot r)^{\textcircled{a}} l =$ 
  s>  $\langle r \cdot t \neq t \cdot r \rangle$ 
       $\langle 0 < k \rangle \langle 0 < m \rangle \langle 0 < l \rangle$ .
  qed

end

```

0.2.4 Locale: Extendable interpretation

Further specification follows from the assumption that the interpretation is extendable, that is, the covered $x \cdot x$ is a factor of a word composed of $\{x, y\}$. Namely, u and v are then conjugate by x .

locale *square-interp-ext* = *square-interp* +
assumes *p-extend*: $\exists pe. pe \in \{x, y\} \wedge p \leq_s pe$ **and**

s-extend: $\exists se. se \in \langle \{x, y\} \rangle \wedge s \leq_p se$

begin

lemma *s-pref-y*: $s \leq_p y$

proof–

obtain *sy ry eu ev ex*

where $(ry \cdot sy)^{\textcircled{a}} eu = u$ **and** $(sy \cdot ry)^{\textcircled{a}} ev = v$ **and**

$(sy \cdot ry)^{\textcircled{a}} eu = p$ **and** $(ry \cdot sy)^{\textcircled{a}} ev = s$ **and**

$(sy \cdot ry)^{\textcircled{a}} ex \cdot sy = x$ **and** $sy \cdot ry \neq ry \cdot sy$ **and**

$0 < eu$ **and** $0 < ev$ **and** $0 < ex$

using *bin-sq-interpE*.

obtain *se* **where** $se \in \langle \{x, y\} \rangle$ **and** $s \leq_p se$

using *s-extend* **by** *blast*

hence $se \neq \varepsilon$ **using** *s-nemp* **by** *force*

from $\langle (sy \cdot ry)^{\textcircled{a}} ex \cdot sy = x \rangle$

have $sy \cdot ry \leq_p x$

unfolding *pow-pos*[*OF* $\langle 0 < ex \rangle$] *rassoc* **by** *force*

have $x \leq_p se \vee y \leq_p se$

using $\langle se \neq \varepsilon \rangle$ *hull.cases*[*OF* $\langle se \in \langle \{x, y\} \rangle \rangle$, *of* $x \leq_p se \vee y \leq_p se$]

prefix-append *triv-pref* *two-elem-cases* **by** *blast*

moreover **have** $\neg x \leq_p se$

proof

assume $x \leq_p se$

from *ruler-eq-len*[*of* $sy \cdot ry se ry \cdot sy$, *OF* *pref-trans*[*OF* $\langle sy \cdot ry \leq_p x \rangle$ *this*]]

show *False*

using $\langle s \leq_p se \rangle$ [*folded* $\langle (ry \cdot sy)^{\textcircled{a}} ev = s \rangle$ [*unfolded* *pow-pos*[*OF* $\langle 0 < ev \rangle$]]]

$\langle sy \cdot ry \neq ry \cdot sy \rangle$ **by** (*force simp add: prefix-def*)

qed

ultimately **have** *y-pref-se*: $y \leq_p se$ **by** *blast*

from *ruler-le*[*OF* $\langle s \leq_p se \rangle$ *this*]

show $s \leq_p y$

using *lenarg*[*OF* $vx \cdot xs$] **unfolding** *uv-y*[*symmetric*] *lenmorph* **by** *linarith*

qed

lemma *rev-square-interp-ext*: *square-interp-ext* (*rev* *x*) (*rev* *y*) (*rev* *s*) (*rev* *p*) (*rev* (*map* *rev* *ws*))

proof–

interpret *i*: *square-interp* (*rev* *x*) (*rev* *y*) (*rev* *s*) (*rev* *p*) (*rev* (*map* *rev* *ws*))

using *rev-square-interp*.

show *?thesis*

proof

show $\exists pe. pe \in \langle \{rev\ x, rev\ y\} \rangle \wedge rev\ s \leq_s pe$

using *s-pref-y* **unfolding** *pref-rev-suf-iff* **by** *blast*

obtain *pe* **where** $pe \in \langle \{x, y\} \rangle$ **and** $p \leq_s pe$

using *p-extend* **by** *blast*
hence $\text{rev } pe \in \langle \{ \text{rev } x, \text{rev } y \} \rangle$
by (*simp add: rev-hull rev-in-conv*)
thus $\exists se. se \in \langle \{ \text{rev } x, \text{rev } y \} \rangle \wedge \text{rev } p \leq_p se$
using $\langle p \leq_s pe \rangle$ [*unfolded suf-rev-pref-iff prefix-def*] *rev-rev-ident* **by** *blast*
qed
qed

lemma *p-suf-y*: $p \leq_s y$

proof–

interpret *i*: *square-interp-ext* (*rev x*) (*rev y*) (*rev s*) (*rev p*) (*rev (map rev ws)*)
using *rev-square-interp-ext*.

from *i.s-pref-y* [*reversed*]

show $p \leq_s y$.

qed

theorem *bin-sq-interp-extE*: **obtains** $r \ t \ k \ m$ **where** $(r \cdot t)^{\textcircled{m}} \cdot r = x$ **and** $(t \cdot r)^{\textcircled{k}} \cdot (r \cdot t)^{\textcircled{k}} = y$
 $(r \cdot t)^{\textcircled{k}} = p$ **and** $(t \cdot r)^{\textcircled{k}} = s$ **and** $r \cdot t \neq t \cdot r$ **and** $u = s$ **and** $v = p$ **and**
 $|p| = |s|$ **and**
 $0 < k$ **and** $0 < m$

proof–

obtain $r \ t \ k \ k' \ m$

where $u: (t \cdot r)^{\textcircled{k}} = u$ **and** $v: (r \cdot t)^{\textcircled{k'}} = v$ **and**

$p: (r \cdot t)^{\textcircled{k}} = p$ **and** $s: (t \cdot r)^{\textcircled{k'}} = s$ **and**

$x: (r \cdot t)^{\textcircled{m}} \cdot r = x$ **and** *code*: $r \cdot t \neq t \cdot r$ **and**

$0 < k' \ 0 < m \ 0 < k$

using *bin-sq-interpE*.

have $|u \cdot v| = |s \cdot p|$

using *lenarg* [*OF px-xu, unfolded lenmorph*] *lenarg* [*OF vx-xs, unfolded lenmorph*]

by *simp*

hence $u \cdot v = s \cdot p$

unfolding *uv-y* **using** *s-pref-y p-suf-y* **by** (*auto simp add: prefix-def suffix-def*)

note $eq = \langle u \cdot v = s \cdot p \rangle$ [*unfolded* $\langle (t \cdot r)^{\textcircled{k}} = u \rangle$ [*symmetric*] $\langle (r \cdot t)^{\textcircled{k'}} = v \rangle$ [*symmetric*],

unfolded $\langle (t \cdot r)^{\textcircled{k'}} = s \rangle$ [*symmetric*] $\langle (r \cdot t)^{\textcircled{k}} = p \rangle$ [*symmetric*]]

from *pows-comm-comm* [*OF this*]

have $k = k'$

using $\langle r \cdot t \neq t \cdot r \rangle$ *eqd-eq(1)* [*OF - swap-len, of t r*] **by** *fastforce*

have $|p| = |s|$

using *lenarg* [*OF p*] *lenarg* [*OF s*] **unfolding** $\langle k = k' \rangle$ *pow-len lenmorph add. commute* [*of |r|*] **by** *fastforce*

thus *thesis*

using *that* [*OF x uv-y*] [*folded* $u \ v \ \langle k = k' \rangle$] $p \ s$ [*folded* $\langle k = k' \rangle$] *code* - - - $\langle 0 < k \ \langle 0 < m \rangle$] $u \ v \ p \ s$ **unfolding** $\langle k = k' \rangle$ **by** *argo*

qed

lemma *ps-len*: $|p| = |s|$ **and** *p-eq-v*: $p = v$ **and** *s-eq-u*: $s = u$

using *bin-sq-interp-extE* by *blast+*

lemma *v-x-x-u*: $v \cdot x = x \cdot u$
 using *vx-xs unfolding s-eq-u*.

lemma *sp-y*: $s \cdot p = y$
 using *p-eq-v s-eq-u uv-y* by *auto*

lemma *p-x-x-s*: $p \cdot x = x \cdot s$
 by (*simp add: px-xu s-eq-u*)

lemma *xy-root*: $x \cdot x \cdot y = (x \cdot p) \cdot (x \cdot p)$
 using *p-x-x-s sp-y* by *force*

theorem *sq-ext-interp*: $ws = [x, y, x] \ s \cdot p = y \ p \cdot x = x \cdot s$
 using *cover-xyx sp-y p-x-x-s*.

end

theorem *bin-sq-interpE*:
 assumes $x \cdot y \neq y \cdot x$ and *primitive* x and $|y| \leq |x|$ and $ws \in \text{lists } \{x, y\}$ and
 $\neg x \sim y$ and
 $p [x, x] \ s \sim_{\mathcal{D}} \ ws$
 obtains $r \ t \ m \ k \ l$ where $(r \cdot t)^{\textcircled{a}} \ m \cdot r = x$ and $(t \cdot r)^{\textcircled{a}} \ k \cdot (r \cdot t)^{\textcircled{a}} \ l = y$
 $(r \cdot t)^{\textcircled{a}} \ k = p$ and $(t \cdot r)^{\textcircled{a}} \ l = s$ and $r \cdot t \neq t \cdot r$ and $0 < k \ 0 < m \ 0 < l$
 using *square-interp.bin-sq-interpE[OF square-interp.intro, OF assms, of thesis]*.

theorem *bin-sq-interp*:
 assumes $x \cdot y \neq y \cdot x$ and *primitive* x and $|y| \leq |x|$ and $ws \in \text{lists } \{x, y\}$ and
 $\neg x \sim y$ and
 $p [x, x] \ s \sim_{\mathcal{D}} \ ws$
 shows $ws = [x, y, x]$
 using *square-interp.cover-xyx[OF square-interp.intro, OF assms]*.

theorem *bin-sq-interp-extE*:
 assumes $x \cdot y \neq y \cdot x$ and *primitive* x and $|y| \leq |x|$ and $ws \in \text{lists } \{x, y\}$ and
 $\neg x \sim y$ and
 $p [x, x] \ s \sim_{\mathcal{D}} \ ws$ and
p-extend: $\exists \ pe. \ pe \in \langle \{x, y\} \rangle \wedge p \leq_s \ pe$ and
s-extend: $\exists \ se. \ se \in \langle \{x, y\} \rangle \wedge s \leq_p \ se$
 obtains $r \ t \ m \ k$ where $(r \cdot t)^{\textcircled{a}} \ m \cdot r = x$ and $(t \cdot r)^{\textcircled{a}} \ k \cdot (r \cdot t)^{\textcircled{a}} \ k = y$
 $(r \cdot t)^{\textcircled{a}} \ k = p$ and $(t \cdot r)^{\textcircled{a}} \ k = s$ and $r \cdot t \neq t \cdot r$ and $0 < k$ and $0 < m$
 using *square-interp-ext.bin-sq-interp-extE[OF square-interp-ext.intro, OF square-interp.intro square-interp-ext-axioms.intro, OF assms, of thesis]*.

end

theory *Binary-Code-Imprimitive*

imports

Combinatorics- Words-Graph-Lemma.Glued-Codes

Binary-Square-Interpretation

begin

This theory focuses on the characterization of imprimitive words which are concatenations of copies of two words (forming a binary code). We follow the article [1] (mainly Théorème 2.1 and Lemme 3.1), while substantially optimizing the proof. See also [3] for an earlier result on this question, and [2] for another proof.

0.3 General primitivity not preserving codes

context *code*

begin

Two nontrivially conjugate elements generated by a code induce a disjoint interpretation.

lemma *shift-disjoint*:

assumes $ws \in \text{lists } \mathcal{C}$ **and** $ws' \in \text{lists } \mathcal{C}$ **and** $z \notin \langle \mathcal{C} \rangle$ **and** $z \cdot \text{concat } ws = \text{concat } ws' \cdot z$

$us \leq_p ws^{\textcircled{n}}$ **and** $vs \leq_p ws'^{\textcircled{n}}$

shows $z \cdot \text{concat } us \neq \text{concat } vs$

using $\langle z \notin \langle \mathcal{C} \rangle \rangle$

proof (*elim contrapos-nn*)

assume $z \cdot \text{concat } us = \text{concat } vs$

have $z \neq \varepsilon$

using $\langle z \notin \langle \mathcal{C} \rangle \rangle$ **by** *blast*

obtain us' **where** $ws^{\textcircled{n}} = us \cdot us'$

using *prefixE[OF* $\langle us \leq_p ws^{\textcircled{n}} \rangle$.

obtain vs' **where** $ws'^{\textcircled{n}} = vs \cdot vs'$

using *prefixE[OF* $\langle vs \leq_p ws'^{\textcircled{n}} \rangle$.

from *conjug-pow[OF* $\langle z \cdot \text{concat } ws = \text{concat } ws' \cdot z \rangle$ [*symmetric*], *symmetric*]

have $z \cdot \text{concat } (ws^{\textcircled{n}}) = \text{concat } (ws'^{\textcircled{n}}) \cdot z$

unfolding *concat-pow*.

from *this* [*unfolded* $\langle ws^{\textcircled{n}} = us \cdot us' \rangle$ $\langle ws'^{\textcircled{n}} = vs \cdot vs' \rangle$ *concat-morph rassoc*

$\langle z \cdot \text{concat } us = \text{concat } vs \rangle$ [*symmetric*] *cancel*]

have $\text{concat } vs' \cdot z = \text{concat } us'..$

show $z \in \langle \mathcal{C} \rangle$

proof (*rule stability*)

have $us \in \text{lists } \mathcal{C}$ **and** $us' \in \text{lists } \mathcal{C}$ **and** $vs \in \text{lists } \mathcal{C}$ **and** $vs' \in \text{lists } \mathcal{C}$

using $\langle ws \in \text{lists } \mathcal{C} \rangle$ $\langle ws' \in \text{lists } \mathcal{C} \rangle$ $\langle ws'^{\textcircled{n}} = vs \cdot vs' \rangle$ $\langle ws^{\textcircled{n}} = us \cdot us' \rangle$

by *inlists*

thus $z \cdot \text{concat } us \in \langle \mathcal{C} \rangle$ **and** $\text{concat } vs' \in \langle \mathcal{C} \rangle$ **and** $\text{concat } us \in \langle \mathcal{C} \rangle$ **and** $\text{concat } vs' \cdot z \in \langle \mathcal{C} \rangle$

unfolding $\langle \text{concat } vs' \cdot z = \text{concat } us' \rangle$ $\langle z \cdot \text{concat } us = \text{concat } vs \rangle$

by (*simp-all add: concat-in-hull'*)

qed
qed

This in particular yields a disjoint extendable interpretation of any prefix

lemma *shift-interp*:

assumes $ws \in \text{lists } \mathcal{C}$ **and** $ws' \in \text{lists } \mathcal{C}$ **and** $z \notin \langle \mathcal{C} \rangle$ **and**
 $\text{conjug: } z \cdot \text{concat } ws = \text{concat } ws' \cdot z$ **and** $|z| \leq |\text{concat } ws'|$
and $us \leq_p ws$ **and** $us \neq \varepsilon$

obtains $p \ s \ vs \ ps$ **where**

$p \ us \ s \sim_{\mathcal{D}} \ vs$ **and** $vs \in \text{lists } \mathcal{C}$
and $s \leq_p \text{concat } (us^{-1} \triangleright (ws \cdot ws))$ **and** $p \leq_s \text{concat } ws$ — extendable
and $ps \cdot vs \leq_p ws' \cdot ws'$ **and** $\text{concat } ps \cdot p = z$

proof–

have $ws' \cdot ws' \in \text{lists } \mathcal{C}$

using $\langle ws' \in \text{lists } \mathcal{C} \rangle$ **by** *inlists*

have $\text{concat } us \neq \varepsilon$

using $\langle us \neq \varepsilon \rangle$ **unfolding** *code-concat-eq-emp-iff*[*OF pref-in-lists*[*OF* $\langle us \leq_p ws \rangle$ $\langle ws \in \text{lists } \mathcal{C} \rangle$]].

have $|\text{concat } ws'| = |\text{concat } ws|$

using *lenarg*[*OF conjug, unfolded lenmorph*] **by** *linarith*

have $z \cdot \text{concat}(ws \cdot ws) = \text{concat}(ws' \cdot ws') \cdot z$

unfolding *rassoc concat-morph conjug*[*symmetric*] **unfolding** *lassoc cancel-right*

using *conjug*.

hence $\text{concat}(ws' \cdot ws') \leq_p z \cdot \text{concat}(ws \cdot ws)$

by *blast*

have $z \cdot \text{concat } ws \leq_p \text{concat}(ws' \cdot ws')$

unfolding *concat-morph conjug pref-cancel-conv* **using** *eq-le-pref*[*OF conjug* $\langle |z| \leq |\text{concat } ws'| \rangle$].

from *prefixE*[*OF pref-shorten*[*OF pref-concat-pref*[*OF* $\langle us \leq_p ws \rangle$] *this*], *unfolded rassoc*]

obtain su **where** *fac-u*[*symmetric*]: $\text{concat}(ws' \cdot ws') = z \cdot \text{concat } us \cdot su$.

from *obtain-fac-interp*[*OF fac-u* $\langle \text{concat } us \neq \varepsilon \rangle$]

obtain $ps \ ss' \ p \ s \ vs$ **where** $p \ (\text{concat } us) \ s \sim_{\mathcal{I}} \ vs$ **and**

$ps \cdot vs \cdot ss' = ws' \cdot ws'$ **and** $\text{concat } ps \cdot p = z$ **and** $s \cdot \text{concat } ss' = su$.

note *fac-interpD*[*OF* $\langle p \ (\text{concat } us) \ s \sim_{\mathcal{I}} \ vs \rangle$]

let $?ss = us^{-1} \triangleright (ws \cdot ws)$

have $us \cdot ?ss = ws \cdot ws$

using $\langle us \leq_p ws \rangle$ **by** *auto*

have $ps \cdot vs \leq_p ws' \cdot ws'$

unfolding $\langle ps \cdot vs \cdot ss' = ws' \cdot ws' \rangle$ [*symmetric*] *lassoc* **using** *triv-pref*.

hence $vs \in \text{lists } \mathcal{C}$

using $\langle ws' \in \text{lists } \mathcal{C} \rangle$

by *inlists*

have $s \leq_p \text{concat } ?ss$

using $\langle \text{concat } (ws' \cdot ws') \leq_p z \cdot \text{concat } (ws \cdot ws) \rangle$
unfolding $\text{arg-cong}[OF \langle ps \cdot vs \cdot ss' = ws' \cdot ws' \rangle, \text{of concat, symmetric}] \langle \text{concat } ps \cdot p = z \rangle [\text{symmetric}]$
 $\text{arg-cong}[OF \langle us \cdot ?ss = ws \cdot ws \rangle, \text{of concat, symmetric}]$
unfolding $\text{concat-morph rassoc pref-cancel-conv}$
 $\langle p \cdot \text{concat } us \cdot s = \text{concat } vs \rangle [\text{symmetric}]$
using append-prefixD **by** auto

have $|p| \leq |\text{concat } ws|$
using $\langle |z| \leq |\text{concat } ws'| \rangle [\text{folded lenarg}[OF \langle \text{concat } ps \cdot p = z \rangle], \text{unfolded } \langle |\text{concat } ws'| = |\text{concat } ws| \rangle]$
by simp

with $\text{eqd}[\text{reversed}, OF \text{conjug}[\text{folded } \langle \text{concat } ps \cdot p = z \rangle, \text{unfolded lassoc, symmetric}] \text{this}]$
have $p \leq_s \text{concat } ws$
by blast

have $\text{disjoint: } p \cdot \text{concat } us' \neq \text{concat } vs' \text{ if } us' \leq_p us \text{ vs}' \leq_p vs \text{ for } us' \text{ vs}'$
proof
have $us' \leq_p ws \cdot ws$
using $\langle us \leq_p ws \rangle \langle us' \leq_p us \rangle$ **by** auto
have $ps \cdot vs' \leq_p ws' \cdot ws'$
using $\langle vs' \leq_p vs \rangle \langle ps \cdot vs \leq_p ws' \cdot ws' \rangle$ $\text{pref-trans same-prefix-prefix}$ **by** metis
assume $p \cdot \text{concat } us' = \text{concat } vs'$
hence $z \cdot \text{concat } us' = \text{concat } (ps \cdot vs')$
unfolding $\text{concat-morph } \langle \text{concat } ps \cdot p = z \rangle [\text{symmetric}]$ rassoc cancel.
thus False
using $\text{shift-disjoint}[OF \langle ws \in \text{lists } \mathcal{C} \rangle \langle ws' \in \text{lists } \mathcal{C} \rangle \langle z \notin \langle \mathcal{C} \rangle \rangle$
 $\langle z \cdot \text{concat } ws = \text{concat } ws' \cdot z \rangle \langle us' \leq_p ws \cdot ws \rangle [\text{folded pow-two}] \langle ps \cdot vs' \leq_p ws' \cdot ws' \rangle [\text{folded pow-two}]$ **by** fast

qed
from $\text{disjoint}[of \varepsilon \varepsilon]$
have $p \neq \varepsilon$ **by** blast
have $s \neq \varepsilon$
using $\langle p \cdot \text{concat } us \cdot s = \text{concat } vs \rangle$ disjoint **by** auto

from $\text{disjoint-interpI}[OF \langle p (\text{concat } us) s \sim_{\mathcal{I}} vs \rangle]$ disjoint
have $p \text{ us } s \sim_{\mathcal{D}} vs$
by blast

from $\text{that}[OF \text{this } \langle vs \in \text{lists } \mathcal{C} \rangle$
 $\langle s \leq_p \text{concat } ?ss \rangle \langle p \leq_s \text{concat } ws \rangle \langle ps \cdot vs \leq_p ws' \cdot ws' \rangle \langle \text{concat } ps \cdot p = z \rangle]$
show thesis.

qed

The conditions are in particular met by imprimitivity witnesses

lemma $\text{imprim-witness-shift:}$

assumes $ws \in \text{lists } \mathcal{C}$ **and** $\text{primitive } ws$ **and** $\neg \text{primitive } (\text{concat } ws)$

obtains $z\ n$ **where** $\text{concat } ws = z^{\textcircled{n}}\ z \notin \langle C \rangle$ **and**
 $z \cdot \text{concat } ws = \text{concat } ws \cdot z$ **and** $|z| < |\text{concat } ws|$ **and** $2 \leq n$
proof–
have $\text{concat } ws \neq \varepsilon$
using $\langle \text{primitive } ws \rangle$ $\text{emp-concat-emp}'[OF \langle ws \in \text{lists } C \rangle]$ emp-not-prim **by**
 blast
obtain $z\ n$ **where** $[\text{symmetric}]$: $z^{\textcircled{n}} = \text{concat } ws$ **and** $2 \leq n$
using $\text{not-prim-primroot-expE}[OF \langle \neg \text{primitive } (\text{concat } ws) \rangle]$ **by** metis

hence $z \neq \varepsilon$
using $\langle \text{concat } ws \neq \varepsilon \rangle$ **by** force

have $z \notin \langle C \rangle$
proof
assume $z \in \langle C \rangle$
then obtain zs **where** $zs \in \text{lists } C$ **and** $\text{concat } zs = z$
using $\text{hull-concat-lists0}$ **by** blast
from $\text{is-code}[OF \langle ws \in \text{lists } C \rangle]$ $\text{pow-in-lists}[OF \langle zs \in \text{lists } C \rangle]$,
 $\text{unfolded concat-pow } \langle \text{concat } ws = z^{\textcircled{n}} \rangle \langle \text{concat } zs = z \rangle$, $\text{of } n]$
show False
using $\langle \text{primitive } ws \rangle$ $\langle 2 \leq n \rangle$ pow-nemp-imprim **by** blast
qed

have $|z| < |\text{concat } ws|$
unfolding $\text{lenarg}[OF \langle \text{concat } ws = z^{\textcircled{n}} \rangle]$, $\text{unfolded lenmorph pow-len}$
using $\text{nemp-len}[OF \langle z \neq \varepsilon \rangle]$ $\langle 2 \leq n \rangle$ **by** simp

from $\text{that}[OF \langle \text{concat } ws = z^{\textcircled{n}} \rangle \langle z \notin \langle C \rangle \rangle - \text{this } \langle 2 \leq n \rangle]$
show thesis
unfolding $\langle \text{concat } ws = z^{\textcircled{n}} \rangle$ pow-comm **by** blast

qed

end

0.4 Covered uniform square

lemma cover-xy-xxx : **assumes** $|x| = |y|$ **and** $p \cdot x \cdot y \cdot s = x \cdot x \cdot x$
shows $x = y$
using append-assoc $\text{assms}(1)$ $\text{assms}(2)$ eq-le-pref le-refl long-pref lq-triv prefI
 $\text{pref-comm-eq}'$ **by** metis

lemma cover-xy-yyy : **assumes** $|x| = |y|$ **and** eq : $p \cdot x \cdot y \cdot s = y \cdot y \cdot y$
shows $x = y$
using $\text{cover-xy-xxx}[\text{reversed}, \text{unfolded rassoc}, OF \langle |x| = |y| \rangle]$ $[\text{symmetric}]$ eq , sym-metric .

lemma cover-xy-xyy : **assumes** $|x| = |y|$ **and** $s \neq \varepsilon$ **and** eq : $p \cdot x \cdot y \cdot s = x \cdot x \cdot y$
shows $x = y$

proof-
have $|p| < |x|$
using $lenarg[OF\ eq]\ nemp-pos-len[OF\ \langle s \neq \varepsilon \rangle]$ **unfolding** $lenmorph$ **by** $linarith$
then obtain t **where** $x = p \cdot t$ **and** $t \neq \varepsilon$
using $eqd[OF\ eq]$ **by** $force$
from $eq[unfolding\ this\ rassoc\ cancel]$
have $p \cdot t = t \cdot p$
by $mismatch$
hence $x \leq_p t \cdot x$
unfolding x **by** $auto$
from $eq[unfolding\ x]$
have $y \leq_p t \cdot y$
using $\langle p \cdot t = t \cdot p \rangle\ \langle p \cdot t \cdot y \cdot s = t \cdot p \cdot t \cdot y \rangle$ $pref-cancel'\ suf-marker-per-root\ triv-pref$ **by** $metis$
show $x = y$
using $same-len-nemp-root-eq[OF\ per-rootI[OF\ \langle x \leq_p t \cdot x \rangle\ \langle t \neq \varepsilon \rangle]\ per-rootI[OF\ \langle y \leq_p t \cdot y \rangle\ \langle t \neq \varepsilon \rangle]\ \langle |x| = |y| \rangle]$.
qed

lemma $cover-xy-xyy$: **assumes** $|x| = |y|$ **and** $p \neq \varepsilon$ **and** $eq: p \cdot x \cdot y \cdot s = x \cdot y \cdot y$
shows $x = y$
using $cover-xy-xyy[reversed, unfolded\ rassoc, OF\ assms(1)[symmetric]\ assms(2)\ eq]..$

lemma $cover-xy-yyx$: **assumes** $|x| = |y|$ **and** $eq: p \cdot x \cdot y \cdot s = y \cdot y \cdot x$
shows $x = y$

proof-
have $|p| \leq |y|$
using $lenarg[OF\ eq]$ **unfolding** $lenmorph$ **by** $linarith$
then obtain t **where** $y = p \cdot t$
using $eqd[OF\ eq]$ **by** $force$
from $eqd-eq[OF\ -\ \langle |x| = |y| \rangle[unfolding\ y\ swap-len[of\ p]],\ unfolded\ rassoc]$ $eq[unfolding\ this\ rassoc\ cancel]$
have $x: x = t \cdot p$ **by** $blast$
from $eq[unfolding\ x\ y\ rassoc\ cancel]$
have $p \cdot t = t \cdot p$
by $mismatch$
thus $x = y$
unfolding $x\ y..$
qed

lemma $cover-xy-yxx$: **assumes** $|x| = |y|$ **and** $eq: p \cdot x \cdot y \cdot s = y \cdot x \cdot x$
shows $x = y$
using $cover-xy-yxx[reversed, unfolded\ rassoc, OF\ assms(1)[symmetric]\ eq]..$

lemma $cover-xy-xyx$: **assumes** $|x| = |y|$ **and** $p \neq \varepsilon$ **and** $s \neq \varepsilon$ **and** $eq: p \cdot x \cdot y \cdot s = x \cdot y \cdot x$
shows $\neg primitive\ (x \cdot y)$

proof

assume *primitive* $(x \cdot y)$
have $p \cdot (x \cdot y) \cdot (s \cdot y) = (x \cdot y) \cdot (x \cdot y)$
unfolding *lassoc eq*[*unfolded lassoc*].
from *prim-overlap-sqE*[*OF* \langle *primitive* $(x \cdot y)$ \rangle *this*]
show *False*
using $\langle p \neq \varepsilon \rangle \langle s \neq \varepsilon \rangle$ **by** *blast*

qed

lemma *cover-xy-yxy*: **assumes** $|x| = |y|$ **and** $p \neq \varepsilon$ **and** $\langle s \neq \varepsilon \rangle$ **and** *eq*: $p \cdot x \cdot y \cdot s = y \cdot x \cdot y$
shows \neg *primitive* $(x \cdot y)$
using *cover-xy-yxy*[*reversed, unfolded rassoc, OF assms(1)[symmetric] assms(3) assms(2) eq*].

theorem *uniform-square-interp*: **assumes** $x \cdot y \neq y \cdot x$ **and** $|x| = |y|$ **and** $vs \in \text{lists } \{x, y\}$

and $p \cdot (x \cdot y) \cdot s \sim_{\mathcal{I}} vs$ **and** $p \neq \varepsilon$

shows \neg *primitive* $(x \cdot y)$ **and** $vs = [x, y, x] \vee vs = [y, x, y]$

proof–

note *fac-interpD*[*OF* $\langle p \cdot (x \cdot y) \cdot s \sim_{\mathcal{I}} vs \rangle$]

have $vs \neq \varepsilon$

using $\langle p \cdot (x \cdot y) \cdot s = \text{concat } vs \rangle$ *assms(5)* **by** *force*

have $|p| < |x|$

using *prefix-length-less*[*OF* $\langle p < p \text{ hd } vs \rangle$] *lists-hd-in-set*[*OF* $\langle vs \neq \varepsilon \rangle \langle vs \in \text{lists } \{x, y\} \rangle$]

$\langle |x| = |y| \rangle$

by *fastforce*

have $|s| < |x|$

using *suffix-length-less*[*OF* $\langle s < s \text{ last } vs \rangle$] $\langle |x| = |y| \rangle$ *lists-hd-in-set*[*reversed, OF* $\langle vs \neq \varepsilon \rangle \langle vs \in \text{lists } \{x, y\} \rangle$]

by *fastforce*

have $|\text{concat } vs| = |x| * |vs|$

using *assms(2–3)*

proof (*induction vs*)

case (*Cons a vs*)

have $|a| = |x|$ **and** $|a \# vs| = \text{Suc } |vs|$ **and**

$|\text{concat } (a \# vs)| = |a| + |\text{concat } vs|$ **and** $|\text{concat } vs| = |x| * |vs|$

using $\langle a \# vs \in \text{lists } \{x, y\} \rangle \langle |x| = |y| \rangle$ *Cons.IH Cons.prem* **by** *auto*

then show *?case* **by** *force*

qed *simp*

note *leneq* = *lenarg*[*OF* $\langle p \cdot (x \cdot y) \cdot s = \text{concat } vs \rangle$, *unfolded this lenmorph* $\langle |x| = |y| \rangle$ *[symmetric]*]

hence $|x| * |vs| < |x| * 4$ **and** $2 * |x| < |x| * |vs|$

using $\langle |p| < |x| \rangle \langle |s| < |x| \rangle$ *nemp-pos-len*[*OF* $\langle p \neq \varepsilon \rangle$] **by** *linarith+*

hence $|vs| = 3$

by *force*

hence $s \neq \varepsilon$


```

using leneq ⟨|p| < |x|⟩ by force

have x ≠ y
  using assms(1) by blast
with ⟨|vs| = 3⟩ ⟨vs ∈ lists {x,y}⟩ ⟨p · (x · y) · s = concat vs⟩
have (¬ primitive (x·y)) ∧ (vs = [x,y,x] ∨ vs = [y,x,y])
proof(list-inspection, simp-all)
  assume p · x · y · s = x · x · x
  from cover-xy-xxx[OF ⟨|x| = |y|⟩ this]
  show False
    using ⟨x ≠ y⟩ by blast
next
  assume p · x · y · s = x · x · y
  from cover-xy-xyy[OF ⟨|x| = |y|⟩ ⟨s ≠ ε⟩ this]
  show False
    using ⟨x ≠ y⟩ by blast
next
  assume p · x · y · s = x · y · x
  from cover-xy-xyx[OF ⟨|x| = |y|⟩ ⟨p ≠ ε⟩ ⟨s ≠ ε⟩ this]
  show ¬ primitive (x · y)
    by blast
next
  assume p · x · y · s = x · y · y
  from cover-xy-xyy[OF ⟨|x| = |y|⟩ ⟨p ≠ ε⟩ this]
  show False
    using ⟨x ≠ y⟩ by blast
next
  assume p · x · y · s = y · x · x
  from cover-xy-yxx[OF ⟨|x| = |y|⟩ this]
  show False
    using ⟨x ≠ y⟩ by blast
next
  assume p · x · y · s = y · x · y
  from cover-xy-yxy[OF ⟨|x| = |y|⟩ ⟨p ≠ ε⟩ ⟨s ≠ ε⟩ this]
  show ¬ primitive (x · y)
    by blast
next
  assume p · x · y · s = y · y · x
  from cover-xy-yyx[OF ⟨|x| = |y|⟩ this]
  show False
    using ⟨x ≠ y⟩ by blast
next
  assume p · x · y · s = y · y · y
  from cover-xy-yyy[OF ⟨|x| = |y|⟩ this]
  show False
    using ⟨x ≠ y⟩ by blast
qed
thus ¬ primitive (x·y) vs = [x,y,x] ∨ vs = [y,x,y]
  by blast+

```

qed

0.4.1 Primitivity (non)preserving uniform binary codes

theorem *bin-uniform-prim-morph*:

assumes $x \cdot y \neq y \cdot x$ **and** $|x| = |y|$ **and** *primitive* $(x \cdot y)$
and $ws \in \text{lists } \{x, y\}$ **and** $2 \leq |ws|$
shows *primitive* $ws \longleftrightarrow$ *primitive* $(\text{concat } ws)$

proof (*standard*, *rule ccontr*)

assume $\langle \text{primitive } ws \rangle$ **and** $\langle \neg \text{primitive } (\text{concat } ws) \rangle$
from *bin-prim-long-pref*[*OF* $\langle ws \in \text{lists } \{x, y\} \rangle \langle \text{primitive } ws \rangle \langle 2 \leq |ws| \rangle$]
obtain ws' **where** $ws \sim ws'$ $[x, y] \leq_p ws'$.
have $ws' \in \text{lists } \{x, y\}$
using *conjug-in-lists'*[*OF* $\langle ws \sim ws' \rangle \langle ws \in \text{lists } \{x, y\} \rangle$].
have *primitive* ws'
using *prim-conjug*[*OF* $\langle \text{primitive } ws \rangle \langle ws \sim ws' \rangle$].
have $\neg \text{primitive } (\text{concat } ws')$
using *conjug-concat-prim-iff* $\langle \neg \text{primitive } (\text{concat } ws) \rangle \langle ws \sim ws' \rangle$ **by** *auto*
interpret *code* $\{x, y\}$
using *bin-code-code*[*OF* $\langle x \cdot y \neq y \cdot x \rangle$].

have $[x, y] \neq \varepsilon$ **by** *blast*

from *imprim-witness-shift*[*OF* $\langle ws' \in \text{lists } \{x, y\} \rangle \langle \text{primitive } ws' \rangle \langle \neg \text{primitive } (\text{concat } ws') \rangle$]

obtain z n **where** $\text{concat } ws' = z^{\textcircled{n}}$ $n \notin \{x, y\}$ $z \cdot \text{concat } ws' = \text{concat } ws' \cdot z$ $|z| < |\text{concat } ws'|$.

from *shift-interp*[*OF* $\langle ws' \in \text{lists } \{x, y\} \rangle \langle ws' \in \text{lists } \{x, y\} \rangle$ *this*(2–3) *less-imp-le*[*OF* *this*(4)] $\langle [x, y] \leq_p ws' \rangle \langle [x, y] \neq \varepsilon \rangle$]

obtain p s vs **where** $p [x, y] s \sim_{\mathcal{D}} vs$ $vs \in \text{lists } \{x, y\}$ $s \leq_p \text{concat } ([x, y]^{-1} \triangleright (ws' \cdot ws'))$
 $p \leq_s \text{concat } ws' \cdot ps \cdot vs \leq_p ws' \cdot ws' \text{concat } ps \cdot p = z$.

from *uniform-square-interp*(1)[*OF* $\langle x \cdot y \neq y \cdot x \rangle \langle |x| = |y| \rangle \langle vs \in \text{lists } \{x, y\} \rangle$ -
-]

$\langle \text{primitive } (x \cdot y) \rangle$ *disj-interpD*[*OF* *this*(1), *simplified*] *disj-interp-nemp*(1)[*OF* *this*(1)]

show *False* **by** *force*

qed (*simp add: prim-concat-prim*)

— A stronger version is implied by the following lemma.

lemma *bin-uniform-imprim*: **assumes** $x \cdot y \neq y \cdot x$ **and** $|x| = |y|$ **and** $\neg \text{primitive } (x \cdot y)$

shows *primitive* x

proof—

have $x \cdot y \neq \varepsilon$ **and** $x \neq \varepsilon$ **and** $y \neq \varepsilon$

using $\langle x \cdot y \neq y \cdot x \rangle$ **by** *blast+*

from *not-prim-expE*[*OF* $\langle \neg \text{primitive } (x \cdot y) \rangle \langle x \cdot y \neq \varepsilon \rangle$]

obtain z k **where** *primitive* z **and** $2 \leq k$ **and** $z^{\textcircled{k}} = x \cdot y$.

hence $0 < k$

by simp
from *split-pow*[*OF* $\langle z^{\textcircled{0}}k = x \cdot y \rangle$ [*symmetric*] $\langle 0 < k \rangle \langle y \neq \varepsilon \rangle$]
obtain $u \ v \ l \ m$ **where** [*symmetric*]: $z = u \cdot v$ **and** $v \neq \varepsilon$ $x = (u \cdot v)^{\textcircled{0}} l \cdot u$ $y = (v \cdot u)^{\textcircled{0}} m \cdot v$ $k = l + m + 1$.
have $u \cdot v \neq v \cdot u$
using $\langle x \cdot y \neq y \cdot x \rangle$ **unfolding** $\langle x = (u \cdot v)^{\textcircled{0}} l \cdot u \rangle \langle y = (v \cdot u)^{\textcircled{0}} m \cdot v \rangle$
shifts **unfolding** *add-exps*[*symmetric*] *add.commute*[*of m*] **by force**
have $u \neq \varepsilon$ **and** $v \neq \varepsilon$ **and** $u \neq v$
using $\langle u \cdot v \neq v \cdot u \rangle$ **by blast+**
have $m = l$ **and** $|u| = |v|$
using *almost-equal-equal*[*OF nemp-len*[*OF* $\langle u \neq \varepsilon \rangle$] *nemp-len*[*OF* $\langle v \neq \varepsilon \rangle$], *of l m*] *lenarg*[*OF* $\langle x = (u \cdot v)^{\textcircled{0}} l \cdot u \rangle$, *unfolded* $\langle |x| = |y| \rangle$, *unfolded* *lenarg*[*OF* $\langle y = (v \cdot u)^{\textcircled{0}} m \cdot v \rangle$]]
unfolding *lenmorph pow-len lenarg*[*OF* $\langle u \cdot v = z \rangle$, *symmetric*] **by algebra+**
from $\langle k = l + m + 1 \rangle$ [*folded Suc-eq-plus1*, *symmetric*]
have $l \neq 0$
using $\langle 2 \leq k \rangle$ [*folded* $\langle \text{Suc}(l+m) = k \rangle$, *unfolded* $\langle m = l \rangle$] **by force**
let $?w = [u, v]^{\textcircled{0}} l \cdot [u]$
have $?w \in \text{lists } \{u, v\}$
by (*induct l*, *simp-all*)
have $2 \leq |?w|$
using $\langle l \neq 0 \rangle$ **unfolding** *lenmorph pow-len* **by fastforce**
have *concat* $?w = x$
using $\langle x = (u \cdot v)^{\textcircled{0}} l \cdot u \rangle$ **by simp**
from *bin-uniform-prim-morph*[*OF* $\langle u \cdot v \neq v \cdot u \rangle \langle |u| = |v| \rangle \langle \text{primitive } z \rangle$] [*folded* $\langle u \cdot v = z \rangle \langle ?w \in \text{lists } \{u, v\} \rangle \langle 2 \leq |?w| \rangle$]
show *primitive x*
unfolding $\langle \text{concat } ?w = x \rangle$ **using** *alternate-prim*[*OF* $\langle u \neq v \rangle$] **by blast**
qed

theorem *bin-uniform-prim-morph'*:

assumes $x \cdot y \neq y \cdot x$ **and** $|x| = |y|$ **and** *primitive* $(x \cdot y) \vee \neg \text{primitive } x \vee \neg \text{primitive } y$
and $ws \in \text{lists } \{x, y\}$ **and** $2 \leq |ws|$
shows *primitive ws* \longleftrightarrow *primitive (concat ws)*
using *bin-uniform-prim-morph*[*OF* *assms*(1-2) - *assms*(4-5)] *bin-uniform-imprim*[*OF* *assms*(1-2)]
bin-uniform-imprim[*OF* *assms*(1-2)] [*symmetric*], *unfolded* *conjug-prim-iff'*[*of y*]]
assms(3) **by blast**

0.5 The main theorem

0.5.1 Imprimitive words with single y

If the shorter word occurs only once, the result is straightforward from the parametric solution of the Lyndon-Schutzenberger equation.

lemma *bin-imprim-single-y*:
assumes *non-comm*: $x \cdot y \neq y \cdot x$ **and**
 $ws \in \text{lists } \{x,y\}$ **and**
 $|y| \leq |x|$ **and**
 $2 \leq \text{count-list } ws \ x$ **and**
 $\text{count-list } ws \ y < 2$ **and**
primitive ws **and**
 $\neg \text{primitive } (\text{concat } ws)$
shows $ws \sim [x,x,y]$ **and** *primitive* x **and** *primitive* y
proof–
have $x \neq y$
using *non-comm* **by** *blast*
have $\text{count-list } ws \ y \neq 0$
proof
assume $\text{count-list } ws \ y = 0$
from *bin-lists-count-zero'*[*OF* $\langle ws \in \text{lists } \{x,y\} \rangle$ *this*]
have $ws \in \text{lists } \{x\}$.
from *prim-exp-one*[*OF* $\langle \text{primitive } ws \rangle$ *sing-lists-exp-count*[*OF* *this*]]
show *False*
using $\langle 2 \leq \text{count-list } ws \ x \rangle$ **by** *simp*
qed
hence $\text{count-list } ws \ y = 1$
using $\langle \text{count-list } ws \ y < 2 \rangle$ **by** *linarith*

from *this bin-count-one-conjug*[*OF* $\langle ws \in \text{lists } \{x,y\} \rangle$ - *this*]
have $ws \sim [x]^{\textcircled{1}} \text{count-list } ws \ x \cdot [y]$
using *non-comm (1)* **by** *metis*
from *conjug-concat-prim-iff*[*OF* *this*]
have $\neg \text{primitive } (x^{\textcircled{1}} (\text{count-list } ws \ x) \cdot y)$
using $\langle \neg \text{primitive } (\text{concat } ws) \rangle$ **by** *simp*

from *not-prim-primroot-expE*[*OF* *this*]
obtain $z \ l$ **where** [*symmetric*]: $z^{\textcircled{1}} l = x^{\textcircled{1}} (\text{count-list } ws \ x) \cdot y^{\textcircled{1}} 1$ **and** $2 \leq l$
unfolding *pow-1*.

interpret *LS-len-le* $x \ y \ \text{count-list } ws \ x \ 1 \ l \ z$
by (*unfold-locales*)
 $(\text{use } \langle 2 \leq \text{count-list } ws \ x \rangle \langle x \cdot y \neq y \cdot x \rangle \langle |y| \leq |x| \rangle$
 $\langle x^{\textcircled{1}} \text{count-list } ws \ x \cdot y^{\textcircled{1}} 1 = z^{\textcircled{1}} l \rangle \langle 2 \leq l \rangle$ **in** *force*) $+$

from *case-j2k1*[*OF* $\langle 2 \leq \text{count-list } ws \ x \rangle$ *refl*]
have *primitive* x **and** *primitive* y **and** $\text{count-list } ws \ x = 2$ **by** *blast+*

with $\langle ws \sim [x]^{\textcircled{1}} \text{count-list } ws \ x \cdot [y] \rangle$ [*unfolded this(3) pow-two append-Cons append-Nil*]
show *primitive* x **and** *primitive* y **and** $ws \sim [x,x,y]$
by *simp-all*

qed

0.5.2 Conjugate words

lemma *bin-imprim-not-conjug*:

assumes $ws \in lists\ \{x,y\}$ **and**

$x \cdot y \neq y \cdot x$ **and**

$2 \leq |ws|$ **and**

primitive ws **and**

$\neg primitive\ (concat\ ws)$

shows $\neg x \sim y$

proof

assume $x \sim y$

hence $|x| = |y|$ **by force**

from *bin-uniform-prim-morph*[*OF* $\langle x \cdot y \neq y \cdot x \rangle$ *this -* $\langle ws \in lists\ \{x,y\} \rangle$ $\langle 2 \leq |ws| \rangle$]

have $\neg primitive\ (x \cdot y)$

using $\langle primitive\ ws \rangle$ $\langle \neg primitive\ (concat\ ws) \rangle$ **by blast**

from *Lyndon-Schutzenberger-conjug*[*OF* $\langle x \sim y \rangle$ *this*]

show *False*

using $\langle x \cdot y \neq y \cdot x \rangle$ **by blast**

qed

0.5.3 Square factor of the longer word and both words primitive (was all_assms)

The main idea of the proof is as follows: Imprimitivity of the concatenation yields (at least) two overlapping factorizations into $\{x, y\}$. Due to the presence of the square $x \cdot x$, these two can be synchronized, which yields that the situation coincides with the canonical form.

lemma *bin-imprim-primitive*:

assumes $x \cdot y \neq y \cdot x$

and *primitive x* **and** *primitive y*

and $|y| \leq |x|$

and $ws \in lists\ \{x, y\}$

and *primitive ws* **and** $\neg primitive\ (concat\ ws)$

and $[x, x] \leq_f ws \cdot ws$

shows $ws \sim [x, x, y]$

proof–

— Preliminaries

have $x \neq y$

using *assms(1)* **by blast**

have $|ws| \neq 1$

using *len-one-concat-in*[*OF* $\langle ws \in lists \{x, y\} \rangle \langle \neg primitive (concat ws) \rangle$
 $\langle primitive x \rangle \langle primitive y \rangle$
by *blast*
with *prim-nemp*[*OF* $\langle primitive ws \rangle$, *THEN* *nemp-le-len*]
have $2 \leq |ws|$
by *auto*
hence $|[x, x]| \leq |ws|$
by *force*
have $\neg x \sim y$
by (*rule bin-imprim-not-conjug*) *fact+*
have *primitive* $[x, x, y]$
using $\langle x \neq y \rangle$ **by** *primitivity-inspection*
have *concat* $[x, x] = x \cdot x$
by *simp*
interpret *xy*: *binary-code* $x y$
using $\langle x \cdot y \neq y \cdot x \rangle$ **by** (*unfold-locales*)

— Rotate *ws* in order to obtain a list with a prefix $[x \cdot x]$

obtain *ws'* **where** $ws \sim ws' [x, x] \leq_p ws'$
using *rotate-into-pos-sq*[*of* $\varepsilon [x, x]$ - *thesis*, *unfolded emp-simps*, *OF* $\langle [x, x] \leq_f$
 $ws \cdot ws \rangle$
 $le0 \langle |[x, x]| \leq |ws| \rangle$] **by** *blast*

have $ws' \in lists \{x, y\}$ **and** *primitive* ws' **and** $\neg primitive (concat ws')$
using *conjug-in-lists'*[*OF* $\langle ws \sim ws' \rangle \langle ws \in lists \{x, y\} \rangle$]
 $prim-conjug$ [*OF* $\langle primitive ws \rangle \langle ws \sim ws' \rangle$]
 $\langle \neg primitive (concat ws) \rangle$ [*unfolded conjug-concat-prim-iff*[*OF* $\langle ws \sim ws' \rangle$]].
have $2 \leq |ws'|$ **and** $[x, x] \neq \varepsilon$ **and** $ws' \neq \varepsilon$
using $\langle [x, x] \leq_p ws' \rangle$ **unfolding** *prefix-def* **by** *auto*
have *concat* $ws' \neq \varepsilon$
using $\langle primitive x \rangle \langle [x, x] \leq_p ws' \rangle$ **by** (*fastforce simp add: prefix-def*)
have $ws' \cdot ws' \cdot ws' \in lists \{x, y\}$ **and** $ws' \cdot ws' \in lists \{x, y\}$
using $\langle ws' \in lists \{x, y\} \rangle$ **by** *inlists*

— The core of the proof

have $ws' = [x, x, y]$
proof(*rule ccontr*)
assume $ws' \neq [x, x, y]$

from *xy.imprim-witness-shift*[*OF* $\langle ws' \in lists \{x, y\} \rangle \langle primitive ws' \rangle \langle \neg primitive$
 $(concat ws') \rangle$]
obtain $z n$ **where** *con-ws*: *concat* $ws' = z^{\textcircled{a}} n$ **and** $z \notin \langle \{x, y\} \rangle$ **and** $z \cdot concat$
 $ws' = concat ws' \cdot z$
and $|z| < |concat ws'|$ **and** $2 \leq n$.
have $0 < n$
using $\langle 2 \leq n \rangle$ **by** *simp*
from *xy.shift-interp*[*OF* $\langle ws' \in lists \{x, y\} \rangle \langle ws' \in lists \{x, y\} \rangle \langle z \notin \langle \{x, y\} \rangle \rangle \langle z$
 $\cdot concat ws' = concat ws' \cdot z \rangle$
 $less-imp-le$ [*OF* $\langle |z| < |concat ws'| \rangle \langle [x, x] \leq_p ws' \rangle \langle [x, x] \neq \varepsilon \rangle$]

obtain $p \ s \ vs \ ps$ **where** $dis: p \ [x,x] \ s \ \sim_{\mathcal{D}} \ vs$ **and** $\langle vs \in lists \ \{x, y\} \rangle$ **and**
 $s \leq_p \ concat \ ([x,x]^{-1}) \langle ws' \cdot ws' \rangle$ **and** $p \leq_s \ concat \ ws'$ **and** $ps \cdot vs \leq_p \ ws' \cdot ws'$
and $concat \ ps \cdot p = z$.

from $disj\text{-}interp\text{-}nemp(1)[OF \ this(1)]$
have $p \neq \varepsilon$ **by** $simp$

have $p \cdot concat \ p1 \neq \ concat \ p2$ **if** $p1 \leq_p \ [x, x]$ **and** $p2 \leq_p \ vs$ **for** $p1 \ p2$
using $\langle p \ [x,x] \ s \ \sim_{\mathcal{D}} \ vs \rangle \ disj\text{-}interpD1$ **that** **by** $blast$

have $ps \in lists \ \{x,y\}$
using $\langle ps \cdot vs \leq_p \ ws' \cdot ws' \rangle \ \langle ws' \in lists \ \{x,y\} \rangle \ \langle ws' \cdot ws' \in lists \ \{x, y\} \rangle$
 $append\text{-}prefixD \ pref\text{-}in\text{-}lists$ **by** $metis$
have $vs \in lists \ \{x,y\}$
using $\langle ws' \in lists \ \{x,y\} \rangle \ pref\text{-}in\text{-}lists[OF \ \langle ps \cdot vs \leq_p \ ws' \cdot ws' \rangle]$ **by** $inlists$
have $[x,x]^{-1} \langle ws' \cdot ws' \rangle \in lists \ \{x,y\}$
using $\langle ws' \in lists \ \{x,y\} \rangle$ **by** $inlists$
have $p \ x \cdot x \ s \ \sim_{\mathcal{I}} \ vs$
using $disj\text{-}interpD[OF \ \langle p \ [x,x] \ s \ \sim_{\mathcal{D}} \ vs \rangle]$ **by** $simp$

interpret $square\text{-}interp\text{-}ext \ x \ y \ p \ s \ vs$

proof ($rule \ square\text{-}interp\text{-}ext.intro[OF \ square\text{-}interp.intro, \ unfolded \ square\text{-}interp\text{-}ext\text{-}axioms\text{-}def]$)

show $(\exists \ pe. \ pe \in \langle \{x, y\} \rangle \wedge p \leq_s \ pe) \wedge (\exists \ se. \ se \in \langle \{x, y\} \rangle \wedge s \leq_p \ se)$

using $\langle s \leq_p \ concat \ ([x,x]^{-1}) \langle ws' \cdot ws' \rangle \rangle \ \langle p \leq_s \ concat \ ws' \rangle$

$\langle [x, x]^{-1} \langle ws' \cdot ws' \rangle \in lists \ \{x, y\} \rangle \ \langle ws' \in lists \ \{x, y\} \rangle \ concat\text{-}in\text{-}hull'$ **by**

$meson$

qed $fact+$

— Establishing the connection between $ws' = [x,x,y]$ and $z = xp$.

define xp **where** $xp = x \cdot p$

have $concat \ [x,x,y] = xp \cdot xp$
by ($simp \ add: \ xxy\text{-}root \ xp\text{-}def$)

hence $ws' \cdot [x,x,y] \neq [x,x,y] \cdot ws'$

using $comm\text{-}prim[OF \ \langle primitive \ ws' \rangle \ \langle primitive \ [x,x,y] \rangle \ \langle ws' \neq [x,x,y] \rangle]$ **by**
 $force$

have $z \cdot xp \neq xp \cdot z$

proof

assume $z \cdot xp = xp \cdot z$

from $comm\text{-}add\text{-}exp[symmetric, \ OF \ this[symmetric], \ of \ 2,$

$THEN \ comm\text{-}add\text{-}exp, \ of \ n, \ unfolded \ pow\text{-}two]$

have $z^{\otimes n} \cdot xp \cdot xp = xp \cdot xp \cdot z^{\otimes n}$

unfolding $rassoc$.

hence $concat \ ws' \cdot concat \ [x,x,y] = concat \ [x,x,y] \cdot concat \ ws'$

unfolding $con\text{-}ws \ \langle concat \ [x,x,y] = xp \cdot xp \rangle \ rassoc$ **by** $simp$

from $xy.is\text{-}code[OF \ - \ - \ this[folded \ concat\text{-}morph]]$

```

have  $ws' \cdot [x, x, y] = [x,x,y] \cdot ws'$ 
  using append-in-lists  $\langle ws' \in lists \{x,y\} \rangle$  by simp
thus False
  using  $\langle ws' \cdot [x, x, y] \neq [x,x,y] \cdot ws' \rangle$  by fastforce
qed

```

```

then interpret binary-code  $z \ xp$ 
  by (unfold-locales)

```

```

have  $\neg concat (ws' \cdot [x, x, y]) \bowtie concat ([x, x, y] \cdot ws')$ 
proof (rule notI)
  assume  $concat (ws' \cdot [x, x, y]) \bowtie concat ([x, x, y] \cdot ws')$ 
  from comm-comp-eq[OF this[unfolded concat-morph], unfolded  $\langle concat [x,x,y]$ 
=  $xp \cdot xp \rangle$  con-ws]
  have  $z^{\textcircled{0}} n \cdot xp^{\textcircled{0}} Suc(Suc\ 0) = xp^{\textcircled{0}} Suc(Suc\ 0) \cdot z^{\textcircled{0}} n$ 
  unfolding pow-Suc pow-zero emp-simps rassoc.
  from comm-drop-exps[OF this]
  show False
  using  $\langle z \cdot xp \neq xp \cdot z \rangle \langle 2 \leq n \rangle$  by force
qed

```

- How the xp/z mismatch is reflected by mismatch in lists x,y ?
- Looking at the first occurrence of z :

```

define lcp-ws where  $lcp-ws = ws' \cdot [x,x,y] \wedge_p [x,x,y] \cdot ws'$ 

```

```

have  $lcp-ws \in lists \{x,y\}$ 
  unfolding lcp-ws-def by inlists

```

```

have lcp-xp-z:  $concat (ws' \cdot [x,x,y]) \wedge_p concat ([x,x,y] \cdot ws') = bin-lcp\ z\ (x \cdot p)$ 
unfolding concat-morph con-ws  $\langle concat [x,x,y] = xp \cdot xp \rangle$  add-exps[symmetric]
  using bin-lcp-pows[OF  $\langle 0 < n \rangle$ , of 2]
  unfolding pow-two pow-pos[OF  $\langle 0 < n \rangle$ ] rassoc xp-def by force

```

```

have  $(concat\ lcp-ws) \cdot bin-lcp\ x\ y = bin-lcp\ z\ (x \cdot p)$ 
proof (rule xy.bin-code-lcp-concat[OF - -  $\langle \neg concat (ws' \cdot [x, x, y]) \bowtie concat$ 
 $([x, x, y] \cdot ws') \rangle$ , folded lcp-ws-def, unfolded lcp-xp-z, symmetric])
  show  $ws' \cdot [x, x, y] \in lists \{x, y\}$  and  $[x, x, y] \cdot ws' \in lists \{x, y\}$ 
  by inlists
qed

```

- Looking at the second occurrence of z :

```

define  $ws''$  where  $ws'' = ps \cdot [x,y]$ 
define lcp-ws' where  $lcp-ws' = ws' \cdot ws'' \wedge_p ws'' \cdot ws'$ 

```

```

have  $lcp-ws' \in lists \{x,y\}$ 
  unfolding lcp-ws'-def
  using  $\langle ps \in lists \{x, y\} \rangle \langle ws' \in lists \{x, y\} \rangle$  ws''-def by inlists

```


have $\text{concat } ws'' = z \cdot xp$
unfolding $ws''\text{-def } xp\text{-def}$ **using** $\langle \text{concat } ps \cdot p = z \rangle$ *xy-root* **by** *fastforce*

have $ws' \cdot ws'' \neq ws'' \cdot ws'$
proof
assume $ws' \cdot ws'' = ws'' \cdot ws'$
from *arg-cong*[*OF this, of concat, unfolded concat-morph con-ws*
 $\langle \text{concat } ws'' = z \cdot xp \rangle$,
unfolded lassoc pow-comm, unfolded rassoc cancel]
show *False*
using $\langle z \cdot xp \neq xp \cdot z \rangle$ *comm-drop-exp'*[*OF - $\langle 0 < n \rangle$*] **by** *blast*
qed

have
 $\text{lcp-}xp\text{-}z'$: $\text{concat } (ws' \cdot ws'') \wedge_p \text{concat } (ws'' \cdot ws') = z \cdot \text{bin-lcp } z (x \cdot p)$
unfolding *concat-morph con-ws* $\langle \text{concat } ws'' = z \cdot xp \rangle$ *pow-Suc*
unfolding *lcp-ext-left[symmetric]* *bin-lcp-def shifts*
unfolding *rassoc lcp-ext-left cancel*
using *bin-lcp-pows*[*OF $\langle 0 < n \rangle$, of $1 \in z^{\otimes(n-1)}$, unfolded pow-1, folded*
pow-pos[*OF $\langle 0 < n \rangle$*]]
unfolding *bin-lcp-def xp-def rassoc emp-simps* **by** *linarith*

have $z \cdot \text{bin-lcp } z (x \cdot p) = \text{concat } (\text{lcp-}ws')$ $\cdot \text{bin-lcp } x y$
unfolding *lcp-}xp\text{-}z'*[*symmetric*] *lcp-}ws'\text{-def}*
proof (*rule xy.bin-code-lcp-concat'*)
show $ws' \cdot ws'' \in \text{lists } \{x, y\}$
unfolding $ws''\text{-def}$ **using** $\langle ws' \cdot ws' \cdot ws' \in \text{lists } \{x, y\} \rangle$ $\langle ps \in \text{lists } \{x, y\} \rangle$
by *inlists*
thus $ws'' \cdot ws' \in \text{lists } \{x, y\}$
by *inlists*
show $\neg \text{concat } (ws' \cdot ws'') \bowtie \text{concat } (ws'' \cdot ws')$
unfolding *concat-morph con-ws* $\langle \text{concat } ws'' = z \cdot xp \rangle$ *pow-pos*[*OF $\langle 0 < n \rangle$*]
unfolding *rassoc comp-cancel*
unfolding *lassoc pow-pos*[*OF $\langle 0 < n \rangle$, symmetric*] *pow-pos'*[*OF $\langle 0 < n \rangle$,*
symmetric]
comm-comp-eq-conv
using *comm-drop-exp'*[*OF - $\langle 0 < n \rangle$, of $z \ n \ xp$*] *non-comm* **by** *argo*
qed

have $\text{concat } \text{lcp-}ws' = z \cdot \text{concat } \text{lcp-}ws$
unfolding *cancel-right*[*of concat lcp-}ws' bin-lcp }x y z \cdot \text{concat } \text{lcp-}ws, *symmetric*]
unfolding *rassoc*[*of z*] $\langle \text{concat } (\text{lcp-}ws) \cdot \text{bin-lcp } x y = \text{bin-lcp } z (x \cdot p) \rangle$ $\langle z \cdot$
 $\text{bin-lcp } z (x \cdot p) = \text{concat } (\text{lcp-}ws') \cdot \text{bin-lcp } x y \rangle$.*

have $\text{lcp-}ws \leq_p ws' \cdot [x, x, y]$
unfolding *lcp-}ws\text{-def}* **using** *longest-common-prefix-prefix1*.
have $\text{lcp-}ws \neq ws' \cdot [x, x, y]$
unfolding *lcp-}ws\text{-def lcp-}pref\text{-conv}*

```

    using ⟨ws' · [x, x, y] ≠ [x, x, y] · ws'⟩ pref-comm-eq by blast
  have lcp-ws ≤p ws' · [x,x]
    using spref-butlast-pref[OF ⟨lcp-ws ≤p ws' · [x,x,y]⟩ ⟨lcp-ws ≠ ws' · [x,x,y]⟩]
    unfolding butlast-append by simp
  from prefixE[OF pref-prolong[OF this ⟨[x,x] ≤p ws'⟩]]
  obtain ws''1 where ws' · ws' · ws' = lcp-ws · ws''1 using rassoc by metis

  have ws' · ps · [x,y] ≤p ws' · ps · [x,y,x]
    by simp
  from pref-trans[OF pref-trans[OF longest-common-prefix-prefix1 this]]
  have lcp-ws' ≤p ws' · ws' · ws'
    unfolding lcp-ws'-def ws''-def using ⟨ps · vs ≤p ws' · ws'⟩[unfolded cover-xyx,
unfolded pref-cancel-conv]
    unfolding pref-cancel-conv[symmetric, of ps · [x,y,x] ws' · ws' ws'] by blast
  from prefixE[OF this]
  obtain ws''2 where ws' · ws' · ws' = lcp-ws' · ws''2.

  have concat lcp-ws' · concat ws''1 = z · concat(lcp-ws) · concat ws''1
    unfolding lassoc ⟨concat lcp-ws' = z · concat lcp-ws⟩..
  also have ... = z · concat (ws' · ws' · ws')
    unfolding rassoc ⟨ws' · ws' · ws' = lcp-ws · ws''1⟩ concat-morph..
  also have ... = concat (ws' · ws' · ws') · z
    unfolding concat-morph con-ws add-exps[symmetric]
    pow-Suc[symmetric] pow-Suc'[symmetric]..
  also have ... = concat lcp-ws' · concat ws''2 · z
    unfolding ⟨ws' · ws' · ws' = lcp-ws' · ws''2⟩ concat-morph rassoc..
  finally have concat ws''1 = concat ws''2 · z
    unfolding cancel.

  from xy.stability[of concat ws''2 concat lcp-ws z,
    folded ⟨concat ws''1 = concat ws''2 · z⟩ ⟨concat lcp-ws' = z · concat lcp-ws⟩]
  have z ∈ ⟨{x, y}⟩
    using ⟨ws' · ws' · ws' = lcp-ws · ws''1⟩ ⟨ws' · ws' · ws' = lcp-ws' · ws''2⟩ ⟨ws'
    · ws' · ws' ∈ lists {x, y}⟩
    append-in-lists-dest append-in-lists-dest' concat-in-hull' by metis
  thus False
    using ⟨z ∉ ⟨{x,y}⟩⟩ by blast
  qed
  thus ws ~ [x, x, y]
    using ⟨ws ~ ws'⟩ by blast
  qed

```

0.5.4 Obtaining primitivity with two squares (refining)

lemma *bin-imprim-both-squares-prim*:

```

assumes x · y ≠ y · x
  and ws ∈ lists {x, y}
  and primitive ws and ¬ primitive (concat ws)
  and [x, x] ≤f ws · ws

```

```

    and [y, y] ≤f ws · ws
    and primitive x and primitive y
  shows False
proof-
  have x ≠ y using ⟨x · y ≠ y · x⟩
  by blast
  from bin-imprim-primitive[OF ⟨x · y ≠ y · x⟩ ⟨primitive x⟩ ⟨primitive y⟩
    - ⟨ws ∈ lists {x,y}⟩ ⟨primitive ws⟩ ⟨¬ primitive (concat ws)⟩ ⟨[x, x] ≤f ws ·
ws⟩]
    bin-imprim-primitive[OF ⟨x · y ≠ y · x⟩[symmetric] ⟨primitive y⟩ ⟨primitive x⟩
    - ⟨ws ∈ lists {x,y}⟩[unfolded insert-commute[of x]] ⟨primitive ws⟩ ⟨¬ primitive
(concat ws)⟩
    ⟨[y, y] ≤f ws · ws⟩]
  have ws ~ [x, x, y] ∨ ws ~ [y, y, x]
  using ⟨x · y ≠ y · x⟩
  by force
  hence |ws| = 3
  using conjug-len by force
  note[simp] = sublist-code(3)
  from ⟨|ws| = 3⟩ ⟨ws ∈ lists {x,y}⟩ ⟨x ≠ y⟩
    ⟨[x, x] ≤f ws · ws⟩ ⟨[y, y] ≤f ws · ws⟩
  show False
  by list-inspection simp-all
qed

```

lemma bin-imprim-both-squares:

```

  assumes x · y ≠ y · x
  and ws ∈ lists {x, y}
  and primitive ws and ¬ primitive (concat ws)
  and [x, x] ≤f ws · ws
  and [y, y] ≤f ws · ws
  shows False
proof (rule bin-imprim-both-squares-prim)
  have x ≠ ε and y ≠ ε and x ≠ y
  using ⟨x · y ≠ y · x⟩ by blast+
  let ?R = λ x. [⊘ x]⊘(e⊘ x)
  define ws' where ws' = concat (map ?R ws)
  show ⊘ x · ⊘ y ≠ ⊘ y · ⊘ x
  using ⟨x · y ≠ y · x⟩[unfolded comp-primroot-conv'[of x y]].
  have [simp]: a = x ∨ a = y ⇒ [⊘ a]⊘ e⊘ a ∈ lists {⊘ x, ⊘ y} for a
  using insert-iff sing-pow-lists[of - {⊘ x, ⊘ y}] by metis
  show ws' ∈ lists {⊘ x, ⊘ y}
  unfolding ws'-def using ⟨ws ∈ lists {x,y}⟩
  by (induction ws, simp-all)

```

— The primitivity of ws' is obtained from the fact that the decompositions into roots is a primitive morphism

```

interpret binary-code x y
using ⟨x · y ≠ y · x⟩ by unfold-locales

```

```

note[simp] = sublist-code(3)
have |ws| ≤ 3 ⇒ ws ∈ lists {x,y} ⇒ x ≠ y ⇒ [x, x] ≤f ws · ws ⇒ [y, y]
≤f ws · ws ⇒ False
  by list-inspection simp-all
from this[OF - ⟨ws ∈ lists {x,y}⟩ ⟨x ≠ y⟩ ⟨[x, x] ≤f ws · ws⟩ ⟨[y, y] ≤f ws · ws⟩]
  roots-prim-morph[OF ⟨ws ∈ lists {x,y}⟩ - ⟨primitive ws⟩]
show primitive ws'
  unfolding ws'-def by fastforce

show ¬ primitive (concat ws^)
  unfolding ws'-def concat-root-dec-eq-concat[OF ⟨ws ∈ lists{x,y}⟩] by fact

have concat(map ?R [x,x]) ≤f ws' · ws' and concat(map ?R [y,y]) ≤f ws' · ws'
  unfolding ws'-def
  using concat-mono-fac[OF map-mono-sublist[OF ⟨[x,x] ≤f ws · ws⟩]]
  concat-mono-fac[OF map-mono-sublist[OF ⟨[y,y] ≤f ws · ws⟩]]
  unfolding concat-morph map-append.

have Suc (Suc (e_ρ x + e_ρ x - 2)) = e_ρ x + e_ρ x
  using Suc-minus2 primroot-exp-nemp[OF ⟨x ≠ ε⟩] by simp
have concat (map ?R [x,x]) = [ρ x] ⊙ (Suc (e_ρ x - 1) + Suc (e_ρ x - 1))
  unfolding Suc-minus-pos[OF primroot-exp-nemp[OF ⟨x ≠ ε⟩]] by (simp add:
add-exps)
hence [ρ x, ρ x] ≤f concat (map ?R [x,x])
  by auto
thus [ρ x, ρ x] ≤f ws' · ws'
  using fac-trans[OF - ⟨concat(map ?R [x,x]) ≤f ws' · ws'⟩] by blast

have Suc (Suc (e_ρ y + e_ρ y - 2)) = e_ρ y + e_ρ y
  using Suc-minus2 primroot-exp-nemp[OF ⟨y ≠ ε⟩] by simp
have concat (map ?R [y,y]) = [ρ y] ⊙ (Suc (e_ρ y - 1) + Suc (e_ρ y - 1))
  unfolding Suc-minus-pos[OF primroot-exp-nemp[OF ⟨y ≠ ε⟩]] by (simp add:
add-exps)
hence [ρ y, ρ y] ≤f concat (map ?R [y,y])
  by auto
thus [ρ y, ρ y] ≤f ws' · ws'
  using fac-trans[OF - ⟨concat(map ?R [y,y]) ≤f ws' · ws'⟩] by blast

show primitive (ρ x) and primitive (ρ y)
  using primroot-prim ⟨x ≠ ε⟩ ⟨y ≠ ε⟩ by blast+
qed

```

0.5.5 Obtaining the square of the longer word (gluing)

lemma bin-imprim-longer-twice:

— 1. If there are both squares, then contradiction; 2. If a square is missing: a) if y appears once: the positive conclusion b) if y appears twice, then gluing preserves presence of the longer word at least twice (because both appear twice) and induction yields $[x', x', y']$ where y' is a suffix of x' , a contradiction with primitivity of words

of the form $xyxy$;

```

assumes  $x \cdot y \neq y \cdot x$ 
  and  $ws \in \text{lists } \{x, y\}$ 
  and  $|y| \leq |x|$ 
  and  $\text{count-list } ws \ x \geq 2$ 
  and  $\text{primitive } ws$  and  $\neg \text{primitive } (\text{concat } ws)$ 
shows  $ws \sim [x,x,y] \wedge \text{primitive } x \wedge \text{primitive } y$ 
using  $\text{assms}$  proof ( $\text{induction } |ws|$   $\text{arbitrary: } x \ y \ ws$   $\text{rule: less-induct}$ )
case  $\text{less}$ 
then show  $?case$ 
proof ( $\text{cases}$ )
  assume  $[x, x] \leq_f ws \cdot ws \wedge [y, y] \leq_f ws \cdot ws$ 
  with  $\text{bin-imprim-both-squares}[OF \langle x \cdot y \neq y \cdot x \rangle \langle ws \in \text{lists } \{x,y\} \rangle \langle \text{primitive } ws \rangle \langle \neg \text{primitive } (\text{concat } ws) \rangle]$ 
  have  $\text{False}$  by  $\text{blast}$ 
  thus  $?case$  by  $\text{blast}$ 
next
assume  $\text{missing-sq: } \neg ([x, x] \leq_f ws \cdot ws \wedge [y, y] \leq_f ws \cdot ws)$ 
then show  $?case$ 
proof ( $\text{cases}$ )
  assume  $\text{count-list } ws \ y < 2$ 
  with  $\text{bin-imprim-single-y}[OF \text{less.prem}(1-4) \text{ this less.prem}(5-6)]$ 
  show  $ws \sim [x,x,y] \wedge \text{primitive } x \wedge \text{primitive } y$ 
  by  $\text{blast}$ 
next
assume  $\neg \text{count-list } ws \ y < 2$  hence  $2 \leq \text{count-list } ws \ y$  by  $\text{simp}$ 

```

— Missing square and two y 's allow gluing

```

define  $x'$  where  $x' = (\text{if } \neg [x, x] \leq_f ws \cdot ws \text{ then } x \text{ else } y)$ 
define  $y'$  where  $y' = (\text{if } \neg [x, x] \leq_f ws \cdot ws \text{ then } y \text{ else } x)$ 

have  $\{x', y'\} = \{x, y\}$ 
  by ( $\text{simp add: doubleton-eq-iff } x'\text{-def } y'\text{-def}$ )
note  $\text{cases} = \text{disjE}[OF \text{this}[\text{unfolded doubleton-eq-iff}]]$ 
have  $\neg [x', x'] \leq_f ws \cdot ws$ 
  using  $\text{missing-sq } x'\text{-def}$  by  $\text{presburger}$ 
have  $\text{count-list } ws \ x' \geq 2$  and  $\text{count-list } ws \ y' \geq 2$ 
  unfolding  $x'\text{-def } y'\text{-def}$  using  $\langle 2 \leq \text{count-list } ws \ x \rangle \langle 2 \leq \text{count-list } ws \ y \rangle$ 
by  $\text{presburger+}$ 
have  $x' \cdot y' \neq y' \cdot x'$ 
  by ( $\text{rule cases, simp-all add: } \langle x \cdot y \neq y \cdot x \rangle \langle x \cdot y \neq y \cdot x \rangle[\text{symmetric}]$ )
have  $x' \neq \varepsilon$  and  $x' \neq y'$  and  $x' \cdot y' \neq y'$ 
  using  $\langle x' \cdot y' \neq y' \cdot x' \rangle$  by  $\text{auto}$ 

```

— rotating last if necessary for successful gluing

```

note  $\text{prim-nemp}[OF \langle \text{primitive } ws \rangle]$ 
hence  $\text{rot: last } ws = x' \implies \text{hd } ws = x' \implies \text{butlast } ws \cdot [x', x'] \cdot \text{tl } ws = ws \cdot ws$ 
using  $\text{append-butlast-last-id hd-tl hd-word rassoc}$  by  $\text{metis}$ 

```

from *this*[*THEN facI*]
have $last\ ws = x' \implies hd\ ws \neq x'$
using $\langle \neg [x', x'] \leq_f ws \cdot ws \rangle$ **by** *blast*
define ws' **where** $ws' = (if\ last\ ws \neq x'\ then\ ws\ else\ tl\ ws \cdot [hd\ ws])$
have $cond: ws' = \varepsilon \vee last\ ws' \neq x'$ — *gluing condition*
unfolding ws' -*def* **using** $\langle last\ ws = x' \implies hd\ ws \neq x' \rangle$ **by** *simp*
have $ws' \sim ws$
unfolding ws' -*def* **using** $\langle ws \neq \varepsilon \rangle$ **by** *fastforce*
hence $counts': count-list\ ws'\ x' \geq 2\ count-list\ ws'\ y' \geq 2$
by (*simp-all add: $\langle 2 \leq count-list\ ws\ x' \rangle \langle 2 \leq count-list\ ws\ y' \rangle count-list-conjug$*)

— *verify induction assumptions of the glued word*

let $?ws = glue\ x'\ ws'$
have $c1: |?ws| < |ws|$
using $len-glue[OF\ cond]$ $conjug-len[OF\ \langle ws' \sim ws \rangle]$ $\langle count-list\ ws'\ x' \geq 2 \rangle$
by *linarith*
hence $c2: (x' \cdot y') \cdot y' \neq y' \cdot x' \cdot y'$
using $\langle x' \cdot y' \neq y' \cdot x' \rangle$ **by** *force*

have $ws' \leq_f ws \cdot ws$
using $conjugE[OF\ \langle ws' \sim ws \rangle]$ *rassoc sublist-appendI* **by** *metis*
hence $\neg [x', x'] \leq_f ws'$
using $\langle \neg [x', x'] \leq_f ws \cdot ws \rangle$ **by** *blast*
have $ws' \in lists\ \{x', y'\}$
using $conjug-in-lists[OF\ \langle ws' \sim ws \rangle \langle ws \in lists\ \{x, y\} \rangle]$ $\langle folded\ \langle \{x', y'\} = \{x, y\} \rangle \rangle$.
have $c3: ?ws \in lists\ \{x' \cdot y', y'\}$
using $single-bin-glue-in-lists[OF\ cond\ \langle \neg [x', x'] \leq_f ws' \rangle \langle ws' \in lists\ \{x', y'\} \rangle]$.

have $c4: 2 \leq count-list\ (glue\ x'\ ws')\ (x' \cdot y')$
using $\langle 2 \leq count-list\ ws'\ x' \rangle$
unfolding $count-list-single-bin-glue(1)[OF\ \langle x' \neq \varepsilon \rangle \langle x' \neq y' \rangle\ cond\ \langle ws' \in lists\ \{x', y'\} \rangle \langle \neg [x', x'] \leq_f ws' \rangle]$.

from $\langle primitive\ ws \rangle$ $[folded\ conjug-prim-iff[OF\ \langle ws' \sim ws \rangle]]$
have $c5: primitive\ (glue\ x'\ ws')$
using $prim-bin-glue\ [OF\ \langle ws' \in lists\ \{x', y'\} \rangle \langle x' \neq \varepsilon \rangle\ cond]$ **by** *blast*

have $count-list\ ws'\ x' \geq 2$
using $\langle count-list\ ws\ x \geq 2 \rangle \langle count-list\ ws\ y \geq 2 \rangle \langle \{x', y'\} = \{x, y\} \rangle$
 $count-list-conjug[OF\ \langle ws' \sim ws \rangle]$ x' -*def* **by** *metis*

have $concat\ (glue\ x'\ ws') = concat\ ws'$
by (*simp add: cond*)

have $c6: \neg primitive\ (concat\ (glue\ x'\ ws'))$
unfolding $\langle concat\ (glue\ x'\ ws') = concat\ ws' \rangle$ **using** $\langle \neg primitive\ (concat\ ws) \rangle \langle ws' \sim ws \rangle$
 $conjug-concat-conjug\ prim-conjug$ **by** *metis*
— *The claim holds by induction*

```

from less.hyps[OF c1 c2 c3 - c4 c5 c6]
have glue  $x' ws' \sim [x' \cdot y', x' \cdot y', y']$  by simp
  — Which is impossible after gluing
from prim-xyxyy[OF  $\langle x' \cdot y' \neq y' \cdot x' \rangle$ ] conjug-prim-iff[OF conjug-concat-conjug[OF
this]]
have False
  using  $\langle \neg \text{primitive } (\text{concat } (\text{glue } x' ws')) \rangle$  by simp
  thus ?case by blast
qed
qed
qed

```

lemma *bin-imprim-both-twice*:

```

assumes  $x \cdot y \neq y \cdot x$ 
and  $ws \in \text{lists } \{x, y\}$ 
and count-list  $ws\ x \geq 2$ 
and count-list  $ws\ y \geq 2$ 
and primitive  $ws$  and  $\neg \text{primitive } (\text{concat } ws)$ 
shows False
proof—
have  $x \neq y$ 
  using  $\langle x \cdot y \neq y \cdot x \rangle$  by blast
from bin-imprim-longer-twice[OF assms(1–2) - assms(3) assms(5–6)]
  bin-imprim-longer-twice[OF assms(1)[symmetric] assms(2)[unfolded insert-commute[of
x]] - assms(4) assms(5–6)]
have or:  $ws \sim [x, x, y] \vee ws \sim [y, y, x]$  by linarith
thus False
proof (rule disjE)
  assume  $ws \sim [x, x, y]$ 
  from  $\langle \text{count-list } ws\ y \geq 2 \rangle$ [unfolded count-list-conjug[OF this]]
  show False
    using  $\langle x \neq y \rangle$  by force
  next
  assume  $ws \sim [y, y, x]$ 
  from  $\langle \text{count-list } ws\ x \geq 2 \rangle$ [unfolded count-list-conjug[OF this]]
  show False
    using  $\langle x \neq y \rangle$  by force
qed
qed

```

0.6 Examples

```

lemma  $x \neq \varepsilon \implies \varepsilon (x \cdot x) \varepsilon \sim_{\mathcal{I}} [x, x]$ 
unfolding factor-interpretation-def
by simp

```

```

lemma assumes  $x = [(0::\text{nat}), 1, 0, 1, 0]$  and  $y = [1, 0, 0, 1]$ 
shows  $[0, 1] (x \cdot x) [1, 0] \sim_{\mathcal{I}} [x, y, x]$ 
unfolding factor-interpretation-def assms by (simp add: suffix-def)

```

0.7 Primitivity non-preserving binary code

In this section, we give the final form of imprimitive words over a given binary code $\{x, y\}$. We start with a lemma, then we show that the only possibility is that such word is conjugate with $x^{\textcircled{a}} j \cdot y^{\textcircled{a}} k$.

lemma *bin-imprim-expsE-y*: **assumes** $x \cdot y \neq y \cdot x$ **and**

$ws \in \text{lists } \{x, y\}$ **and**

$2 \leq |ws|$ **and**

primitive ws **and**

$\neg \text{primitive } (\text{concat } ws)$ **and**

count-list ws y = 1

obtains $j k$ **where** $1 \leq j \ 1 \leq k \ j = 1 \vee k = 1$

$ws \sim [x]^{\textcircled{a}} j \cdot [y]^{\textcircled{a}} k$

proof–

have $x \neq y$ **using** $\langle x \cdot y \neq y \cdot x \rangle$ **by** *blast*

obtain $j1 \ j2$ **where** $[x]^{\textcircled{a}} j1 \cdot [y] \cdot [x]^{\textcircled{a}} j2 = ws$

using *bin-count-one-decompose*[*OF* $\langle ws \in \text{lists } \{x, y\} \ \langle x \neq y \ \langle \text{count-list } ws \ y = 1 \rangle \rangle$].

have $1 \leq j2 + j1$

using $\langle [x]^{\textcircled{a}} j1 \cdot [y] \cdot [x]^{\textcircled{a}} j2 = ws \ \langle 2 \leq |ws| \rangle$ *not-less-eq-eq* **by** *fastforce*

have $ws \sim [x]^{\textcircled{a}} (j2+j1) \cdot [y]^{\textcircled{a}} 1$

using *conjugI'*[*of* $[x]^{\textcircled{a}} j1 \cdot [y] \ [x]^{\textcircled{a}} j2$]

unfolding $\langle [x]^{\textcircled{a}} j1 \cdot [y] \cdot [x]^{\textcircled{a}} j2 = ws \rangle$ [*symmetric*] *add-exps rassoc pow-1*.

from *that*[*OF* $\langle 1 \leq j2 + j1 \rangle$ - - *this*]

show *?thesis*

by *blast*

qed

lemma *bin-imprim-expsE*: **assumes** $x \cdot y \neq y \cdot x$ **and**

$ws \in \text{lists } \{x, y\}$ **and**

$2 \leq |ws|$ **and**

primitive ws **and**

$\neg \text{primitive } (\text{concat } ws)$

obtains $j k$ **where** $1 \leq j \ 1 \leq k \ j = 1 \vee k = 1$

$ws \sim [x]^{\textcircled{a}} j \cdot [y]^{\textcircled{a}} k$

proof–

note $\langle ws \in \text{lists } \{x, y\} \rangle$ [*unfolded insert-commute*[*of x*]]

from *bin-lists-count-zero*[*OF* $\langle ws \in \text{lists } \{x, y\} \rangle$]

sing-lists-exp-len[*of ws y*]

prim-exp-one[*OF* $\langle \text{primitive } ws \rangle$, *of* $[y] \ |ws|$]

have *count-list ws x* $\neq 0$

using $\langle 2 \leq |ws| \rangle$ **by** *fastforce*

from *bin-lists-count-zero*[*OF* $\langle ws \in \text{lists } \{y, x\} \rangle$]

sing-lists-exp-len[*of ws x*]

prim-exp-one[*OF* $\langle \text{primitive } ws \rangle$, *of* $[x] \ |ws|$]

have *count-list ws y* $\neq 0$


```

using  $\langle 2 \leq |ws| \rangle$  by fastforce

consider count-list ws x = 1 | count-list ws y = 1
using bin-imprim-both-twice[OF  $\langle x \cdot y \neq y \cdot x \rangle \langle ws \in \text{lists } \{x, y\} \rangle$  - -
   $\langle \text{primitive } ws \rangle \langle \neg \text{primitive } (\text{concat } ws) \rangle$ 
   $\langle \text{count-list } ws \ x \neq 0 \rangle \langle \text{count-list } ws \ y \neq 0 \rangle$ 
unfolding One-less-Two-le-iff[symmetric] less-one[symmetric] by fastforce
thus thesis
proof(cases)
  assume count-list ws x = 1
  from bin-imprim-expsE-y[reversed, OF  $\langle x \cdot y \neq y \cdot x \rangle \langle ws \in \text{lists } \{y, x\} \rangle \langle 2 \leq$ 
 $|ws| \rangle$ 
     $\langle \text{primitive } ws \rangle \langle \neg \text{primitive } (\text{concat } ws) \rangle \langle \text{count-list } ws \ x = 1 \rangle$ ]
  show thesis
  using that by metis
next
  assume count-list ws y = 1
  from bin-imprim-expsE-y[OF  $\langle x \cdot y \neq y \cdot x \rangle \langle ws \in \text{lists } \{x, y\} \rangle \langle 2 \leq |ws| \rangle$ 
     $\langle \text{primitive } ws \rangle \langle \neg \text{primitive } (\text{concat } ws) \rangle \langle \text{count-list } ws \ y = 1 \rangle$ ]
  show ?thesis
  using that.
qed
qed

```

0.7.1 The target theorem

Given a binary code $\{x, y\}$ such that there is a primitive factorisation ws over it whose concatenation is imprimitive, we finally show that there are integers j and k (depending only on $\{x, y\}$) such that any other such factorisation ws' is conjugate to $[x]^{\textcircled{j}} \cdot [y]^{\textcircled{k}}$.

theorem *bin-imprim-code*: **assumes** $x \cdot y \neq y \cdot x$ **and** $ws \in \text{lists } \{x, y\}$ **and** $2 \leq |ws|$ **and** *primitive ws* **and** $\neg \text{primitive } (\text{concat } ws)$

obtains $j \ k$ **where** $1 \leq j$ **and** $1 \leq k$ **and** $j = 1 \vee k = 1$

$\wedge ws. ws \in \text{lists } \{x, y\} \implies 2 \leq |ws| \implies$

$(\text{primitive } ws \wedge \neg \text{primitive } (\text{concat } ws) \iff ws \sim [x]^{\textcircled{j}} \cdot [y]^{\textcircled{k}})$ **and**

$|y| \leq |x| \implies 2 \leq j \implies j = 2 \wedge \text{primitive } x \wedge \text{primitive } y$ **and**

$|y| \leq |x| \implies 2 \leq k \implies j = 1 \wedge \text{primitive } x$

proof–

obtain $j \ k$ **where** $1 \leq j \ 1 \leq k \ j = 1 \vee k = 1$

$ws \sim [x]^{\textcircled{j}} \cdot [y]^{\textcircled{k}}$

using *bin-imprim-expsE*[*OF* $\langle x \cdot y \neq y \cdot x \rangle$]

using *assms by metis*

have $\neg \text{primitive } (x^{\textcircled{j}} \cdot y^{\textcircled{k}})$

using $\langle \neg \text{primitive } (\text{concat } ws) \rangle$

unfolding *concat-morph concat-sing-pow*

conjug-prim-iff[*OF* *conjug-concat-conjug*[*OF* $\langle ws \sim [x]^{\textcircled{j}} \cdot [y]^{\textcircled{k}} \rangle$]].

from *not-prim-primroot-expE*[*OF this*]
obtain $z\ l$ **where** [*symmetric*]: $z^{\textcircled{a}l} = x^{\textcircled{a}j} \cdot y^{\textcircled{a}k}$ **and** $2 \leq l$.

show *thesis*

proof (*rule that*[*of j k*])

show $1 \leq j\ 1 \leq k\ j = 1 \vee k = 1$ **by** *fact+*

fix ws'

assume *hyps*: $ws' \in \text{lists } \{x, y\}\ 2 \leq |ws'|$

show $\text{primitive } ws' \wedge \neg \text{primitive } (\text{concat } ws') \iff ws' \sim [x]^{\textcircled{a}j} \cdot [y]^{\textcircled{a}k}$

proof

assume $\text{primitive } ws' \wedge \neg \text{primitive } (\text{concat } ws')$

hence *prems*: $\text{primitive } ws' \wedge \neg \text{primitive } (\text{concat } ws')$ **by** *blast+*

obtain $j'\ k'$ **where** $1 \leq j'\ 1 \leq k'\ j' = 1 \vee k' = 1$

$ws' \sim [x]^{\textcircled{a}j'} \cdot [y]^{\textcircled{a}k'}$

using *bin-imprim-expsE*[*OF* $\langle x \cdot y \neq y \cdot x \rangle$ *hyps prems*].

have $\neg \text{primitive } (x^{\textcircled{a}j'} \cdot y^{\textcircled{a}k'})$

using $\langle \neg \text{primitive } (\text{concat } ws') \rangle$

unfolding *concat-morph concat-sing-pow*

conjug-prim-iff[*OF* *conjug-concat-conjug*[*OF* $\langle ws' \sim [x]^{\textcircled{a}j'} \cdot [y]^{\textcircled{a}k'} \rangle$]].

have $j = j'\ k = k'$

using *LS-unique*[*OF* $\langle x \cdot y \neq y \cdot x \rangle$

$\langle 1 \leq j \rangle \langle 1 \leq k \rangle \langle \neg \text{primitive } (x^{\textcircled{a}j} \cdot y^{\textcircled{a}k}) \rangle$

$\langle 1 \leq j' \rangle \langle 1 \leq k' \rangle \langle \neg \text{primitive } (x^{\textcircled{a}j'} \cdot y^{\textcircled{a}k'}) \rangle$].

show $ws' \sim [x]^{\textcircled{a}j} \cdot [y]^{\textcircled{a}k}$

unfolding $\langle j = j' \rangle \langle k = k' \rangle$ **by** *fact*

next

assume $ws' \sim [x]^{\textcircled{a}j} \cdot [y]^{\textcircled{a}k}$

note *conjug-trans*[*OF* $\langle ws \sim [x]^{\textcircled{a}j} \cdot [y]^{\textcircled{a}k} \rangle$ *conjug-sym*[*OF this*]]

from *prim-conjug*[*OF* $\langle \text{primitive } ws \rangle$ *this*]

$\langle \neg \text{primitive } (\text{concat } ws) \rangle$ [*unfolded conjug-concat-prim-iff*[*OF* $\langle ws \sim ws' \rangle$]]

show $\text{primitive } ws' \wedge \neg \text{primitive } (\text{concat } ws')$ **by** *blast*

qed

next

assume $|y| \leq |x|$

interpret *LS-len-le* $x\ y\ j\ k\ l\ z$

by *unfold-locales fact+*

assume $2 \leq j$

with *jk-small*

have $k = 1$ **by** *fastforce*

from *case-j2k1*[*OF* $\langle 2 \leq j \rangle$ *this*]

show $j = 2 \wedge \text{primitive } x \wedge \text{primitive } y$

by *blast*

next

assume $|y| \leq |x|$

interpret *LS-len-le* $x\ y\ j\ k\ l\ z$
by *unfold-locales fact+*

assume $2 \leq k$
show $j = 1 \wedge \text{primitive } x$
using $\langle 2 \leq k \rangle \langle j = 1 \vee k = 1 \rangle$ *case-j1k2-primitive* **by** *auto*
qed
qed

— Formulation in terms of (binary) primitive morphism

definition *bin-imprim-code* **where** *bin-imprim-code* $x\ y \equiv x \cdot y \neq y \cdot x \wedge (\neg \text{bin-prim } x\ y)$

theorem *bin-imprim-code'*: **assumes** *bin-imprim-code* $x\ y$
obtains $j\ k$ **where** $1 \leq j$ **and** $1 \leq k$ **and** $j = 1 \vee k = 1$
 $\bigwedge ws. ws \in \text{lists } \{x, y\} \implies 2 \leq |ws| \implies$
 $(\text{primitive } ws \wedge \neg \text{primitive } (\text{concat } ws) \longleftrightarrow ws \sim [x]^{\textcircled{j}} \cdot [y]^{\textcircled{k}})$ **and**
 $|y| \leq |x| \implies 2 \leq j \implies j = 2 \wedge \text{primitive } x \wedge \text{primitive } y$ **and**
 $|y| \leq |x| \implies 2 \leq k \implies j = 1 \wedge \text{primitive } x$

proof—

thm *bin-imprim-code*
obtain ws **where** $x \cdot y \neq y \cdot x$
and $ws \in \text{lists } \{x, y\}$ **and** $2 \leq |ws|$ **and** *primitive* ws **and** $\neg \text{primitive } (\text{concat } ws)$
using *assms unfolding bin-imprim-code-def bin-prim-altdef2* **by** *blast*
from *bin-imprim-code[OF this]* **that**
show *thesis*
by *blast*
qed

end

theory *Binary-Imprimitive-Deduction*
imports
Binary-Code-Imprimitive.Binary-Code-Imprimitive

begin

0.8 Upper bound of the power exponent in the canonical imprimitivity witness

lemma *LS-power-len-ge*:
assumes $y^{\textcircled{k}} \cdot x = z^{\textcircled{l}}$
and $k * |y| \geq |z| + |y| - 1$
shows $x \cdot y = y \cdot x$

proof (*rule nemp-comm*)
assume $y \neq \varepsilon$
have $y^{\textcircled{k}} \leq_p z \cdot y^{\textcircled{k}}$
using $\langle y^{\textcircled{k}} \cdot x = z^{\textcircled{l}} \rangle$
by (*blast intro!: pref-prod-root*)
moreover have $y^{\textcircled{k}} \leq_p y \cdot y^{\textcircled{k}}$
by (*blast intro!: pref-pow-ext'*)
moreover have $1 \leq \text{gcd } |z| |y|$
using $\langle y \neq \varepsilon \rangle$
by (*simp flip: less-eq-Suc-le*)
from *this* $\langle k * |y| \geq |z| + |y| - 1 \rangle$
have $|z| + |y| - (\text{gcd } |z| |y|) \leq k * |y|$
by (*rule le-trans[OF diff-le-mono2]*)
ultimately have $z \cdot y = y \cdot z$
unfolding *pow-len[symmetric]* **by** (*fact per-lemma-comm*)
with $\langle y^{\textcircled{k}} \cdot x = z^{\textcircled{l}} \rangle$
show $x \cdot y = y \cdot x$
by (*fact LS-comm*)
qed

lemma *LS-root-len-ge*:

assumes $y^{\textcircled{k}} \cdot x = z^{\textcircled{l}}$
and $1 \leq k$ **and** $2 \leq l$
and $x \cdot y \neq y \cdot x$
shows $(k - 1) * |y| + 2 \leq |z|$
proof (*intro leI notI*)
assume $|z| < (k - 1) * |y| + 2$
then have $|z| + |y| \leq \text{Suc } (k - 1) * |y| + 1$
by *simp*
also have $\dots = k * |y| + 1$
using $\langle 1 \leq k \rangle$ **by** *simp*
finally have $k * |y| \geq |z| + |y| - 1$
unfolding *le-diff-conv*.
from $\langle x \cdot y \neq y \cdot x \rangle$ *LS-power-len-ge*[*OF* $\langle y^{\textcircled{k}} \cdot x = z^{\textcircled{l}} \rangle$ *this*]
show *False..*
qed

lemma *LS-root-len-le*:

assumes $y^{\textcircled{k}} \cdot x = z^{\textcircled{l}}$
and $1 \leq k$ **and** $2 \leq l$
and $x \cdot y \neq y \cdot x$
shows $|z| \leq |x| + |y| - 2$
proof –
have $|x| + k * |y| = l * |z|$
using *lenarg*[*OF* $\langle y^{\textcircled{k}} \cdot x = z^{\textcircled{l}} \rangle$]
by (*simp only: pow-len lenmorph add.commute*[*of* $|x|$])
have $|z| \leq (l - 1) * |z|$
using *diff-le-mono*[*OF* $\langle 2 \leq l, \text{ of } 1 \rangle$] **by** *simp*
also have $\dots = |x| + k * |y| - |z|$

unfolding *diff-mult-distrib* $\langle |x| + k * |y| = l * |z| \rangle$ [*symmetric*] **by** *simp*
also have $\dots \leq |x| + k * |y| - ((k - 1) * |y| + 2)$
using *LS-root-len-ge* [*OF assms*]
by (*rule diff-le-mono2*)
also have $\dots \leq |x| + |y| - 2$
using $\langle 1 \leq k \rangle$ **unfolding** *diff-diff-eq* [*symmetric*]
by (*intro diff-le-mono*) (*simp add: le-diff-conv add.assoc diff-mult-distrib*)
finally show $|z| \leq |x| + |y| - 2$.
qed

lemma *LS-exp-le'*:

assumes $y @ k \cdot x = z @ l$
and $2 \leq l$
and $x \cdot y \neq y \cdot x$
shows $k \leq (|x| - 4) \text{ div } |y| + 2$
proof (*cases* $1 \leq k$)
assume $1 \leq k$
have $|y| > 0$
using $\langle x \cdot y \neq y \cdot x \rangle$ **by** *blast*
have $(k - 1) * |y| + 2 \leq |z|$
using *LS-root-len-ge* $\langle y @ k \cdot x = z @ l \rangle$ $\langle 1 \leq k \rangle$ $\langle 2 \leq l \rangle$ $\langle x \cdot y \neq y \cdot x \rangle$.
also have $|z| \leq |x| + |y| - 2$
using *LS-root-len-le* $\langle y @ k \cdot x = z @ l \rangle$ $\langle 1 \leq k \rangle$ $\langle 2 \leq l \rangle$ $\langle x \cdot y \neq y \cdot x \rangle$.
finally have $(k - 1) * |y| + 2 \leq |x| + |y| - 2$.
then have $(k - 1) * |y| + 2 - |y| - 2 \leq |x| + |y| - 2 - |y| - 2$
by (*intro diff-le-mono*)
then have $(k - 1) * |y| + 2 - 2 - |y| \leq |x| - (2 + 2)$
unfolding *diff-commute* [*of - 2 |y|*] **unfolding** *diff-add-inverse2* *diff-diff-eq*.
then have $(k - (1 + 1)) * |y| \leq |x| - 4$
unfolding *diff-add-inverse2* *nat-distrib* *diff-diff-eq* *mult-1*
by *presburger*
then show $k \leq (|x| - 4) \text{ div } |y| + 2$
using $\langle |y| > 0 \rangle$
by (*simp only: less-eq-div-iff-mult-less-eq one-add-one flip: le-diff-conv*)
qed (*simp add: trans-le-add2*)

lemma *LS-exp-le*:

assumes $x \cdot y @ k = z @ l$
and $2 \leq l$
and $x \cdot y \neq y \cdot x$
shows $k \leq (|x| - 4) \text{ div } |y| + 2$
using *LS-exp-le'* [*reversed, OF* $\langle x \cdot y @ k = z @ l \rangle$ $\langle 2 \leq l \rangle$ $\langle x \cdot y \neq y \cdot x \rangle$] [*symmetric*].

thm *bin-imprim-expsE*

lemma *bin-imprim-code-witnessE*:

assumes $x \cdot y \neq y \cdot x$ **and** $|y| \leq |x|$
and $ws \in \text{lists } \{x, y\}$ **and** $2 \leq |ws|$
and *primitive* ws **and** $\neg \text{primitive } (\text{concat } ws)$

obtains $ws \sim [x, x, y]$
 $|k$ **where** $1 \leq k$ **and** $k \leq (|x| - 4) \text{ div } |y| + 2$
and $ws \sim [x] \cdot [y]^{\textcircled{a} k}$
proof –
obtain $j k$
where $1 \leq j$ **and** $1 \leq k$ **and** $j = 1 \vee k = 1$
and *witness-iff*: $\bigwedge ws. ws \in \text{lists } \{x, y\} \implies 2 \leq |ws| \implies$
 $(\text{primitive } ws \wedge \neg \text{primitive } (\text{concat } ws)) \iff ws \sim [x]^{\textcircled{a} j} \cdot [y]^{\textcircled{a} k}$
and
 $j\text{-ge}$: $|y| \leq |x| \implies 2 \leq j \implies j = 2 \wedge \text{primitive } x \wedge \text{primitive } y$ **and**
 $k\text{-ge}$: $|y| \leq |x| \implies 2 \leq k \implies j = 1 \wedge \text{primitive } x$
by (*fact bin-imprim-code*[*OF assms*(1, 3 – 6)])
have $ws \sim [x]^{\textcircled{a} j} \cdot [y]^{\textcircled{a} k}$
using *witness-iff*[*OF* $\langle ws \in \text{lists } \{x, y\} \rangle \langle 2 \leq |ws| \rangle \langle \text{primitive } ws \rangle \langle \neg \text{primitive } (\text{concat } ws) \rangle$]
by *simp*
show *thesis*
proof (*cases*)
assume $2 \leq j$
from $j\text{-ge}$ [*OF* $\langle |y| \leq |x| \rangle \text{this} \rangle \langle j = 1 \vee k = 1 \rangle$]
have $j = 2$ **and** $k = 1$
by *simp-all*
then have $ws \sim [x, x, y]$
using $\langle ws \sim [x]^{\textcircled{a} j} \cdot [y]^{\textcircled{a} k} \rangle$ **by** *simp*
then show *thesis..*
next
assume $\neg 2 \leq j$
with $\langle 1 \leq j \rangle$ **have** $j = 1$
by *simp*
then have $ws \sim [x] \cdot [y]^{\textcircled{a} k}$
using $\langle ws \sim [x]^{\textcircled{a} j} \cdot [y]^{\textcircled{a} k} \rangle$ **by** *simp*
then have $\neg \text{primitive } (x \cdot y^{\textcircled{a} k})$
using $\langle \neg \text{primitive } (\text{concat } ws) \rangle$ **unfolding** *conjug-concat-prim-iff*[*OF* $\langle ws \sim [x] \cdot [y]^{\textcircled{a} k} \rangle$]
by *simp*
moreover have $x \cdot y^{\textcircled{a} k} \neq \varepsilon$
using $\langle x \cdot y \neq y \cdot x \rangle$ **by** (*intro notI*) *simp*
ultimately obtain $z l$
where $2 \leq l$ **and** $z^{\textcircled{a} l} = x \cdot y^{\textcircled{a} k}$
by (*elim not-prim-expE*)
have $k \leq (|x| - 4) \text{ div } |y| + 2$
using *LS-exp-le*[*OF* $\langle z^{\textcircled{a} l} = x \cdot y^{\textcircled{a} k} \rangle$ [*symmetric*] $\langle 2 \leq l \rangle \langle x \cdot y \neq y \cdot x \rangle$].
from $\langle 1 \leq k \rangle$ **this** $\langle ws \sim [x] \cdot [y]^{\textcircled{a} k} \rangle$
show *thesis..*
qed
qed

0.8.1 Optimality of the exponent upper bound

lemma *examples-bound-optimality*:

fixes $m\ k$ **and** $x\ y\ z :: \text{binA list}$

assumes $1 \leq m$ **and** $k' = 0 \implies m = 1$

defines $x \equiv a \cdot b \cdot (b \cdot (a \cdot b)^{\textcircled{m}})^{\textcircled{k' \cdot b \cdot a}}$

and $y \equiv b \cdot (a \cdot b)^{\textcircled{m}}$

and $z \equiv a \cdot b \cdot (b \cdot (a \cdot b)^{\textcircled{m}})^{\textcircled{k' + 1}}$

and $k \equiv k' + 2$

shows $|y| \leq |x|$ **and** $x \cdot y^{\textcircled{k}} = z \cdot z$ **and** $k = (|x| - 4) \text{ div } |y| + 2$

proof –

obtain m' **where** $m: m = m' + 1$

using $\langle 1 \leq m \rangle$ **using** *add.commute le-Suc-ex* **by** *blast*

have $x\text{-len}: |x| = k' * (2 * m + 1) + 4$

unfolding $x\text{-def}$ **by** (*simp add: pow-len*)

have $y\text{-len}: |y| = 2 * m + 1$

unfolding $y\text{-def}$ **by** (*simp add: pow-len*)

have $z\text{-len}: |z| = (k' + 1) * (2 * m + 1) + 2$

unfolding $z\text{-def}$ **by** (*simp add: pow-len*)

show $|y| \leq |x|$

using $\langle k' = 0 \implies m = 1 \rangle$ **unfolding** $x\text{-len}$ $y\text{-len}$

by (*cases k'*) (*simp-all add: pow-len*)

show $x \cdot y^{\textcircled{k}} = z \cdot z$

unfolding $x\text{-def}$ $y\text{-def}$ $z\text{-def}$ $k\text{-def}$

by *comparison*

show $k = (|x| - 4) \text{ div } |y| + 2$

proof –

have $|x| - 4 = k' * |y|$

unfolding $x\text{-len}$ $y\text{-len}$ **by** *simp*

have $|y| \neq 0$

unfolding $y\text{-def}$ **by** *blast*

have $k' = (|x| - 4) \text{ div } |y|$

unfolding $\langle |x| - 4 = k' * |y| \rangle$ *nonzero-mult-div-cancel-right[OF $\langle |y| \neq 0 \rangle$]*..

then show $k = (|x| - 4) \text{ div } |y| + 2$

unfolding $k\text{-def}$ **by** *blast*

qed

qed

0.9 Characterization of binary primitivity preserving morphisms given by a pair of words

lemma *len-le-not-bin-primE*:

assumes $|y| \leq |x|$

and $\neg \text{bin-prim } x\ y$

obtains $\neg \text{primitive } (x \cdot x \cdot y)$

| k **where** $1 \leq k$ **and** $k \leq (|x| - 4) \text{ div } |y| + 2$

and $\neg \text{primitive } (x \cdot y^{\textcircled{k}})$

proof (*cases*)

assume $x \cdot y = y \cdot x$

have \neg *primitive* $(x \cdot x \cdot y)$
using $\langle x \cdot y = y \cdot x \rangle \langle |y| \leq |x| \rangle$
by (*cases* $x \neq \varepsilon$) (*intro comm-not-prim, simp-all*)
then show *thesis..*
next
assume $x \cdot y \neq y \cdot x$
then have $x \neq y$
by *blast*
obtain *ws* **where**
 $ws \in \text{lists } \{x, y\}$ **and** $2 \leq |ws|$ **and** *primitive* *ws* **and** \neg *primitive* (*concat ws*)
using $\langle \neg \text{bin-prim } x \ y \rangle$ **unfolding** *bin-prim-concat-prim-pres-conv*[*OF* $\langle x \neq y \rangle$]
by *blast*
then consider $ws \sim [x, x, y]$
 $|k$ **where** $1 \leq k$ **and** $k \leq (|x| - 4) \text{ div } |y| + 2$
and $ws \sim [x] \cdot [y]^{\textcircled{a}} k$
by (*rule bin-imprim-code-witnessE*[*OF* $\langle x \cdot y \neq y \cdot x \rangle \langle |y| \leq |x| \rangle$])
then show *thesis*
proof (*cases*)
assume $ws \sim [x, x, y]$
then have \neg *primitive* $(x \cdot x \cdot y)$
using $\langle \neg \text{primitive} (\text{concat } ws) \rangle$
by (*simp add: conjug-concat-prim-iff*)
then show *thesis..*
next
fix k
assume $1 \leq k$ **and** $k \leq (|x| - 4) \text{ div } |y| + 2$ **and** $ws \sim [x] \cdot [y]^{\textcircled{a}} k$
have \neg *primitive* $(x \cdot y^{\textcircled{a}} k)$
using $\langle ws \sim [x] \cdot [y]^{\textcircled{a}} k \rangle \langle \neg \text{primitive} (\text{concat } ws) \rangle$
by (*simp add: conjug-concat-prim-iff*)
from $\langle 1 \leq k \rangle \langle k \leq (|x| - 4) \text{ div } |y| + 2 \rangle$ *this*
show *thesis..*
qed
qed

lemma *bin-prim-xyk*:
assumes *bin-prim* $x \ y$ **and** $0 < k$
shows *primitive* $(x \cdot y^{\textcircled{a}} k)$
proof –
have *primitive* $([x] \cdot [y]^{\textcircled{a}} k)$
using *bin-prim-code*[*OF* $\langle \text{bin-prim } x \ y \rangle$]
by (*intro prim-abk*) *blast*
from *bin-prim-concat-prim-pres*[*OF* $\langle \text{bin-prim } x \ y \rangle - - \text{this} \rangle \langle 0 < k \rangle$]
show *primitive* $(x \cdot y^{\textcircled{a}} k)$
by (*simp add: pow-in-lists*)
qed

lemma *len-le-bin-prim-iff*:
assumes $|y| \leq |x|$
shows

$bin\text{-}prim\ x\ y \longleftrightarrow primitive\ (x \cdot x \cdot y) \wedge (\forall k. 1 \leq k \wedge k \leq (|x| - 4)\ div\ |y| + 2 \longrightarrow primitive\ (x \cdot y^{\textcircled{a}}\ k))$
 (is $bin\text{-}prim\ x\ y \longleftrightarrow (?xxy \wedge ?xyk)$)
proof (intro iffI[OF - contrapos-pp])
 assume $bin\text{-}prim\ x\ y$
 show $?xxy \wedge ?xyk$
proof (intro conjI allI impI)
 show $primitive\ (x \cdot x \cdot y)$
 using $bin\text{-}prim\text{-}xyk[OF\ \langle bin\text{-}prim\ x\ y \rangle[symmetric],\ of\ 2]\ conjug\text{-}prim\text{-}iff'$
 by (simp add: $conjug\text{-}prim\text{-}iff'[of\ y]$)
 show $primitive\ (x \cdot y^{\textcircled{a}}\ k)$ if $1 \leq k \wedge k \leq (|x| - 4)\ div\ |y| + 2$ for k
 using $bin\text{-}prim\text{-}xyk[OF\ \langle bin\text{-}prim\ x\ y \rangle,\ of\ k]\ conjunct1[OF\ that]$
 by fastforce
 qed
next
 assume $\neg bin\text{-}prim\ x\ y$
 then consider $\neg primitive\ (x \cdot x \cdot y)$
 | k where $1 \leq k$ and $k \leq (|x| - 4)\ div\ |y| + 2$
 and $\neg primitive\ (x \cdot y^{\textcircled{a}}\ k)$
 by (elim len-le-not-bin-primE[OF $\langle |y| \leq |x| \rangle$])
 then show $\neg (?xxy \wedge ?xyk)$
 by (cases) blast+
 qed

lemma $len\text{-}eq\text{-}bin\text{-}prim\text{-}iff$:
 assumes $|x| = |y|$
 shows $bin\text{-}prim\ x\ y \longleftrightarrow primitive\ (x \cdot y)$
proof
 show $bin\text{-}prim\ x\ y \implies primitive\ (x \cdot y)$
 using $bin\text{-}prim\text{-}xyk[of\ -\ -\ 1]$
 by simp
 assume $primitive\ (x \cdot y)$
 then have $x \cdot y \neq y \cdot x$
 using $assms\ eq\text{-}append\text{-}not\text{-}prim$ by auto
 from $this\ bin\text{-}uniform\text{-}prim\text{-}morph[OF\ this\ \langle |x| = |y| \rangle \langle primitive\ (x \cdot y) \rangle]$
 show $bin\text{-}prim\ x\ y$
 unfolding $bin\text{-}prim\text{-}altdef2$
 by simp
 qed

theorem $bin\text{-}prim\text{-}iff$:
 $bin\text{-}prim\ x\ y \longleftrightarrow$
 (if $|y| < |x|$
 then $primitive\ (x \cdot x \cdot y) \wedge (\forall k. 1 \leq k \wedge k \leq (|x| - 4)\ div\ |y| + 2 \longrightarrow primitive\ (x \cdot y^{\textcircled{a}}\ k))$
 else if $|x| < |y|$
 then $primitive\ (y \cdot y \cdot x) \wedge (\forall k. 1 \leq k \wedge k \leq (|y| - 4)\ div\ |x| + 2 \longrightarrow primitive\ (y \cdot x^{\textcircled{a}}\ k))$
 else $primitive\ (x \cdot y)$)

```

)
proof (cases rule: linorder-cases)
  assume  $|x| < |y|$ 
  then show ?thesis
    unfolding bin-prim-commutes[of x y]
    unfolding len-le-bin-prim-iff[OF less-imp-le[OF <|x| < |y|>]]
    by (simp only: if-not-P[OF less-not-sym] if-P)
next
  assume  $|y| < |x|$ 
  then show ?thesis
    unfolding len-le-bin-prim-iff[OF less-imp-le[OF <|y| < |x|>]]
    by (simp only: if-P)
next
  assume  $|x| = |y|$ 
  then show ?thesis
    unfolding len-eq-bin-prim-iff[OF <|x| = |y|>]
    by simp
qed

```

0.9.1 Code equation for *bin-prim* predicate

```

context
begin

```

```

private lemma all-less-Suc-conv:  $(\forall k < n. P (Suc k)) \longleftrightarrow (\forall k \leq n. k \geq 1 \longrightarrow P k)$ 

```

```

proof (intro iffI allI impI)
  fix k
  assume  $\forall k < n. P (Suc k)$  and  $k \leq n$  and  $1 \leq k$ 
  then show  $P k$ 
    by (elim allE[of - k - 1]) (simp only: Suc-diff-1)
qed simp

```

```

lemma bin-prim-iff' [code]:

```

```

  bin-prim x y  $\longleftrightarrow$ 
    (if  $|y| < |x|$ 
      then primitive (x · x · y)  $\wedge$   $(\forall k < (|x| - 4) \text{ div } |y| + 2. \text{primitive } (x \cdot y^{\textcircled{a}} (Suc k)))$ 
      else if  $|x| < |y|$ 
        then bin-prim y x
        else primitive (x · y)
    )

```

```

proof (cases rule: linorder-cases)
  show  $|x| < |y| \implies ?thesis$ 
    unfolding bin-prim-commutes[of x y]
    by simp
next
  assume  $|y| < |x|$ 
  then show ?thesis

```

```

    using len-le-bin-prim-iff[OF less-imp-le[OF <|y| < |x|>]]
    unfolding all-less-Suc-conv[where P =  $\lambda k. \text{primitive } (- \cdot - \textcircled{\small k})$ ]
    unfolding conj-comms[of 1 ≤ -] imp-conjL
    by (simp only: if-P)
next
  assume |x| = |y|
  then show ?thesis
    unfolding len-eq-bin-prim-iff[OF <|x| = |y|>]
    by simp
qed

end
value bin-prim (a·b·b·a·a) b — True
value bin-prim (a·b·b·a) b — False
value bin-prim (a·b·b·a) (b·a·b·a·b) — False
value bin-prim (a·b) (a·b) — False
value bin-prim (a·b) (a·b·a·b) — False
value bin-prim (a·b·b·a·a) (b·b·b·b·b) — True

```

0.10 Characterization of binary imprimitivity codes

theorem *bin-imprim-code-iff*:

```

bin-imprim-code x y  $\longleftrightarrow$  x · y  $\neq$  y · x  $\wedge$ 
  (if |y| < |x|
    then  $\neg$  primitive (x · x · y)  $\vee$  ( $\exists k. 1 \leq k \wedge k \leq (|x| - 4) \text{ div } |y| + 2 \wedge \neg$ 
      primitive (x · y  $\textcircled{\small k}$ ))
    else if |x| < |y|
      then  $\neg$  primitive (y · y · x)  $\vee$  ( $\exists k. 1 \leq k \wedge k \leq (|y| - 4) \text{ div } |x| + 2 \wedge \neg$ 
        primitive (y · x  $\textcircled{\small k}$ ))
      else  $\neg$  primitive (x · y)
  )
unfolding bin-imprim-code-def bin-prim-iff
by (simp only: de-Morgan-conj not-all not-imp conj-assoc flip: if-image[of - Not])

```

```

value bin-imprim-code (a·b·b·a·a) b — False
value bin-imprim-code (a·b·b·a) b — True
value bin-imprim-code (a·b·b·a) (b·a·b·a·b) — True
value bin-imprim-code (a·b) (a·b) — False
value bin-imprim-code (a·b) (a·b·a·b) — False
value bin-imprim-code (a·b·b·a·a) (b·b·b·b·b) — False

```

end

References

- [1] E. Barbin-Le Rest and M. Le Rest. Sur la combinatoire des codes à deux mots. *Theor. Comput. Sci.*, 41:61–80, 1985.
- [2] J. Mañuch. Defect effect of bi-infinite words in the two-element case. *Discret. Math. Theor. Comput. Sci.*, 4(2):273–290, 2001.
- [3] J.-C. Spehner. *Quelques problèmes d'extension, de conjugaison et de présentation des sous-monoïdes d'un monoïde libre*. PhD thesis, Université Paris VII, Paris, 1976.