

The Jordan-Hölder Theorem

Jakob von Raumer

April 14, 2026

Abstract

This submission contains theories that lead to a formalization of the proof of the Jordan-Hölder theorem about composition series of finite groups. The theories formalize the notions of isomorphism classes of groups, simple groups, normal series, composition series, maximal normal subgroups. Furthermore, they provide proofs of the second isomorphism theorem for groups, the characterization theorem for maximal normal subgroups as well as many useful lemmas about normal subgroups and factor groups. The formalization is based on the work work in my first AFP submission [vR14] while the proof of the Jordan-Hölder theorem itself is inspired by course notes of Stuart Rankin [Ran05].

Contents

1	Facts about maximal normal subgroups	1
2	Normal series and Composition series	4
2.1	Preliminaries	4
2.2	Normal Series	6
2.3	Composition Series	11
3	Isomorphism Classes of Groups	24
4	The Jordan-Hölder Theorem	26

```
theory MaximalNormalSubgroups
imports HOL-Algebra.Algebra
begin
```

1 Facts about maximal normal subgroups

A maximal normal subgroup of G is a normal subgroup which is not contained in other any proper normal subgroup of G .

```
locale max-normal-subgroup = normal +
```

assumes *proper*: $H \neq \text{carrier } G$
assumes *max-normal*: $\bigwedge J. J \triangleleft G \implies J \neq H \implies J \neq \text{carrier } G \implies \neg (H \subseteq J)$

Another characterization of maximal normal subgroups: The factor group is simple.

theorem (in *normal*) *max-normal-simple-quotient*:

assumes *finite*: *finite* (*carrier* G)

shows *max-normal-subgroup* $H \ G = \text{simple-group } (G \text{ Mod } H)$

proof

assume *max-normal-subgroup* $H \ G$

then interpret *maxH*: *max-normal-subgroup* $H \ G$.

show *simple-group* $(G \text{ Mod } H)$ **unfolding** *simple-group-def* *simple-group-axioms-def*

proof (*intro conjI factorgroup-is-group allI impI disjCI*)

have *gt0*: $0 < \text{card } (\text{rcosets } H)$

by (*metis gr-zeroI lagrange-finite assms mult-is-0 order-gt-0-iff-finite subgroup-axioms*)

from *maxH.proper* *finite* **have** $\text{carrier } (G \text{ Mod } H) \neq \{\mathbf{1}_{G \text{ Mod } H}\}$ **using** *fact-group-trivial-iff* **by** *auto*

hence $1 \neq \text{order } (G \text{ Mod } H)$ **using** *factorgroup-is-group* *group.order-one-triv-iff* **by** *metis*

with *gt0* **show** $1 < \text{order } (G \text{ Mod } H)$ **unfolding** *order-def* *FactGroup-def* **by** *auto*

next

fix A'

assume *A'normal*: $A' \triangleleft G \text{ Mod } H$ **and** *A'nottriv*: $A' \neq \{\mathbf{1}_{G \text{ Mod } H}\}$

define A **where** $A = \bigcup A'$

have *A2*: $A \triangleleft G$ **using** *A'normal* **unfolding** *A-def* **by** (*rule factgroup-subgroup-union-normal*)

have $H \in A'$ **using** *A'normal* *normal-imp-subgroup* *subgroup.one-closed* **un-**

folding *FactGroup-def* **by** *force*

hence $H \subseteq A$ **unfolding** *A-def* **by** *auto*

hence *A1*: $H \triangleleft (G \setminus \text{carrier } := A)$

by (*simp add: A2 normal-axioms normal-invE(1) normal-restrict-supergroup*)

have *A3*: $A' = \text{rcosets}_{G \setminus \text{carrier } := A} H$

unfolding *A-def* **using** *factgroup-subgroup-union-factor* *A'normal* *normal-imp-subgroup*

by *auto*

from *A1* **interpret** *normalHA*: *normal* $H \ (G \setminus \text{carrier } := A)$ **by** *metis*

have $H \subseteq A$ **using** *normalHA.is-subgroup* *subgroup.subset* **by** *force*

with *A2* **have** $A = H \vee A = \text{carrier } G$ **using** *maxH.max-normal* **by** *auto*

thus $A' = \text{carrier } (G \text{ Mod } H)$

proof

assume $A = H$

hence $\text{carrier } (G \setminus \text{carrier } := A) \text{ Mod } H = \{\mathbf{1}_{(G \setminus \text{carrier } := A) \text{ Mod } H}\}$

using *cosets-finite subgroup-in-rcosets subset assms normalHA.fact-group-trivial-iff*

by *force*

then have $A' = \{\mathbf{1}_{G \text{ Mod } H}\}$

using *A3* **unfolding** *FactGroup-def* **by** *simp*

with *A'nottriv* **show** *?thesis..*

next

```

    assume A = carrier G
    thus A' = carrier (G Mod H) using A3 unfolding FactGroup-def by simp
  qed
qed
next
assume simple: simple-group (G Mod H)
show max-normal-subgroup H G
proof
  from simple have carrier (G Mod H) ≠ {1G Mod H} unfolding simple-group-def
  simple-group-axioms-def order-def by auto
  with finite fact-group-trivial-iff show H ≠ carrier G by auto
next
fix A
assume A: A ◁ G A ≠ H A ≠ carrier G
show ¬ H ⊆ A
proof
  assume HA: H ⊆ A
  hence H ◁ (G⟨carrier := A⟩) by (metis A(1) inv-op-closed2 is-subgroup
  normal-inv-iff normal-restrict-supergroup)
  then interpret normalHA: normal H (G⟨carrier := A⟩) by simp
  from finite have finiteA: finite A
  by (meson A(1) normal-inv-iff finite-subset subgroup.subset)
  have rcosets(G⟨carrier := A⟩) H ◁ G Mod H
  by (simp add: A(1) HA normal-axioms normality-factorization)
  with simple have rcosets(G⟨carrier := A⟩) H = {1G Mod H} ∨ rcosets(G⟨carrier := A⟩)
  H = carrier (G Mod H)
  unfolding simple-group-def simple-group-axioms-def by auto
  thus False
proof
  assume rcosetsG⟨carrier := A⟩ H = {1G Mod H}
  with finiteA have H = A
  using normalHA.fact-group-trivial-iff unfolding FactGroup-def by auto
  with A(2) show ?thesis by simp
next
assume AHGH: rcosetsG⟨carrier := A⟩ H = carrier (G Mod H)
have A = carrier G unfolding FactGroup-def RCOSETS-def
proof
  show A ⊆ carrier G using A(1) normal-imp-subgroup subgroup.subset by
metis
next
show carrier G ⊆ A
proof
  fix x
  assume x: x ∈ carrier G
  hence H #> x ∈ rcosets H unfolding RCOSETS-def by auto
  with AHGH have H #> x ∈ rcosetsG⟨carrier := A⟩ H unfolding
FactGroup-def by simp
  then obtain x' where x': x' ∈ A H #>x = H #>G⟨carrier := A⟩ x'

```

```

unfolding RCOSETS-def by auto
  hence  $H \#> x = H \#> x'$  unfolding r-coset-def by auto
  hence  $x \in H \#> x'$  by (metis is-subgroup rcos-self x)
  hence  $x \in A \#> x'$  using HA unfolding r-coset-def by auto
  thus  $x \in A$  using  $x'(1)$  unfolding r-coset-def using subgroup.m-closed
A(1) normal-imp-subgroup by force
  qed
  qed
  with A(3) show ?thesis by simp
  qed
  qed
  qed
  qed
end

```

```

theory CompositionSeries
imports
  MaximalNormalSubgroups Secondary-Sylow.SndSylow
begin

hide-const (open) Divisibility.prime

```

2 Normal series and Composition series

2.1 Preliminaries

A subgroup which is unique in cardinality is normal:

```

lemma (in group) unique-sizes-subgrp-normal:
  assumes fin: finite (carrier G)
  assumes  $\exists! Q. Q \in \text{subgroups-of-size } q$ 
  shows (THE Q. Q \in subgroups-of-size q)  $\triangleleft G$ 
proof -
  from assms obtain Q where  $Q \in \text{subgroups-of-size } q$  by auto
  define Q where  $Q = (\text{THE } Q. Q \in \text{subgroups-of-size } q)$ 
  with assms have Qsize: Q \in subgroups-of-size q using theI by metis
  hence QG: subgroup Q G and cardQ: card Q = q unfolding subgroups-of-size-def
by auto
  from QG have  $Q \triangleleft G$  apply(rule normalI)
proof
  fix g
  assume  $g \in \text{carrier } G$ 
  hence invg: inv g \in carrier G by (metis inv-closed)
  with fin Qsize have conjugation-action q (inv g) Q \in subgroups-of-size q by
(metis conjugation-is-size-invariant)
  with g Qsize have  $(\text{inv } g) <\# (Q \#> \text{inv } (\text{inv } g)) \in \text{subgroups-of-size } q$ 
unfolding conjugation-action-def by auto

```

```

with invg g have inv g <# (Q #> g) = Q by (metis Qsize assms(2) inv-inv)
with QG QG g show Q #> g = g <# Q by (rule conj-wo-inv)
qed
with Q-def show ?thesis by simp
qed

```

A group whose order is the product of two distinct primes p and q where $p < q$ has a unique subgroup of size q :

lemma (in group) *pq-order-unique-subgrp*:

```

assumes finite: finite (carrier G)
assumes orderG: order G = q * p
assumes primep: prime p and primeq: prime q and pq: p < q
shows  $\exists! Q. Q \in (\text{subgroups-of-size } q)$ 

```

proof –

```

from primep primeq pq have nqdvdp:  $\neg (q \text{ dvd } p)$  by (metis less-not-refl3
prime-nat-iff)

```

```

define calM where calM = {s. s  $\subseteq$  carrier G  $\wedge$  card s = q ^ 1}

```

```

define RelM where RelM = {(N1, N2). N1  $\in$  calM  $\wedge$  N2  $\in$  calM  $\wedge$  ( $\exists g \in$  carrier
G. N1 = N2 #> g)}

```

```

interpret syl: snd-sylow G q 1 p calM RelM

```

```

unfolding snd-sylow-def sylow-def snd-sylow-axioms-def sylow-axioms-def

```

```

using is-group primeq orderG finite nqdvdp calM-def RelM-def by auto

```

```

obtain Q where Q: Q  $\in$  subgroups-of-size q by (metis (lifting, mono-tags)
mem-Collect-eq power-one-right subgroups-of-size-def syl.sylow-thm)

```

```

thus ?thesis

```

```

proof (rule ex1I)

```

```

fix P

```

```

assume P: P  $\in$  subgroups-of-size q

```

```

have card (subgroups-of-size q) mod q = 1 by (metis power-one-right syl.p-sylow-mod-p)

```

```

moreover have card (subgroups-of-size q) dvd p by (metis power-one-right
syl.num-sylow-dvd-remainder)

```

```

then have card (subgroups-of-size q) = p  $\vee$  card (subgroups-of-size q) = 1

```

```

using primep by (auto simp add: prime-nat-iff)

```

```

ultimately have card (subgroups-of-size q) = 1 using pq

```

```

by auto

```

```

with Q P show P = Q by (auto simp: card-Suc-eq)

```

```

qed

```

```

qed

```

... And this unique subgroup is normal.

corollary (in group) *pq-order-subgrp-normal*:

```

assumes finite: finite (carrier G)

```

```

assumes orderG: order G = q * p

```

```

assumes primep: prime p and primeq: prime q and pq: p < q

```

```

shows (THE Q. Q  $\in$  subgroups-of-size q)  $\triangleleft$  G

```

```

using assms by (metis pq-order-unique-subgrp unique-sizes-subgrp-normal)

```

The trivial subgroup is normal in every group.

lemma (in group) *trivial-subgroup-is-normal*:
 shows $\{1\} \triangleleft G$
unfolding *normal-def normal-axioms-def r-coset-def l-coset-def* **by** (auto intro:
normalI subgroupI simp: is-group)

2.2 Normal Series

We define a normal series as a locale which fixes one group G and a list \mathfrak{G} of subsets of G 's carrier. This list must begin with the trivial subgroup, end with the carrier of the group itself and each of the list items must be a normal subgroup of its successor.

locale *normal-series* = group +
 fixes \mathfrak{G}
 assumes *notempty*: $\mathfrak{G} \neq []$
 assumes *hd*: $\text{hd } \mathfrak{G} = \{1\}$
 assumes *last*: $\text{last } \mathfrak{G} = \text{carrier } G$
 assumes *normal*: $\bigwedge i. i + 1 < \text{length } \mathfrak{G} \implies (\mathfrak{G} ! i) \triangleleft G(\text{carrier} := \mathfrak{G} ! (i + 1))$

lemma (in *normal-series*) *is-normal-series: normal-series* $G \ \mathfrak{G}$ **by** (rule *normal-series-axioms*)

For every group there is a "trivial" normal series consisting only of the group itself and its trivial subgroup.

lemma (in group) *trivial-normal-series*:
 shows *normal-series* $G \ [\{1\}, \text{carrier } G]$
unfolding *normal-series-def normal-series-axioms-def*
using *is-group trivial-subgroup-is-normal* **by** auto

We can also show that the normal series presented above is the only such with a length of two:

lemma (in *normal-series*) *length-two-unique*:
 assumes *length* $\mathfrak{G} = 2$
 shows $\mathfrak{G} = [\{1\}, \text{carrier } G]$
proof(rule *nth-equalityI*)
 from *assms* show *length* $\mathfrak{G} = \text{length } [\{1\}, \text{carrier } G]$ **by** auto
next
 show $\mathfrak{G} ! i = [\{1\}, \text{carrier } G] ! i$ **if** $i: i < \text{length } \mathfrak{G}$ **for** i
proof –
 have $i = 0 \vee i = 1$ **using** *that assms* **by** auto
 thus $\mathfrak{G} ! i = [\{1\}, \text{carrier } G] ! i$
proof(rule *disjE*)
 assume $i: i = 0$
 hence $\mathfrak{G} ! i = \text{hd } \mathfrak{G}$ **by** (*metis hd-conv-nth notempty*)
 thus $\mathfrak{G} ! i = [\{1\}, \text{carrier } G] ! i$ **using** *hd i* **by** *simp*
next
 assume $i: i = 1$
 with *assms* have $\mathfrak{G} ! i = \text{last } \mathfrak{G}$ **by** (*metis diff-add-inverse last-conv-nth nat-1-add-1 notempty*)

```

    thus  $\mathfrak{S} ! i = [\{1\}, \text{carrier } G] ! i$  using last i by simp
  qed
qed
qed

```

We can construct new normal series by expanding existing ones: If we append the carrier of a group G to a normal series for a normal subgroup $H \triangleleft G$ we receive a normal series for G .

```

lemma (in group) normal-series-extend:
  assumes normal: normal-series ( $G \langle \text{carrier} := H \rangle$ )  $\mathfrak{S}$ 
  assumes HG:  $H \triangleleft G$ 
  shows normal-series  $G$  ( $\mathfrak{S} @ [\text{carrier } G]$ )
proof –
  from normal interpret normalH: normal-series ( $G \langle \text{carrier} := H \rangle$ )  $\mathfrak{S}$ .
  from normalH.hd have hd  $\mathfrak{S} = \{1\}$  by simp
  with normalH.notempty have hdTriv: hd ( $\mathfrak{S} @ [\text{carrier } G]$ ) =  $\{1\}$  by (metis
hd-append2)
  show ?thesis unfolding normal-series-def normal-series-axioms-def using is-group
  proof auto
    fix  $x$ 
    assume  $x \in \text{hd } (\mathfrak{S} @ [\text{carrier } G])$ 
    with hdTriv show  $x = 1$  by simp
  next
  from hdTriv show  $1 \in \text{hd } (\mathfrak{S} @ [\text{carrier } G])$  by simp
  next
  fix  $i$ 
  assume  $i: i < \text{length } \mathfrak{S}$ 
  show ( $\mathfrak{S} @ [\text{carrier } G]$ ) !  $i \triangleleft G \langle \text{carrier} := (\mathfrak{S} @ [\text{carrier } G]) ! \text{Suc } i \rangle$ 
  proof (cases  $i + 1 < \text{length } \mathfrak{S}$ )
    case True
    with normalH.normal have  $\mathfrak{S} ! i \triangleleft G \langle \text{carrier} := \mathfrak{S} ! (i + 1) \rangle$  by auto
    with  $i$  have ( $\mathfrak{S} @ [\text{carrier } G]$ ) !  $i \triangleleft G \langle \text{carrier} := \mathfrak{S} ! (i + 1) \rangle$  using
nth-append by metis
    with True show ( $\mathfrak{S} @ [\text{carrier } G]$ ) !  $i \triangleleft G \langle \text{carrier} := (\mathfrak{S} @ [\text{carrier } G]) !$ 
(Suc  $i$ ) using nth-append Suc-eq-plus1 by metis
  next
  case False
  with  $i$  have  $i2: i + 1 = \text{length } \mathfrak{S}$  by simp
  from  $i$  have ( $\mathfrak{S} @ [\text{carrier } G]$ ) !  $i = \mathfrak{S} ! i$  by (metis nth-append)
  also from  $i2$  normalH.notempty have  $\dots = \text{last } \mathfrak{S}$  by (metis add-diff-cancel-right'
last-conv-nth)
  also from normalH.last have  $\dots = H$  by simp
  finally have ( $\mathfrak{S} @ [\text{carrier } G]$ ) !  $i = H$ .
  moreover from  $i2$  have ( $\mathfrak{S} @ [\text{carrier } G]$ ) !  $(i + 1) = \text{carrier } G$  by (metis
nth-append-length)
  ultimately show ?thesis using HG by auto
  qed
qed
qed

```

All entries of a normal series for G are subgroups of G .

lemma (in *normal-series*) *normal-series-subgroups*:

shows $i < \text{length } \mathfrak{G} \implies \text{subgroup } (\mathfrak{G} ! i) G$

proof –

have $i + 1 < \text{length } \mathfrak{G} \implies \text{subgroup } (\mathfrak{G} ! i) G$

proof (induction $\text{length } \mathfrak{G} - (i + 2)$ arbitrary: i)

case 0

hence $i: i + 2 = \text{length } \mathfrak{G}$ **by** *simp*

hence $ii: i + 1 = \text{length } \mathfrak{G} - 1$ **by** *force*

from i *normal* **have** $\mathfrak{G} ! i \triangleleft G(\text{carrier} := \mathfrak{G} ! (i + 1))$ **by** *auto*

with ii *last notempty* **show** $\text{subgroup } (\mathfrak{G} ! i) G$ **using** *last-conv-nth normal-imp-subgroup* **by** *fastforce*

next

case (*Suc k*)

from *Suc(3)* *normal* **have** $i: \text{subgroup } (\mathfrak{G} ! i) (G(\text{carrier} := \mathfrak{G} ! (i + 1)))$

using *normal-imp-subgroup* **by** *auto*

from *Suc(2)* **have** $k: k = \text{length } \mathfrak{G} - ((i + 1) + 2)$ **by** *arith*

with *Suc* **have** $\text{subgroup } (\mathfrak{G} ! (i + 1)) G$ **by** *simp*

with i **show** $\text{subgroup } (\mathfrak{G} ! i) G$

using *incl-subgroup* **by** *blast*

qed

moreover **have** $i + 1 = \text{length } \mathfrak{G} \implies \text{subgroup } (\mathfrak{G} ! i) G$

using *last notempty last-conv-nth* **by** (*metis add-diff-cancel-right' subgroup-self*)

ultimately **show** $i < \text{length } \mathfrak{G} \implies \text{subgroup } (\mathfrak{G} ! i) G$ **by** *force*

qed

The second to last entry of a normal series is a normal subgroup of G .

lemma (in *normal-series*) *normal-series-snd-to-last*:

shows $\mathfrak{G} ! (\text{length } \mathfrak{G} - 2) \triangleleft G$

proof (*cases* $2 \leq \text{length } \mathfrak{G}$)

case *False*

with *notempty* **have** $\text{length}: \text{length } \mathfrak{G} = 1$ **by** (*metis Suc-eq-plus1 leI length-0-conv less-2-cases plus-nat.add-0*)

with *hd* **have** $\mathfrak{G} ! (\text{length } \mathfrak{G} - 2) = \{1\}$ **using** *hd-conv-nth notempty* **by** *auto*

with length **show** *?thesis* **by** (*metis trivial-subgroup-is-normal*)

next

case *True*

hence $(\text{length } \mathfrak{G} - 2) + 1 < \text{length } \mathfrak{G}$ **by** *arith*

with *normal last* **have** $\mathfrak{G} ! (\text{length } \mathfrak{G} - 2) \triangleleft G(\text{carrier} := \mathfrak{G} ! ((\text{length } \mathfrak{G} - 2) + 1))$ **by** *auto*

have $1 + (1 + (\text{length } \mathfrak{G} - (1 + 1))) = \text{length } \mathfrak{G}$

using *True le-add-diff-inverse* **by** *presburger*

then **have** $\mathfrak{G} ! (\text{length } \mathfrak{G} - 2) \triangleleft G(\text{carrier} := \mathfrak{G} ! (\text{length } \mathfrak{G} - 1))$

by (*metis <math>\mathfrak{G} ! (\text{length } \mathfrak{G} - 2) \triangleleft G(\text{carrier} := \mathfrak{G} ! (\text{length } \mathfrak{G} - 2 + 1))>* *add.commute add-diff-cancel-left' one-add-one*)

with *notempty last* **show** *?thesis* **using** *last-conv-nth* **by** *force*

qed

Just like the expansion of normal series, every prefix of a normal series is

again a normal series.

```

lemma (in normal-series) normal-series-prefix-closed:
  assumes  $i \leq \text{length } \mathfrak{G}$  and  $0 < i$ 
  shows normal-series ( $G(\text{carrier} := \mathfrak{G} ! (i - 1))$ ) (take  $i \ \mathfrak{G}$ )
unfolding normal-series-def normal-series-axioms-def
using assms
apply (auto simp: hd del:equalityI)
  apply (simp add: is-group normal-series-subgroups subgroup.subgroup-is-group)
  apply (simp add: last-conv-nth min.absorb2 notempty)
using assms(1) normal apply simp
done

```

If a group's order is the product of two distinct primes p and q , where $p < q$, we can construct a normal series using the only subgroup of size q .

```

lemma (in group) pq-order-normal-series:
  assumes finite: finite (carrier  $G$ )
  assumes orderG: order  $G = q * p$ 
  assumes primep: prime  $p$  and primeq: prime  $q$  and pq:  $p < q$ 
  shows normal-series  $G$  [{1}, (THE  $H$ .  $H \in \text{subgroups-of-size } q$ ), carrier  $G$ ]
proof -
  define  $H$  where  $H = (\text{THE } H. H \in \text{subgroups-of-size } q)$ 
  with assms have  $HG$ :  $H \triangleleft G$  by (metis pq-order-subgrp-normal)
  then interpret groupH: group  $G(\text{carrier} := H)$  unfolding normal-def by
(metis subgroup-imp-group)
  have normal-series ( $G(\text{carrier} := H)$ ) [{1},  $H$ ] using groupH.trivial-normal-series
by auto
  with  $HG$  show ?thesis unfolding H-def by (metis append-Cons append-Nil
normal-series-extend)
qed

```

The following defines the list of all quotient groups of the normal series:

```

definition (in normal-series) quotients
  where quotients = map ( $\lambda i. G(\text{carrier} := \mathfrak{G} ! (i + 1)) \text{ Mod } \mathfrak{G} ! i$ ) [0.. $((\text{length } \mathfrak{G}) - 1)$ ]

```

The list of quotient groups has one less entry than the series itself:

```

lemma (in normal-series) quotients-length:
  shows length quotients + 1 = length  $\mathfrak{G}$ 
proof -
  have length quotients + 1 = length [0.. $((\text{length } \mathfrak{G}) - 1)$ ] + 1 unfolding
quotients-def by simp
  also have ... = (length  $\mathfrak{G} - 1$ ) + 1 by (metis diff-zero length-upt)
  also with notempty have ... = length  $\mathfrak{G}$ 
  by (simp add: ac-simps)
  finally show ?thesis .
qed

```

```

lemma (in normal-series) last-quotient:

```

assumes $\text{length } \mathfrak{G} > 1$
shows $\text{last quotients} = G \text{ Mod } \mathfrak{G} ! (\text{length } \mathfrak{G} - 1 - 1)$
proof –
from *assms* **have** *lsimp*: $\text{length } \mathfrak{G} - 1 - 1 + 1 = \text{length } \mathfrak{G} - 1$ **by** *auto*
from *assms* **have** $\text{quotients} \neq []$ **unfolding** *quotients-def* **by** *auto*
hence $\text{last quotients} = \text{quotients} ! (\text{length } \text{quotients} - 1)$ **by** (*metis last-conv-nth*)
also have $\dots = \text{quotients} ! (\text{length } \mathfrak{G} - 1 - 1)$ **by** (*metis add-diff-cancel-left'*
quotients-length add commute)
also have $\dots = G(\text{carrier} := \mathfrak{G} ! ((\text{length } \mathfrak{G} - 1 - 1) + 1)) \text{ Mod } \mathfrak{G} ! (\text{length } \mathfrak{G} - 1 - 1)$
unfolding *quotients-def* **using** *assms* **by** *auto*
also have $\dots = G(\text{carrier} := \mathfrak{G} ! (\text{length } \mathfrak{G} - 1)) \text{ Mod } \mathfrak{G} ! (\text{length } \mathfrak{G} - 1 - 1)$ **using** *lsimp* **by** *simp*
also have $\dots = G \text{ Mod } \mathfrak{G} ! (\text{length } \mathfrak{G} - 1 - 1)$ **using** *last last-conv-nth notempty*
by *force*
finally show *?thesis* .
qed

The next lemma transports the constituting properties of a normal series along an isomorphism of groups.

lemma (*in normal-series*) *normal-series-iso*:
assumes *H*: *group H*
assumes *iso*: $\Psi \in \text{iso } G H$
shows *normal-series H* (*map (image Ψ) ℑ*)
apply (*simp add: normal-series-def normal-series-axioms-def*)
using *H notempty* **apply** *simp*
proof (*rule conjI*)
from *H is-group iso* **have** *group-hom*: *group-hom G H Ψ* **unfolding** *group-hom-def group-hom-axioms-def iso-def* **by** *auto*
have $\text{hd} (\text{map} (\text{image } \Psi) \mathfrak{G}) = \Psi \text{ ' } \{1\}$ **by** (*metis hd-map hd notempty*)
also have $\dots = \{\Psi 1\}$ **by** (*metis image-empty image-insert*)
also have $\dots = \{1_H\}$ **using** *group-hom group-hom.hom-one* **by** *auto*
finally show $\text{hd} (\text{map} ((\cdot) \Psi) \mathfrak{G}) = \{1_H\}$.
next
show $\text{last} (\text{map} ((\cdot) \Psi) \mathfrak{G}) = \text{carrier } H \wedge (\forall i. \text{Suc } i < \text{length } \mathfrak{G} \longrightarrow \Psi \text{ ' } \mathfrak{G} ! i \triangleleft H(\text{carrier} := \Psi \text{ ' } \mathfrak{G} ! \text{Suc } i))$
proof (*auto del: equalityI*)
have $\text{last} (\text{map} ((\cdot) \Psi) \mathfrak{G}) = \Psi \text{ ' } (\text{carrier } G)$ **using** *last last-map notempty* **by** *metis*
also have $\dots = \text{carrier } H$ **using** *iso unfolding iso-def bij-betw-def* **by** *simp*
finally show $\text{last} (\text{map} ((\cdot) \Psi) \mathfrak{G}) = \text{carrier } H$.
next
fix *i*
assume *i*: $\text{Suc } i < \text{length } \mathfrak{G}$
hence *norm*: $\mathfrak{G} ! i \triangleleft G(\text{carrier} := \mathfrak{G} ! \text{Suc } i)$ **using** *normal* **by** *simp*
moreover have $\text{restrict } \Psi (\mathfrak{G} ! \text{Suc } i) \in \text{iso} (G(\text{carrier} := \mathfrak{G} ! \text{Suc } i))$
 $(H(\text{carrier} := \Psi \text{ ' } \mathfrak{G} ! \text{Suc } i))$
by (*metis H i is-group iso iso-restrict normal-series-subgroups*)
moreover have $\text{group} (G(\text{carrier} := \mathfrak{G} ! \text{Suc } i))$ **by** (*metis i normal-series-subgroups*)

subgroup-imp-group)
moreover hence *subgroup* ($\mathfrak{G} ! \text{Suc } i$) G **by** (*metis* i *normal-series-subgroups*)
hence *subgroup* ($\Psi \text{ ' } \mathfrak{G} ! \text{Suc } i$) H
by (*simp add*: H *iso subgroup.iso-subgroup*)
hence *group* ($H \langle \text{carrier} := \Psi \text{ ' } \mathfrak{G} ! \text{Suc } i \rangle$) **by** (*metis* H *subgroup.subgroup-is-group*)
ultimately have *restrict* $\Psi \text{ ' } \mathfrak{G} ! \text{Suc } i$ $\text{ ' } \mathfrak{G} ! i \triangleleft H \langle \text{carrier} := \Psi \text{ ' } \mathfrak{G} ! \text{Suc } i \rangle$
using *is-group* H *iso-normal-subgroup* **by** (*auto cong del: image-cong-simp*)
moreover from *norm* **have** $\mathfrak{G} ! i \subseteq \mathfrak{G} ! \text{Suc } i$ **unfolding** *normal-def subgroup-def* **by** *auto*
hence $\{y. \exists x \in \mathfrak{G} ! i. y = (\text{if } x \in \mathfrak{G} ! \text{Suc } i \text{ then } \Psi x \text{ else undefined})\} = \{y. \exists x \in \mathfrak{G} ! i. y = \Psi x\}$ **by** *auto*
ultimately show $\Psi \text{ ' } \mathfrak{G} ! i \triangleleft H \langle \text{carrier} := \Psi \text{ ' } \mathfrak{G} ! \text{Suc } i \rangle$ **unfolding** *restrict-def image-def* **by** *auto*
qed
qed

2.3 Composition Series

A composition series is a normal series where all consecutive factor groups are simple:

locale *composition-series = normal-series +*
assumes *simplefact*: $\bigwedge i. i + 1 < \text{length } \mathfrak{G} \implies \text{simple-group } (G \langle \text{carrier} := \mathfrak{G} ! (i + 1) \rangle \text{ Mod } \mathfrak{G} ! i)$

lemma (*in composition-series*) *is-composition-series*:
shows *composition-series* G \mathfrak{G}
by (*rule composition-series-axioms*)

A composition series for a group G has length one if and only if G is the trivial group.

lemma (*in composition-series*) *composition-series-length-one*:
shows $(\text{length } \mathfrak{G} = 1) = (\mathfrak{G} = [\{1\}])$
proof
assume $\text{length } \mathfrak{G} = 1$
with *hd* **have** $\text{length } \mathfrak{G} = \text{length } [\{1\}] \wedge (\forall i < \text{length } \mathfrak{G}. \mathfrak{G} ! i = [\{1\}] ! i)$ **using** *hd-conv-nth notempty* **by** *force*
thus $\mathfrak{G} = [\{1\}]$ **using** *list-eq-iff-nth-eq* **by** *blast*
next
assume $\mathfrak{G} = [\{1\}]$
thus $\text{length } \mathfrak{G} = 1$ **by** *simp*
qed

lemma (*in composition-series*) *composition-series-triv-group*:
shows $(\text{carrier } G = \{1\}) = (\mathfrak{G} = [\{1\}])$
proof
assume $G: \text{carrier } G = \{1\}$
have $\text{length } \mathfrak{G} = 1$
proof (*rule ccontr*)

```

    assume length  $\mathfrak{G} \neq 1$ 
    with notempty have length: length  $\mathfrak{G} \geq 2$  by (metis Suc-eq-plus1 length-0-conv
less-2-cases not-less plus-nat.add-0)
    with simplefact hd hd-conv-nth notempty have simple-group (G⟦carrier :=  $\mathfrak{G}
! 1$ ⟧ Mod {1}) by force
    moreover have SG: subgroup ( $\mathfrak{G} ! 1$ ) G using length normal-series-subgroups
by auto
    hence group (G⟦carrier :=  $\mathfrak{G} ! 1$ ⟧) by (metis subgroup-imp-group)
    ultimately have simple-group (G⟦carrier :=  $\mathfrak{G} ! 1$ ⟧) using group.trivial-factor-iso
simple-group.iso-simple by fastforce
    moreover from SG G have carrier (G⟦carrier :=  $\mathfrak{G} ! 1$ ⟧) = {1} unfolding
subgroup-def by auto
    ultimately show False using simple-group.simple-not-triv by force
qed
    thus  $\mathfrak{G} = [\{1\}]$  by (metis composition-series-length-one)
next
    assume  $\mathfrak{G} = [\{1\}]$ 
    with last show carrier G = {1} by auto
qed

```

The inner elements of a composition series may not consist of the trivial subgroup or the group itself.

lemma (in *composition-series*) *inner-elements-not-triv*:

```

    assumes  $i + 1 < \text{length } \mathfrak{G}$ 
    assumes  $i > 0$ 
    shows  $\mathfrak{G} ! i \neq \{1\}$ 
proof
    from assms have  $(i - 1) + 1 < \text{length } \mathfrak{G}$  by simp
    hence simple: simple-group (G⟦carrier :=  $\mathfrak{G} ! ((i - 1) + 1)$ ⟧ Mod  $\mathfrak{G} ! (i - 1)$ )
using simplefact by auto
    assume  $i: \mathfrak{G} ! i = \{1\}$ 
    moreover from assms have  $(i - 1) + 1 = i$  by auto
    ultimately have G⟦carrier :=  $\mathfrak{G} ! ((i - 1) + 1)$ ⟧ Mod  $\mathfrak{G} ! (i - 1) = G⟦carrier
:= \{1\}$ ⟧ Mod  $\mathfrak{G} ! (i - 1)$  using  $i$  by auto
    hence order (G⟦carrier :=  $\mathfrak{G} ! ((i - 1) + 1)$ ⟧ Mod  $\mathfrak{G} ! (i - 1)$ ) = 1 unfolding
FactGroup-def order-def RCOSETS-def by force
    thus False using  $i$  simple unfolding simple-group-def simple-group-axioms-def
by auto
qed

```

A composition series of a simple group always is its trivial one.

lemma (in *composition-series*) *composition-series-simple-group*:

```

    shows (simple-group G) = ( $\mathfrak{G} = [\{1\}, \text{carrier } G]$ )
proof
    assume  $\mathfrak{G} = [\{1\}, \text{carrier } G]$ 
    with simplefact have simple-group (G Mod {1}) by auto
    moreover have the-elem  $\in$  iso (G Mod {1}) G by (rule trivial-factor-iso)
    ultimately show simple-group G by (metis is-group simple-group.iso-simple)
next

```

```

assume simple: simple-group  $G$ 
have  $\text{length } \mathfrak{G} > 1$ 
proof (rule ccontr)
  assume  $\neg 1 < \text{length } \mathfrak{G}$ 
  hence  $\text{length } \mathfrak{G} = 1$  by (simp add: Suc-leI antisym notempty)
  hence  $\text{carrier } G = \{1\}$  using hd last by (metis composition-series-length-one
composition-series-triv-group)
  hence  $\text{order } G = 1$  unfolding order-def by auto
  with simple show False unfolding simple-group-def simple-group-axioms-def
by auto
qed
moreover have  $\text{length } \mathfrak{G} \leq 2$ 
proof (rule ccontr)
  define  $k$  where  $k = \text{length } \mathfrak{G} - 2$ 
  assume  $\neg (\text{length } \mathfrak{G} \leq 2)$ 
  hence gt2:  $\text{length } \mathfrak{G} > 2$  by simp
  hence ksmall:  $k + 1 < \text{length } \mathfrak{G}$  unfolding k-def by auto
  from gt2 have  $\text{carrier: } \mathfrak{G} ! (k + 1) = \text{carrier } G$  using notempty last last-conv-nth
k-def
  by (metis Nat.add-diff-assoc Nat.diff-cancel <\neg length \mathfrak{G} \le 2> add.commute
nat-le-linear one-add-one)
  from normal ksmall have  $\mathfrak{G} ! k \triangleleft G \langle \text{carrier} := \mathfrak{G} ! (k + 1) \rangle$  by simp
  from simplefact ksmall have simplek: simple-group ( $G \langle \text{carrier} := \mathfrak{G} ! (k + 1) \rangle$ )
Mod  $\mathfrak{G} ! k$  by simp
  from simplefact ksmall have simplek': simple-group ( $G \langle \text{carrier} := \mathfrak{G} ! ((k -$ 
 $1) + 1) \rangle \text{Mod } \mathfrak{G} ! (k - 1))$  by auto
  have  $\mathfrak{G} ! k \triangleleft G$  using carrier k-def gt2 normal ksmall by force
  with simple have  $(\mathfrak{G} ! k) = \text{carrier } G \vee (\mathfrak{G} ! k) = \{1\}$  unfolding sim-
ple-group-def simple-group-axioms-def by simp
  thus False
  proof (rule disjE)
    assume  $\mathfrak{G} ! k = \text{carrier } G$ 
    hence  $G \langle \text{carrier} := \mathfrak{G} ! (k + 1) \rangle \text{Mod } \mathfrak{G} ! k = G \text{Mod } (\text{carrier } G)$  using
carrier by auto
    with simplek self-factor-not-simple show False by auto
  next
    assume  $\mathfrak{G} ! k = \{1\}$ 
    with ksmall k-def gt2 show False using inner-elements-not-triv by auto
  qed
qed
ultimately have  $\text{length } \mathfrak{G} = 2$  by simp
thus  $\mathfrak{G} = [\{1\}, \text{carrier } G]$  by (rule length-two-unique)
qed

```

Two consecutive elements in a composition series are distinct.

lemma (in *composition-series*) *entries-distinct*:

```

assumes finite: finite (carrier  $G$ )
assumes  $i: i + 1 < \text{length } \mathfrak{G}$ 
shows  $\mathfrak{G} ! i \neq \mathfrak{G} ! (i + 1)$ 

```

```

proof
  from finite have finite ( $\mathfrak{G} ! (i + 1)$ )
    using i normal-series-subgroups subgroup.subset rev-finite-subset by metis
  hence fin: finite (carrier ( $G \langle \text{carrier} := \mathfrak{G} ! (i + 1) \rangle$ )) by auto
  from i have norm:  $\mathfrak{G} ! i \triangleleft (G \langle \text{carrier} := \mathfrak{G} ! (i + 1) \rangle)$  by (rule normal)
  assume  $\mathfrak{G} ! i = \mathfrak{G} ! (i + 1)$ 
  hence  $\mathfrak{G} ! i = \text{carrier} (G \langle \text{carrier} := \mathfrak{G} ! (i + 1) \rangle)$  by auto
  hence carrier ( $(G \langle \text{carrier} := (\mathfrak{G} ! (i + 1)) \rangle) \text{Mod } (\mathfrak{G} ! i)$ ) =  $\{1_{(G \langle \text{carrier} := \mathfrak{G} ! (i + 1) \rangle) \text{Mod } \mathfrak{G} ! i}\}$ 
    using norm fin normal.fact-group-trivial-iff by metis
  hence  $\neg$  simple-group ( $(G \langle \text{carrier} := (\mathfrak{G} ! (i + 1)) \rangle) \text{Mod } (\mathfrak{G} ! i)$ ) by (metis simple-group.simple-not-triv)
  thus False by (metis i simplefact)
qed

```

The normal series for groups of order $p * q$ is even a composition series:

```

lemma (in group) pq-order-composition-series:
  assumes finite: finite (carrier  $G$ )
  assumes orderG: order  $G = q * p$ 
  assumes primep: prime  $p$  and primeq: prime  $q$  and pq:  $p < q$ 
  shows composition-series  $G \{1\}$ , (THE  $H$ .  $H \in \text{subgroups-of-size } q$ ), carrier  $G$ 
unfolding composition-series-def composition-series-axioms-def
apply(auto)
using assms apply(rule pq-order-normal-series)
proof -
  define  $H$  where  $H = (\text{THE } H$ .  $H \in \text{subgroups-of-size } q)$ 
  from assms have exi:  $\exists ! Q$ .  $Q \in (\text{subgroups-of-size } q)$  by (auto simp: pq-order-unique-subgrp)
  hence Hsize:  $H \in \text{subgroups-of-size } q$  unfolding H-def using theI' by metis
  hence HsubG: subgroup  $H G$  unfolding subgroups-of-size-def by auto
  then interpret Hgroup: group  $G \langle \text{carrier} := H \rangle$  by (metis subgroup-imp-group)
  fix  $i$ 
  assume  $i < \text{Suc } (\text{Suc } 0)$ 
  hence  $i = 0 \vee i = 1$  by auto
  thus simple-group ( $G \langle \text{carrier} := [H, \text{carrier } G] ! i \rangle \text{Mod } \{1\}$ ,  $H$ , carrier  $G$  !  $i$ )
  proof
    assume  $i = 0$ 
    from Hsize have orderH: order ( $G \langle \text{carrier} := H \rangle$ ) =  $q$  unfolding subgroups-of-size-def order-def by simp
    hence order-eq-q: order ( $G \langle \text{carrier} := H \rangle \text{Mod } \{1\}$ ) =  $q$ 
      using Hgroup.trivial-factor-iso iso-same-order by auto
    have normal  $\{1\}$  ( $G \langle \text{carrier} := H \rangle$ )
      by (simp add: HsubG group.normal-restrict-supergroup subgroup.one-closed trivial-subgroup-is-normal)
    hence group ( $G \langle \text{carrier} := H \rangle \text{Mod } \{1\}$ ) by (metis normal.factorgroup-is-group)
    with orderH primeq have simple-group ( $G \langle \text{carrier} := H \rangle \text{Mod } \{1\}$ )
      by (metis order-eq-q group.prime-order-simple)
    with  $i$  show ?thesis by simp
  next
    assume  $i = 1$ 
    from assms exi have  $H \triangleleft G$  unfolding H-def by (metis pq-order-subgrp-normal)

```

```

hence groupGH: group (G Mod H) by (metis normal.factorgroup-is-group)
from primeq have q ≠ 0 by (metis not-prime-0)
from HsubG finite orderG have card (rcosets H) * card H = q * p unfolding
subgroups-of-size-def using lagrange by simp
with Hsize have card (rcosets H) * q = q * p unfolding subgroups-of-size-def
by simp
with ⟨q ≠ 0⟩ have card (rcosets H) = p by auto
hence order (G Mod H) = p unfolding order-def FactGroup-def by auto
with groupGH primep have simple-group (G Mod H) by (metis group.prime-order-simple)
with i show ?thesis by auto
qed
qed

```

Prefixes of composition series are also composition series.

```

lemma (in composition-series) composition-series-prefix-closed:
  assumes i ≤ length  $\mathfrak{G}$  and 0 < i
  shows composition-series (G⟦carrier :=  $\mathfrak{G} ! (i - 1)$ ⟧) (take i  $\mathfrak{G}$ )
unfolding composition-series-def composition-series-axioms-def
proof auto
  from assms show normal-series (G⟦carrier :=  $\mathfrak{G} ! (i - Suc\ 0)$ ⟧) (take i  $\mathfrak{G}$ ) by
(metis One-nat-def normal-series-prefix-closed)
next
  fix j
  assume j: Suc j < length  $\mathfrak{G}$  Suc j < i
  with simplefact show simple-group (G⟦carrier :=  $\mathfrak{G} ! Suc\ j$ ⟧ Mod  $\mathfrak{G} ! j$ ) by
(metis Suc-eq-plus1)
qed

```

The second element in a composition series is simple group.

```

lemma (in composition-series) composition-series-snd-simple:
  assumes 2 ≤ length  $\mathfrak{G}$ 
  shows simple-group (G⟦carrier :=  $\mathfrak{G} ! 1$ ⟧)
proof -
  from assms interpret compTake: composition-series G⟦carrier :=  $\mathfrak{G} ! 1$ ⟧ take
2  $\mathfrak{G}$  by (metis add-diff-cancel-right' composition-series-prefix-closed one-add-one
zero-less-numeral)
  from assms have length (take 2  $\mathfrak{G}$ ) = 2 by (metis add-diff-cancel-right' ap-
pend-take-drop-id diff-diff-cancel length-append length-drop)
  hence (take 2  $\mathfrak{G}$ ) = [ $\mathbf{1}_{(G⟦carrier := \mathfrak{G} ! 1)$ }, carrier (G⟦carrier :=  $\mathfrak{G} ! 1$ ⟧)]
by (rule compTake.length-two-unique)
  thus ?thesis by (metis compTake.composition-series-simple-group)
qed

```

As a stronger way to state the previous lemma: An entry of a composition series is simple if and only if it is the second one.

```

lemma (in composition-series) composition-snd-simple-iff:
  assumes i < length  $\mathfrak{G}$ 
  shows (simple-group (G⟦carrier :=  $\mathfrak{G} ! i$ ⟧)) = (i = 1)

```

```

proof
  assume simp1: simple-group ( $G \setminus \text{carrier} := \mathfrak{G} ! i$ )
  hence  $\mathfrak{G} ! i \neq \{1\}$  using simple-group.simple-not-triv by force
  hence  $i \neq 0$  using hd hd-conv-nth notempty by auto
  then interpret compTake: composition-series  $G \setminus \text{carrier} := \mathfrak{G} ! i$ ) take (Suc  $i$ )
 $\mathfrak{G}$ 
  using assms composition-series-prefix-closed by (metis diff-Suc-1 less-eq-Suc-le zero-less-Suc)
  from simp1 have (take (Suc  $i$ )  $\mathfrak{G}$ ) = [ $\{1_{G \setminus \text{carrier} := \mathfrak{G} ! i}\}$ , carrier ( $G \setminus \text{carrier} := \mathfrak{G} ! i$ )]
  by (metis compTake.composition-series-simple-group)
  hence length (take (Suc  $i$ )  $\mathfrak{G}$ ) = 2 by auto
  hence min (length  $\mathfrak{G}$ ) (Suc  $i$ ) = 2 by (metis length-take)
  with assms have Suc  $i$  = 2 by force
  thus  $i = 1$  by simp
next
  assume  $i: i = 1$ 
  with assms have  $2 \leq \text{length } \mathfrak{G}$  by simp
  with  $i$  show simple-group ( $G \setminus \text{carrier} := \mathfrak{G} ! i$ ) by (metis composition-series-snd-simple)
qed

```

The second to last entry of a normal series is not only a normal subgroup but actually even a *maximal* normal subgroup.

lemma (*in composition-series*) *snd-to-last-max-normal*:

```

assumes finite: finite (carrier  $G$ )
assumes length: length  $\mathfrak{G} > 1$ 
shows max-normal-subgroup ( $\mathfrak{G} ! (\text{length } \mathfrak{G} - 2)$ )  $G$ 
unfolding max-normal-subgroup-def max-normal-subgroup-axioms-def
proof (auto del: equalityI)
  show  $\mathfrak{G} ! (\text{length } \mathfrak{G} - 2) \triangleleft G$  by (rule normal-series-snd-to-last)
next
  define  $G'$  where  $G' = \mathfrak{G} ! (\text{length } \mathfrak{G} - 2)$ 
  from length have length21:  $\text{length } \mathfrak{G} - 2 + 1 = \text{length } \mathfrak{G} - 1$  by arith
  from length have  $\text{length } \mathfrak{G} - 2 + 1 < \text{length } \mathfrak{G}$  by arith
  with simplefact have simple-group ( $G \setminus \text{carrier} := \mathfrak{G} ! ((\text{length } \mathfrak{G} - 2) + 1)$ )
  Mod  $G'$ ) unfolding  $G'$ -def by auto
  with length21 have simple-last: simple-group ( $G \text{ Mod } G'$ ) using last notempty last-conv-nth by fastforce
  {
    assume snd-to-last-eq:  $G' = \text{carrier } G$ 
    hence carrier ( $G \text{ Mod } G'$ ) =  $\{1_{G \text{ Mod } G'}\}$ 
    using normal-series-snd-to-last finite normal.fact-group-trivial-iff unfolding
     $G'$ -def by metis
    with snd-to-last-eq have  $\neg \text{simple-group}$  ( $G \text{ Mod } G'$ ) by (metis self-factor-not-simple)
    with simple-last show False unfolding  $G'$ -def by auto
  }
  {
    have  $G'G$ :  $G' \triangleleft G$  unfolding  $G'$ -def by (rule normal-series-snd-to-last)
    fix  $J$ 

```

```

assume J:  $J \triangleleft G$   $J \neq G'$   $J \neq \text{carrier } G$   $G' \subseteq J$ 
hence  $JG'GG'$ :  $\text{rcosets}_{G(\text{carrier} := J)} G' \triangleleft G \text{ Mod } G'$  using normal-
ity-factorization normal-series-snd-to-last unfolding  $G'$ -def by auto
from  $G'G$   $J(1,4)$  have  $G'J$ :  $G' \triangleleft (G(\text{carrier} := J))$  by (metis normal-imp-subgroup
normal-restrict-supergroup)
from finite  $J(1)$  have  $\text{fin}J$ : finite  $J$  by (auto simp: normal-imp-subgroup
subgroup-finite)
from  $JG'GG'$  simple-last have  $\text{rcosets}_{G(\text{carrier} := J)} G' = \{\mathbf{1}_{G \text{ Mod } G'}\} \vee$ 
 $\text{rcosets}_{G(\text{carrier} := J)} G' = \text{carrier } (G \text{ Mod } G')$ 
unfolding simple-group-def simple-group-axioms-def by auto
thus False
proof
assume  $\text{rcosets}_{G(\text{carrier} := J)} G' = \{\mathbf{1}_{G \text{ Mod } G'}\}$ 
hence  $\text{rcosets}_{G(\text{carrier} := J)} G' = \{\mathbf{1}_{(G(\text{carrier} := J)) \text{ Mod } G'}\}$  unfolding
FactGroup-def by simp
hence  $G' = J$  using  $G'J$   $\text{fin}J$  normal.fact-group-trivial-iff unfolding Fact-
Group-def by fastforce
with  $J(2)$  show False by simp
next
assume facts-eq:  $\text{rcosets}_{G(\text{carrier} := J)} G' = \text{carrier } (G \text{ Mod } G')$ 
have  $J = \text{carrier } G$ 
proof
show  $J \subseteq \text{carrier } G$  using  $J(1)$  normal-imp-subgroup subgroup.subset by
force
next
show  $\text{carrier } G \subseteq J$ 
proof
fix  $x$ 
assume  $x$ :  $x \in \text{carrier } G$ 
hence  $G' \#> x \in \text{carrier } (G \text{ Mod } G')$  unfolding FactGroup-def
RCOSETS-def by auto
hence  $G' \#> x \in \text{rcosets}_{G(\text{carrier} := J)} G'$  using facts-eq by auto
then obtain  $j$  where  $j$ :  $j \in J$   $G' \#> x = G' \#> j$  unfolding RCOSETS-def
r-coset-def by force
hence  $x \in G' \#> j$  using  $G'G$  normal-imp-subgroup  $x$  repr-independenceD
by fastforce
then obtain  $g'$  where  $g'$ :  $g' \in G'$   $x = g' \otimes j$  unfolding r-coset-def by
auto
hence  $g' \in J$  using  $G'J$  normal-imp-subgroup subgroup.subset by force
with  $g'(2)$   $j(1)$  show  $x \in J$  using  $J(1)$  normal-imp-subgroup sub-
group.m-closed by fastforce
qed
qed
with  $J(3)$  show False by simp
qed
}
qed

```

For the next lemma we need a few facts about removing adjacent duplicates.

lemma *remdups-adj-obtain-adjacency*:

assumes $i + 1 < \text{length } (\text{remdups-adj } xs)$ $\text{length } xs > 0$

obtains j **where** $j + 1 < \text{length } xs$

$(\text{remdups-adj } xs) ! i = xs ! j$ $(\text{remdups-adj } xs) ! (i + 1) = xs ! (j + 1)$

using *assms* **proof** (*induction xs arbitrary: i thesis*)

case *Nil*

hence *False* **by** (*metis length-greater-0-conv*)

thus *thesis..*

next

case (*Cons x xs*)

then have $xs \neq []$

by *auto*

then obtain $y \ xs'$ **where** $xs: xs = y \# \ xs'$

by (*cases xs*) *blast*

from $\langle xs \neq [] \rangle$ **have** $lenxs: \text{length } xs > 0$ **by** *simp*

from xs **have** $rem: \text{remdups-adj } (x \# \ xs) = (\text{if } x = y \text{ then } \text{remdups-adj } (y \# \ xs') \text{ else } x \# \ \text{remdups-adj } (y \# \ xs'))$ **using** *remdups-adj.simps(3)* **by** *auto*

show *thesis*

proof (*cases x = y*)

case *True*

with $rem \ xs$ **have** $rem2: \text{remdups-adj } (x \# \ xs) = \text{remdups-adj } xs$ **by** *auto*

with *Cons(3)* **have** $i + 1 < \text{length } (\text{remdups-adj } xs)$ **by** *simp*

with *Cons.IH lenxs* **obtain** k **where** $j: k + 1 < \text{length } xs$ $\text{remdups-adj } xs ! i = xs ! k$

$\text{remdups-adj } xs ! (i + 1) = xs ! (k + 1)$ **by** *auto*

thus *thesis* **using** *Cons(2) rem2* **by** *auto*

next

case *False*

with $rem \ xs$ **have** $rem2: \text{remdups-adj } (x \# \ xs) = x \# \ \text{remdups-adj } xs$ **by** *auto*

show *thesis*

proof (*cases i*)

case 0

have $0 + 1 < \text{length } (x \# \ xs)$ **using** $lenxs$ **by** *auto*

moreover have $\text{remdups-adj } (x \# \ xs) ! i = (x \# \ xs) ! 0$

proof –

have $\text{remdups-adj } (x \# \ xs) ! i = (x \# \ \text{remdups-adj } (y \# \ xs')) ! 0$ **using** xs $rem2 \ 0$ **by** *simp*

also have $\dots = x$ **by** *simp*

also have $\dots = (x \# \ xs) ! 0$ **by** *simp*

finally show *?thesis*.

qed

moreover have $\text{remdups-adj } (x \# \ xs) ! (i + 1) = (x \# \ xs) ! (0 + 1)$

proof –

have $\text{remdups-adj } (x \# \ xs) ! (i + 1) = (x \# \ \text{remdups-adj } (y \# \ xs')) ! 1$

using $xs \ rem2 \ 0$ **by** *simp*

also have $\dots = \text{remdups-adj } (y \# \ xs') ! 0$ **by** *simp*

also have $\dots = (y \# \ (\text{remdups } (y \# \ xs'))) ! 0$ **by** (*metis nth-Cons' remdups-adj-Cons-alt*)

also have $\dots = y$ by *simp*
 also have $\dots = (x \# xs) ! (0 + 1)$ unfolding *xs* by *simp*
 finally show *?thesis*.
 qed
 ultimately show *thesis* by (rule *Cons.prem1*)
 next
 case (*Suc k*)
 with *Cons(3)* have $k + 1 < \text{length} (\text{remdups-adj } (x \# xs)) - 1$ by *auto*
 also have $\dots \leq \text{length} (\text{remdups-adj } xs) + 1 - 1$ by (*metis One-nat-def*
le-refl list.size(4) rem2)
 also have $\dots = \text{length} (\text{remdups-adj } xs)$ by *simp*
 finally have $k + 1 < \text{length} (\text{remdups-adj } xs)$.
 with *Cons.IH lenxs* obtain *j* where *j*: $j + 1 < \text{length } xs \text{ remdups-adj } xs ! k$
 $= xs ! j$
 $\text{remdups-adj } xs ! (k + 1) = xs ! (j + 1)$ by *auto*
 from *j(1)* have *Suc j + 1* $< \text{length} (x \# xs)$ by *simp*
 moreover have $\text{remdups-adj } (x \# xs) ! i = (x \# xs) ! (\text{Suc } j)$
 proof –
 have $\text{remdups-adj } (x \# xs) ! i = (x \# \text{remdups-adj } xs) ! i$ using *rem2* by
simp
 also have $\dots = (\text{remdups-adj } xs) ! k$ using *Suc* by *simp*
 also have $\dots = xs ! j$ using *j(2)*.
 also have $\dots = (x \# xs) ! (\text{Suc } j)$ by *simp*
 finally show *?thesis*.
 qed
 moreover have $\text{remdups-adj } (x \# xs) ! (i + 1) = (x \# xs) ! (\text{Suc } j + 1)$
 proof –
 have $\text{remdups-adj } (x \# xs) ! (i + 1) = (x \# \text{remdups-adj } xs) ! (i + 1)$
 using *rem2* by *simp*
 also have $\dots = (\text{remdups-adj } xs) ! (k + 1)$ using *Suc* by *simp*
 also have $\dots = xs ! (j + 1)$ using *j(3)*.
 also have $\dots = (x \# xs) ! (\text{Suc } j + 1)$ by *simp*
 finally show *?thesis*.
 qed
 ultimately show *thesis* by (rule *Cons.prem1*)
 qed
 qed
 qed

lemma *hd-remdups-adj[simp]*: $\text{hd} (\text{remdups-adj } xs) = \text{hd } xs$
 by (*induction xs rule: remdups-adj.induct*) *simp-all*

lemma *remdups-adj-adjacent*:

$\text{Suc } i < \text{length} (\text{remdups-adj } xs) \implies \text{remdups-adj } xs ! i \neq \text{remdups-adj } xs ! \text{Suc } i$

proof (*induction xs arbitrary: i rule: remdups-adj.induct*)

case (*3 x y xs i*)

thus *?case* by (*cases i, cases x = y*) (*simp, auto simp: hd-conv-nth[symmetric]*)

qed *simp-all*

Intersecting each entry of a composition series with a normal subgroup of G

and removing all adjacent duplicates yields another composition series.

lemma (in *composition-series*) *intersect-normal*:

assumes *finite*: *finite* (*carrier* *G*)

assumes *KG*: $K \triangleleft G$

shows *composition-series* ($G \setminus \text{carrier} := K$) (*remdups-adj* (*map* ($\lambda H. K \cap H$) \mathfrak{G}))

unfolding *composition-series-def* *composition-series-axioms-def* *normal-series-def* *normal-series-axioms-def*

apply (*auto simp only: conjI del: equalityI*)

proof –

show *group* ($G \setminus \text{carrier} := K$) **using** *KG normal-imp-subgroup subgroup-imp-group*
by *auto*

next

– Show, that removing adjacent duplicates doesn't result in an empty list.

assume *remdups-adj* (*map* ($((\cap) K) \mathfrak{G}$) = []

hence *map* ($((\cap) K) \mathfrak{G}$) = [] **by** (*metis remdups-adj-Nil-iff*)

hence \mathfrak{G} = [] **by** (*metis Nil-is-map-conv*)

with *notempty* **show** *False..*

next

– Show, that the head of the reduced list is still the trivial group

have $\mathfrak{G} = \{1\} \# \text{tl } \mathfrak{G}$ **using** *notempty hd* **by** (*metis list.sel(1,3) neq-Nil-conv*)

hence *map* ($((\cap) K) \mathfrak{G}$) = *map* ($((\cap) K)$ ($\{1\} \# \text{tl } \mathfrak{G}$)) **by** *simp*

hence *remdups-adj* (*map* ($((\cap) K) \mathfrak{G}$)) = *remdups-adj* ($(K \cap \{1\}) \# (\text{map } ((\cap) K) (\text{tl } \mathfrak{G}))$) **by** *simp*

also have $\dots = (K \cap \{1\}) \# \text{tl } (\text{remdups-adj } ((K \cap \{1\}) \# (\text{map } ((\cap) K) (\text{tl } \mathfrak{G}))))$ **by** *simp*

finally have *hd* (*remdups-adj* (*map* ($((\cap) K) \mathfrak{G}$))) = $K \cap \{1\}$ **using** *list.sel(1)*
by *metis*

thus *hd* (*remdups-adj* (*map* ($((\cap) K) \mathfrak{G}$))) = $\{1\}_{G \setminus \text{carrier} := K}$

using *KG normal-imp-subgroup subgroup.one-closed* **by** *force*

next

– Show that the last entry is really $K \cap G$. Since we don't have a lemma ready to talk about the last entry of a reduced list, we reverse the list twice.

have *rev* \mathfrak{G} = (*carrier* *G*) $\# \text{tl } (\text{rev } \mathfrak{G})$ **by** (*metis list.sel(1,3) last last-rev neq-Nil-conv notempty rev-is-Nil-conv rev-rev-ident*)

hence *rev* (*map* ($((\cap) K) \mathfrak{G}$)) = *map* ($((\cap) K)$ ($(\text{carrier } G) \# \text{tl } (\text{rev } \mathfrak{G})$)) **by** (*metis rev-map*)

hence *rev*: *rev* (*map* ($((\cap) K) \mathfrak{G}$)) = $(K \cap (\text{carrier } G)) \# (\text{map } ((\cap) K) (\text{tl } (\text{rev } \mathfrak{G})))$ **by** *simp*

have *last* (*remdups-adj* (*map* ($((\cap) K) \mathfrak{G}$))) = *hd* (*rev* (*remdups-adj* (*map* ($((\cap) K) \mathfrak{G}$))))

by (*metis hd-rev map-is-Nil-conv notempty remdups-adj-Nil-iff*)

also have $\dots = \text{hd } (\text{remdups-adj } (\text{rev } (\text{map } ((\cap) K) \mathfrak{G})))$ **by** (*metis remdups-adj-rev*)

also have $\dots = \text{hd } (\text{remdups-adj } ((K \cap (\text{carrier } G)) \# (\text{map } ((\cap) K) (\text{tl } (\text{rev } \mathfrak{G}))))$ **by** (*metis rev*)

also have $\dots = \text{hd } ((K \cap (\text{carrier } G)) \# (\text{remdups-adj } ((K \cap (\text{carrier } G)) \# (\text{map } ((\cap) K) (\text{tl } (\text{rev } \mathfrak{G}))))))$ **by** (*metis list.sel(1) remdups-adj-Cons-alt*)

also have $\dots = K$ **using** *KG normal-imp-subgroup subgroup.subset* **by** *force*

finally show *last* (*remdups-adj* (*map* ($((\cap) K) \mathfrak{G}$))) = *carrier* ($G \setminus \text{carrier} := K$)

by auto
next
— The induction step, using the second isomorphism theorem for groups.
fix j
assume j: $j + 1 < \text{length } (\text{remdups-adj } (\text{map } ((\cap) K) \mathfrak{G}))$
have KGnotempty: $(\text{map } ((\cap) K) \mathfrak{G}) \neq []$ using notempty by (metis Nil-is-map-conv)
with j obtain i where i: $i + 1 < \text{length } (\text{map } ((\cap) K) \mathfrak{G})$
 $(\text{remdups-adj } (\text{map } ((\cap) K) \mathfrak{G})) ! j = (\text{map } ((\cap) K) \mathfrak{G}) ! i$
 $(\text{remdups-adj } (\text{map } ((\cap) K) \mathfrak{G})) ! (j + 1) = (\text{map } ((\cap) K) \mathfrak{G}) ! (i + 1)$
using remdups-adj-obtain-adjacency by force
from i(1) have i': $i + 1 < \text{length } \mathfrak{G}$ by (metis length-map)
hence GiSi: $\mathfrak{G} ! i \triangleleft G(\text{carrier} := \mathfrak{G} ! (i + 1))$ by (metis normal)
hence GiSi': $\mathfrak{G} ! i \subseteq \mathfrak{G} ! (i + 1)$ using normal-imp-subgroup subgroup.subset
by force
from i' have finGSi: finite $(\mathfrak{G} ! (i + 1))$ using normal-series-subgroups finite
by (metis subgroup-finite)
from GiSi KG i' normal-series-subgroups have GSiKnormGSi: $\mathfrak{G} ! (i + 1) \cap K$
 $\triangleleft G(\text{carrier} := \mathfrak{G} ! (i + 1))$
using second-isomorphism-grp.normal-subgrp-intersection-normal
unfolding second-isomorphism-grp-def second-isomorphism-grp-axioms-def by
auto
with GiSi have $\mathfrak{G} ! i \cap (\mathfrak{G} ! (i + 1) \cap K) \triangleleft G(\text{carrier} := \mathfrak{G} ! (i + 1))$
by (metis group.normal-subgroup-intersect group.subgroup-imp-group i' is-group
is-normal-series normal-series.normal-series-subgroups)
hence $K \cap (\mathfrak{G} ! i \cap \mathfrak{G} ! (i + 1)) \triangleleft G(\text{carrier} := \mathfrak{G} ! (i + 1))$ by (metis
inf-commute inf-left-commute)
hence KGinormGSi: $K \cap \mathfrak{G} ! i \triangleleft G(\text{carrier} := \mathfrak{G} ! (i + 1))$ using GiSi' by
(metis le-iff-inf)
moreover have $K \cap \mathfrak{G} ! i \subseteq K \cap \mathfrak{G} ! (i + 1)$ using GiSi' by auto
moreover have groupGSi: group $(G(\text{carrier} := \mathfrak{G} ! (i + 1)))$ using i normal-series-subgroups
subgroup-imp-group by auto
moreover have subKGSiGSi: subgroup $(K \cap \mathfrak{G} ! (i + 1)) (G(\text{carrier} := \mathfrak{G} ! (i + 1)))$
by (metis GSiKnormGSi inf-sup-aci(1) normal-imp-subgroup)
ultimately have fstgoal: $K \cap \mathfrak{G} ! i \triangleleft G(\text{carrier} := \mathfrak{G} ! (i + 1), \text{carrier} := K$
 $\cap \mathfrak{G} ! (i + 1))$
using group.normal-restrict-supergroup by force
thus remdups-adj $(\text{map } ((\cap) K) \mathfrak{G}) ! j \triangleleft G(\text{carrier} := K, \text{carrier} := \text{remdups-adj}$
 $(\text{map } ((\cap) K) \mathfrak{G}) ! (j + 1))$
using i by auto
from simplefact have Gisimple: simple-group $(G(\text{carrier} := \mathfrak{G} ! (i + 1))) \text{ Mod}$
 $\mathfrak{G} ! i$ using i' by simp
hence Gimax: max-normal-subgroup $(\mathfrak{G} ! i) (G(\text{carrier} := \mathfrak{G} ! (i + 1)))$
using normal.max-normal-simple-quotient GiSi finGSi by force
from GSiKnormGSi GiSi have $\mathfrak{G} ! i \langle \# \rangle G(\text{carrier} := \mathfrak{G} ! (i + 1)) \mathfrak{G} ! (i + 1)$
 $\cap K \triangleleft (G(\text{carrier} := \mathfrak{G} ! (i + 1)))$
using groupGSi group.normal-subgroup-set-mult-closed set-mult-consistent by
fastforce
hence $\mathfrak{G} ! i \langle \# \rangle \mathfrak{G} ! (i + 1) \cap K \triangleleft G(\text{carrier} := \mathfrak{G} ! (i + 1))$ unfolding
set-mult-def by auto

hence $\mathfrak{G} ! i <\#\rangle K \cap \mathfrak{G} ! (i + 1) \triangleleft G(\text{carrier} := \mathfrak{G} ! (i + 1))$ **using**
inf-commute by metis
moreover have $\mathfrak{G} ! i \subseteq \mathfrak{G} ! i <\#\rangle G(\text{carrier} := \mathfrak{G} ! (i + 1)) K \cap \mathfrak{G} ! (i + 1)$
using *second-isomorphism-grp.H-contained-in-set-mult*
unfolding *second-isomorphism-grp-def second-isomorphism-grp-axioms-def*
using *subKGSiGSi GiSi normal-imp-subgroup* **by** *fastforce*
hence $\mathfrak{G} ! i \subseteq \mathfrak{G} ! i <\#\rangle K \cap \mathfrak{G} ! (i + 1)$ **unfolding** *set-mult-def* **by** *auto*
ultimately have *KGdisj*: $\mathfrak{G} ! i <\#\rangle K \cap \mathfrak{G} ! (i + 1) = \mathfrak{G} ! i \vee \mathfrak{G} ! i <\#\rangle K$
 $\cap \mathfrak{G} ! (i + 1) = \mathfrak{G} ! (i + 1)$
using *Gimax* **unfolding** *max-normal-subgroup-def max-normal-subgroup-axioms-def*
by *auto*
obtain φ **where** $\varphi \in \text{iso } (G(\text{carrier} := K \cap \mathfrak{G} ! (i + 1)) \text{Mod } (\mathfrak{G} ! i \cap (K \cap$
 $\mathfrak{G} ! (i + 1))))$
 $(G(\text{carrier} := \mathfrak{G} ! i <\#\rangle G(\text{carrier} := \mathfrak{G} ! (i + 1)) K \cap \mathfrak{G} ! (i + 1))$
 $\text{Mod } \mathfrak{G} ! i)$
using *second-isomorphism-grp.normal-intersection-quotient-isom*
unfolding *second-isomorphism-grp-def second-isomorphism-grp-axioms-def*
using *GiSi subKGSiGSi normal-imp-subgroup* **by** *fastforce*
hence $\varphi \in \text{iso } (G(\text{carrier} := K \cap \mathfrak{G} ! (i + 1)) \text{Mod } (K \cap \mathfrak{G} ! (i + 1) \cap \mathfrak{G} ! i))$
 $(G(\text{carrier} := \mathfrak{G} ! i <\#\rangle G(\text{carrier} := \mathfrak{G} ! (i + 1)) K \cap \mathfrak{G} ! (i +$
 $1)) \text{Mod } \mathfrak{G} ! i)$
by *(metis inf-commute)*
hence $\varphi \in \text{iso } (G(\text{carrier} := K \cap \mathfrak{G} ! (i + 1)) \text{Mod } (K \cap (\mathfrak{G} ! (i + 1) \cap \mathfrak{G} !$
 $i)))$
 $(G(\text{carrier} := \mathfrak{G} ! i <\#\rangle G(\text{carrier} := \mathfrak{G} ! (i + 1)) K \cap \mathfrak{G} ! (i + 1))$
 $\text{Mod } \mathfrak{G} ! i)$
by *(metis Int-assoc)*
hence $\varphi \in \text{iso } (G(\text{carrier} := K \cap \mathfrak{G} ! (i + 1)) \text{Mod } (K \cap \mathfrak{G} ! i))$
 $(G(\text{carrier} := \mathfrak{G} ! i <\#\rangle G(\text{carrier} := \mathfrak{G} ! (i + 1)) K \cap \mathfrak{G} ! (i + 1))$
 $\text{Mod } \mathfrak{G} ! i)$
by *(metis GiSi' Int-absorb2 Int-commute)*
hence $\varphi: \varphi \in \text{iso } (G(\text{carrier} := K \cap \mathfrak{G} ! (i + 1)) \text{Mod } (K \cap \mathfrak{G} ! i))$
 $(G(\text{carrier} := \mathfrak{G} ! i <\#\rangle K \cap \mathfrak{G} ! (i + 1)) \text{Mod } \mathfrak{G} ! i)$
unfolding *set-mult-def* **by** *auto*
from *fstgoal* **have** *KGsiKGigroup*: $\text{group } (G(\text{carrier} := K \cap \mathfrak{G} ! (i + 1)) \text{Mod}$
 $(K \cap \mathfrak{G} ! i))$ **using** *normal.factorgroup-is-group* **by** *auto*
from *KGdisj* **show** *simple-group* $(G(\text{carrier} := K, \text{carrier} := \text{remdups-adj } (\text{map}$
 $((\cap) K) \mathfrak{G} ! (j + 1)) \text{Mod } \text{remdups-adj } (\text{map } ((\cap) K) \mathfrak{G} ! j)$
proof *auto*
have *groupGi*: $\text{group } (G(\text{carrier} := \mathfrak{G} ! i))$ **using** *i' normal-series-subgroups*
subgroup-imp-group **by** *auto*
assume $\mathfrak{G} ! i <\#\rangle K \cap \mathfrak{G} ! \text{Suc } i = \mathfrak{G} ! i$
with φ **have** $\varphi \in \text{iso } (G(\text{carrier} := K \cap \mathfrak{G} ! (i + 1)) \text{Mod } (K \cap \mathfrak{G} ! i))$
 $(G(\text{carrier} := \mathfrak{G} ! i) \text{Mod } \mathfrak{G} ! i)$ **by** *auto*
moreover obtain ψ **where** $\psi \in \text{iso } (G(\text{carrier} := \mathfrak{G} ! i) \text{Mod } (\text{carrier}$
 $(G(\text{carrier} := \mathfrak{G} ! i)))) (G(\text{carrier} := \{1_{G(\text{carrier} := \mathfrak{G} ! i)}\}))$
using *group.self-factor-iso groupGi* **by** *force*
ultimately obtain π **where** $\pi \in \text{iso } (G(\text{carrier} := K \cap \mathfrak{G} ! (i + 1)) \text{Mod } (K$

$\cap \mathfrak{G} ! i)$ ($G(\text{carrier} := \{1\})$)
using *iso-set-trans* **by** *fastforce*
hence $\text{order } (G(\text{carrier} := K \cap \mathfrak{G} ! (i + 1)) \text{ Mod } (K \cap \mathfrak{G} ! i)) = \text{order}$
 $(G(\text{carrier} := \{1\}))$
by (*meson iso-same-order*)
hence $\text{order } (G(\text{carrier} := K \cap \mathfrak{G} ! (i + 1)) \text{ Mod } (K \cap \mathfrak{G} ! i)) = 1$ **unfolding**
order-def **by** *auto*
hence $\text{carrier } (G(\text{carrier} := K \cap \mathfrak{G} ! (i + 1)) \text{ Mod } (K \cap \mathfrak{G} ! i)) = \{1_{G(\text{carrier} := K \cap \mathfrak{G} ! (i + 1)) \text{ Mod } (K \cap \mathfrak{G} ! i)}$
using *group.order-one-triv-iff KGsiKGigroup* **by** *blast*
moreover from *fstgoal* **have** $K \cap \mathfrak{G} ! i \triangleleft G(\text{carrier} := K \cap \mathfrak{G} ! (i + 1))$ **by**
auto
moreover from *finGSi* **have** *finite* ($\text{carrier } (G(\text{carrier} := K \cap \mathfrak{G} ! (i + 1)))$)
by *auto*
ultimately have $K \cap \mathfrak{G} ! i = \text{carrier } (G(\text{carrier} := K \cap \mathfrak{G} ! (i + 1)))$ **by**
(*metis normal.fact-group-trivial-iff*)
hence ($\text{remdups-adj } (\text{map } ((\cap) K) \mathfrak{G}) ! j = (\text{remdups-adj } (\text{map } ((\cap) K) \mathfrak{G})$)
 $! (j + 1)$ **using** i **by** *auto*
with j **have** *False* **using** *remdups-adj-adjacent KGnotempty Suc-eq-plus1* **by**
metis
thus *simple-group* ($G(\text{carrier} := \text{remdups-adj } (\text{map } ((\cap) K) \mathfrak{G}) ! \text{Suc } j) \text{ Mod}$
 $\text{remdups-adj } (\text{map } ((\cap) K) \mathfrak{G}) ! j$)..
next
assume $\mathfrak{G} ! i <\#\#> K \cap \mathfrak{G} ! \text{Suc } i = \mathfrak{G} ! \text{Suc } i$
with φ **have** $\varphi \in \text{iso } (G(\text{carrier} := K \cap \mathfrak{G} ! (i + 1)) \text{ Mod } (K \cap \mathfrak{G} ! i))$
 $(G(\text{carrier} := \mathfrak{G} ! (i + 1)) \text{ Mod } \mathfrak{G} ! i)$
by *auto*
then obtain φ' **where** $\varphi' \in \text{iso } (G(\text{carrier} := \mathfrak{G} ! (i + 1)) \text{ Mod } \mathfrak{G} ! i)$
 $(G(\text{carrier} := K \cap \mathfrak{G} ! (i + 1)) \text{ Mod } (K \cap \mathfrak{G} ! i))$
using *KGsiKGigroup group.iso-set-sym* **by** *auto*
with *Gisimple KGsiKGigroup* **have** *simple-group* ($G(\text{carrier} := K \cap \mathfrak{G} ! (i +$
 $1)) \text{ Mod } (K \cap \mathfrak{G} ! i)$) **by** (*metis simple-group.iso-simple*)
with i **show** *simple-group* ($G(\text{carrier} := \text{remdups-adj } (\text{map } ((\cap) K) \mathfrak{G}) ! \text{Suc}$
 $j) \text{ Mod } \text{remdups-adj } (\text{map } ((\cap) K) \mathfrak{G}) ! j$)
by *auto*
qed
qed

lemma (*in group*) *composition-series-extend*:

assumes *composition-series* ($G(\text{carrier} := H)$) \mathfrak{H}
assumes *simple-group* ($G \text{ Mod } H$) $H \triangleleft G$
shows *composition-series* $G (\mathfrak{H} @ [\text{carrier } G])$
unfolding *composition-series-def* *composition-series-axioms-def*
proof *auto*
from *assms(1)* **interpret** *comp \mathfrak{H}* : *composition-series* $G(\text{carrier} := H) \mathfrak{H}$.
show *normal-series* $G (\mathfrak{H} @ [\text{carrier } G])$ **using** *assms(3)* *comp \mathfrak{H} .is-normal-series*
by (*metis normal-series-extend*)
fix i
assume $i: i < \text{length } \mathfrak{H}$
show *simple-group* ($G(\text{carrier} := (\mathfrak{H} @ [\text{carrier } G]) ! \text{Suc } i) \text{ Mod } (\mathfrak{H} @ [\text{carrier}$

```

G]) ! i)
proof (cases i = length  $\mathfrak{H} - 1$ )
  case True
    hence ( $\mathfrak{H} @ [carrier\ G]$ ) ! Suc i = carrier G by (metis i diff-Suc-1 lessE
nth-append-length)
    moreover have ( $\mathfrak{H} @ [carrier\ G]$ ) ! i =  $\mathfrak{H} ! i$  by (metis butlast-snoc i nth-butlast)
    hence ( $\mathfrak{H} @ [carrier\ G]$ ) ! i = H using True last-conv-nth comp $\mathfrak{H}$ .notempty
comp $\mathfrak{H}$ .last by auto
    ultimately show ?thesis using assms(2) by auto
  next
  case False
    hence Suc i < length  $\mathfrak{H}$  using i by auto
    hence ( $\mathfrak{H} @ [carrier\ G]$ ) ! Suc i =  $\mathfrak{H} ! Suc\ i$  using nth-append by metis
    moreover from i have ( $\mathfrak{H} @ [carrier\ G]$ ) ! i =  $\mathfrak{H} ! i$  using nth-append by
metis
    ultimately show ?thesis using <Suc i < length  $\mathfrak{H}$ > comp $\mathfrak{H}$ .simplefact by auto
  qed
qed

```

lemma (in composition-series) entries-mono:

```

assumes i ≤ j j < length  $\mathfrak{G}$ 
shows  $\mathfrak{G} ! i \subseteq \mathfrak{G} ! j$ 
using assms proof (induction j - i arbitrary: i j)
  case 0
    hence i = j by auto
    thus  $\mathfrak{G} ! i \subseteq \mathfrak{G} ! j$  by auto
  next
  case (Suc k i j)
    hence i': i + (Suc k) = j i + 1 < length  $\mathfrak{G}$  by auto
    hence ij: i + 1 ≤ j by auto
    have  $\mathfrak{G} ! i \subseteq \mathfrak{G} ! (i + 1)$  using i' normal normal-imp-subgroup subgroup.subset
by force
    moreover have j - (i + 1) = k j < length  $\mathfrak{G}$  using Suc assms by auto
    hence  $\mathfrak{G} ! (i + 1) \subseteq \mathfrak{G} ! j$  using Suc(1) ij by auto
    ultimately show  $\mathfrak{G} ! i \subseteq \mathfrak{G} ! j$  by simp
  qed

```

end

theory GroupIsoClasses

imports

HOL-Algebra.Coset

begin

3 Isomorphism Classes of Groups

We construct a quotient type for isomorphism classes of groups.

```

typedef 'a group = {G :: 'a monoid. group G}
proof
  show  $\bigwedge a. (\text{carrier} = \{a\}, \text{mult} = (\lambda x y. x), \text{one} = a) \in \{G. \text{group } G\}$ 
  unfolding group-def group-axioms-def monoid-def Units-def by auto
qed

definition group-iso-rel :: 'a group  $\Rightarrow$  'a group  $\Rightarrow$  bool
  where group-iso-rel G H =  $(\exists \varphi. \varphi \in \text{iso} (\text{Rep-group } G) (\text{Rep-group } H))$ 

quotient-type 'a group-iso-class = 'a group / group-iso-rel
  morphisms Rep-group-iso Abs-group-iso
proof (rule equivpI)
  show reflp group-iso-rel
  proof (rule reflpI)
    fix G :: 'b group
    show group-iso-rel G G
      unfolding group-iso-rel-def using iso-set-refl by blast
    qed
  next
  show symp group-iso-rel
  proof (rule sympI)
    fix G H :: 'b group
    assume group-iso-rel G H
    then obtain  $\varphi$  where  $\varphi \in \text{iso} (\text{Rep-group } G) (\text{Rep-group } H)$  unfolding
    group-iso-rel-def by auto
    then obtain  $\varphi'$  where  $\varphi' \in \text{iso} (\text{Rep-group } H) (\text{Rep-group } G)$  using group.iso-sym
    Rep-group
    using group.iso-set-sym by blast
    thus group-iso-rel H G unfolding group-iso-rel-def by auto
    qed
  next
  show transp group-iso-rel
  proof (rule transpI)
    fix G H I :: 'b group
    assume group-iso-rel G H group-iso-rel H I
    then obtain  $\varphi \psi$  where  $\varphi \in \text{iso} (\text{Rep-group } G) (\text{Rep-group } H)$   $\psi \in \text{iso}$ 
     $(\text{Rep-group } H) (\text{Rep-group } I)$ 
    unfolding group-iso-rel-def by auto
    then obtain  $\pi$  where  $\pi \in \text{iso} (\text{Rep-group } G) (\text{Rep-group } I)$ 
    using iso-set-trans by blast
    thus group-iso-rel G I unfolding group-iso-rel-def by auto
    qed
  qed

```

This assigns to a given group the group isomorphism class

```

definition (in group) iso-class :: 'a group-iso-class
  where iso-class = Abs-group-iso (Abs-group (monoid.truncate G))

```

Two isomorphic groups do indeed have the same isomorphism class:

```

lemma iso-classes-iff:
  assumes group G
  assumes group H
  shows  $(\exists \varphi. \varphi \in \text{iso } G H) = (\text{group.iso-class } G = \text{group.iso-class } H)$ 
proof –
  from assms(1,2) have groups:group (monoid.truncate G) group (monoid.truncate H)
  unfolding monoid.truncate-def group-def group-axioms-def Units-def monoid-def
by auto
  have  $(\exists \varphi. \varphi \in \text{iso } G H) = (\exists \varphi. \varphi \in \text{iso } (\text{monoid.truncate } G) (\text{monoid.truncate } H))$ 
  unfolding iso-def hom-def monoid.truncate-def by auto
  also have  $\dots = \text{group-iso-rel } (\text{Abs-group } (\text{monoid.truncate } G)) (\text{Abs-group } (\text{monoid.truncate } H))$ 
  unfolding group-iso-rel-def using groups group.Abs-group-inverse by (metis mem-Collect-eq)
  also have  $\dots = (\text{group.iso-class } G = \text{group.iso-class } H)$  using group.iso-class-def assms group-iso-class.abs-eq-iff by metis
  finally show ?thesis.
qed

end

```

```

theory JordanHolder
imports
  CompositionSeries
  MaximalNormalSubgroups
  HOL-Library.Multiset
  GroupIsoClasses
begin

```

4 The Jordan-Hölder Theorem

```

locale jordan-hoelder = group
  + comp $\mathfrak{H}$ ?: composition-series G  $\mathfrak{H}$ 
  + comp $\mathfrak{G}$ ?: composition-series G  $\mathfrak{G}$  for  $\mathfrak{H}$  and  $\mathfrak{G}$ 
  + assumes finite: finite (carrier G)

```

Before we finally start the actual proof of the theorem, one last lemma: Cancelling the last entry of a normal series results in a normal series with quotients being all but the last of the original ones.

```

lemma (in normal-series) quotients-butlast:
  assumes length  $\mathfrak{G} > 1$ 
  shows butlast quotients = normal-series.quotients (G(|carrier :=  $\mathfrak{G} ! (\text{length } \mathfrak{G} - 1 - 1)$ )) (take (length  $\mathfrak{G} - 1$ )  $\mathfrak{G}$ )
proof (rule nth-equalityI)
  define n where n = length  $\mathfrak{G} - 1$ 

```

hence $n = \text{length } (\text{take } n \mathfrak{G})$ $n > 0$ $n < \text{length } \mathfrak{G}$ **using** *assms notempty* **by**
auto
interpret *normal* \mathfrak{G} *butlast*: *normal-series* ($G(\text{carrier} := \mathfrak{G} ! (n - 1))$) *take* $n \mathfrak{G}$

using *normal-series-prefix-closed* $\langle n > 0 \rangle$ $\langle n < \text{length } \mathfrak{G} \rangle$ **by** *auto*
have $\text{length } (\text{butlast } \text{quotients}) = \text{length } \text{quotients} - 1$ **by** (*metis length-butlast*)
also have $\dots = \text{length } \mathfrak{G} - 1 - 1$ **by** (*metis add-diff-cancel-right'* *quotients-length*)
also have $\dots = \text{length } (\text{take } n \mathfrak{G}) - 1$ **by** (*metis* $\langle n = \text{length } (\text{take } n \mathfrak{G}) \rangle$ *n-def*)
also have $\dots = \text{length } \text{normal}\mathfrak{G}\text{butlast.quotients}$ **by** (*metis normal* \mathfrak{G} *butlast.quotients-length*
diff-add-inverse2)
finally show $\text{length } (\text{butlast } \text{quotients}) = \text{length } \text{normal}\mathfrak{G}\text{butlast.quotients}$.
have $\forall i < \text{length } (\text{butlast } \text{quotients}). \text{butlast } \text{quotients} ! i = \text{normal}\mathfrak{G}\text{butlast.quotients}$
 $! i$
proof *auto*
fix i
assume $i: i < \text{length } \text{quotients} - \text{Suc } 0$
hence $i': i < \text{length } \mathfrak{G} - 1$ $i < n$ $i + 1 < n$ **unfolding** *n-def* **using** *quo-*
tients-length **by** *auto*
from i **have** $\text{butlast } \text{quotients} ! i = \text{quotients} ! i$ **by** (*metis One-nat-def*
length-butlast nth-butlast)
also have $\dots = G(\text{carrier} := \mathfrak{G} ! (i + 1)) \text{Mod } \mathfrak{G} ! i$ **unfolding** *quotients-def*
using $i'(1)$ **by** *auto*
also have $\dots = G(\text{carrier} := (\text{take } n \mathfrak{G}) ! (i + 1)) \text{Mod } (\text{take } n \mathfrak{G}) ! i$ **using**
 $i'(2,3)$ *nth-take* **by** *metis*
also have $\dots = \text{normal}\mathfrak{G}\text{butlast.quotients} ! i$ **unfolding** *normal* \mathfrak{G} *butlast.quotients-def*
using i' **by** *fastforce*
finally show $\text{butlast } (\text{normal-series.quotients } G \mathfrak{G}) ! i = \text{normal-series.quotients}$
 $(G(\text{carrier} := \mathfrak{G} ! (n - \text{Suc } 0))) (\text{take } n \mathfrak{G}) ! i$ **by** *auto*
qed
thus $\bigwedge i. i < \text{length } (\text{butlast } \text{quotients})$
 $\implies \text{butlast } \text{quotients} ! i$
 $= \text{normal-series.quotients } (G(\text{carrier} := \mathfrak{G} ! (\text{length } \mathfrak{G} - 1 - 1)))$
 $(\text{take } (\text{length } \mathfrak{G} - 1) \mathfrak{G}) ! i$
unfolding *n-def* **by** *auto*
qed

The main part of the Jordan Hölder theorem is its statement about the uniqueness of a composition series. Here, uniqueness up to reordering and isomorphism is modelled by stating that the multisets of isomorphism classes of all quotients are equal.

theorem *jordan-hoelder-multisets*:

assumes *group* G
assumes *finite* (*carrier* G)
assumes *composition-series* $G \mathfrak{G}$
assumes *composition-series* $G \mathfrak{H}$
shows $\text{mset } (\text{map } \text{group.iso-class } (\text{normal-series.quotients } G \mathfrak{G}))$
 $= \text{mset } (\text{map } \text{group.iso-class } (\text{normal-series.quotients } G \mathfrak{H}))$
using *assms*
proof (*induction* $\text{length } \mathfrak{G}$ *arbitrary*: $\mathfrak{G} \mathfrak{H}$ G *rule*: *full-nat-induct*)

```

case (1  $\mathfrak{G}$   $\mathfrak{H}$   $G$ )
then interpret comp $\mathfrak{G}$ : composition-series  $G$   $\mathfrak{G}$  by simp
from 1 interpret comp $\mathfrak{H}$ : composition-series  $G$   $\mathfrak{H}$  by simp
from 1 interpret grp $G$ : group  $G$  by simp
show ?case
proof (cases length  $\mathfrak{G} \leq 2$ )
next
  case True
  hence length  $\mathfrak{G} = 0 \vee$  length  $\mathfrak{G} = 1 \vee$  length  $\mathfrak{G} = 2$  by arith
  with comp $\mathfrak{G}$ .notempty have length  $\mathfrak{G} = 1 \vee$  length  $\mathfrak{G} = 2$  by simp
  thus ?thesis
  proof (auto simp del: mset-map)
    — First trivial case:  $\mathfrak{G}$  is the trivial group.
    assume length  $\mathfrak{G} = \text{Suc } 0$ 
    hence length: length  $\mathfrak{G} = 1$  by simp
    hence length [] + 1 = length  $\mathfrak{G}$  by auto
    moreover from length have char $\mathfrak{G}$ :  $\mathfrak{G} = [\{1_G\}]$  by (metis comp $\mathfrak{G}$ .composition-series-length-one)
    hence carrier  $G = \{1_G\}$  by (metis comp $\mathfrak{G}$ .composition-series-triv-group)
    with length char $\mathfrak{G}$  have  $\mathfrak{G} = \mathfrak{H}$  using comp $\mathfrak{H}$ .composition-series-triv-group
  by simp
  thus ?thesis by simp
  next
    — Second trivial case:  $\mathfrak{G}$  is simple.
    assume length  $\mathfrak{G} = 2$ 
    hence  $\mathfrak{G}$ char:  $\mathfrak{G} = [\{1_G\}, \text{carrier } G]$  by (metis comp $\mathfrak{G}$ .length-two-unique)
    hence simple: simple-group  $G$  by (metis comp $\mathfrak{G}$ .composition-series-simple-group)
    hence  $\mathfrak{H} = [\{1_G\}, \text{carrier } G]$  using comp $\mathfrak{H}$ .composition-series-simple-group
  by auto
  with  $\mathfrak{G}$ char have  $\mathfrak{G} = \mathfrak{H}$  by simp
  thus ?thesis by simp
  qed
next
  case False
  — Non-trivial case:  $\mathfrak{G}$  has length at least 3.
  hence length: length  $\mathfrak{G} \geq 3$  by simp
  — First we show that  $\mathfrak{H}$  must have a length of at least 3.
  hence  $\neg$  simple-group  $G$  using comp $\mathfrak{G}$ .composition-series-simple-group by auto
  hence  $\mathfrak{H} \neq [\{1_G\}, \text{carrier } G]$  using comp $\mathfrak{H}$ .composition-series-simple-group by
  auto
  hence length  $\mathfrak{H} \neq 2$  using comp $\mathfrak{H}$ .length-two-unique by auto
  moreover from length have carrier  $G \neq \{1_G\}$  using comp $\mathfrak{G}$ .composition-series-length-one
  comp $\mathfrak{G}$ .composition-series-triv-group by auto
  hence length  $\mathfrak{H} \neq 1$  using comp $\mathfrak{H}$ .composition-series-length-one comp $\mathfrak{H}$ .composition-series-triv-group
  by auto
  moreover from comp $\mathfrak{H}$ .notempty have length  $\mathfrak{H} \neq 0$  by simp
  ultimately have length $\mathfrak{H}$ big: length  $\mathfrak{H} \geq 3$  using comp $\mathfrak{H}$ .notempty by arith
  define  $m$  where  $m = \text{length } \mathfrak{H} - 1$ 
  define  $n$  where  $n = \text{length } \mathfrak{G} - 1$ 
  from length $\mathfrak{H}$ big have  $m'$ :  $m > 0$   $m < \text{length } \mathfrak{H}$   $(m - 1) + 1 < \text{length } \mathfrak{H}$   $m$ 

```

$- 1 = \text{length } \mathfrak{H} - 2 m - 1 + 1 = \text{length } \mathfrak{H} - 1 m - 1 < \text{length } \mathfrak{H}$
unfolding m -def by auto
from length **have** n' : $n > 0$ $n < \text{length } \mathfrak{G} (n - 1) + 1 < \text{length } \mathfrak{G} n - 1 < \text{length } \mathfrak{G} \text{ Suc } n \leq \text{length } \mathfrak{G}$
 $n - 1 = \text{length } \mathfrak{G} - 2 n - 1 + 1 = \text{length } \mathfrak{G} - 1$ **unfolding** n -def by auto
define $\mathfrak{G}Pn$ **where** $\mathfrak{G}Pn = G(\text{carrier} := \mathfrak{G} ! (n - 1))$
define $\mathfrak{H}Pm$ **where** $\mathfrak{H}Pm = G(\text{carrier} := \mathfrak{H} ! (m - 1))$
then interpret $\text{grp}\mathfrak{G}Pn$: *group* $\mathfrak{G}Pn$ **unfolding** $\mathfrak{G}Pn$ -def **using** n' by (*metis compG.normal-series-subgroups compG.subgroup-imp-group*)
interpret $\text{grp}\mathfrak{H}Pm$: *group* $\mathfrak{H}Pm$ **unfolding** $\mathfrak{H}Pm$ -def **using** m' *compH.normal-series-subgroups 1(2) group.subgroup-imp-group* by force
have finGbl : *finite* (*carrier* $\mathfrak{G}Pn$) **using** $\langle n - 1 < \text{length } \mathfrak{G} \rangle$ $1(3)$ **unfolding** $\mathfrak{G}Pn$ -def **using** *compG.normal-series-subgroups compG.subgroup-finite* by auto
have finHbl : *finite* (*carrier* $\mathfrak{H}Pm$) **using** $\langle m - 1 < \text{length } \mathfrak{H} \rangle$ $1(3)$ **unfolding** $\mathfrak{H}Pm$ -def **using** *compH.normal-series-subgroups compG.subgroup-finite* by auto
have $\text{quots}\mathfrak{G}\text{notempty}$: *compG.quotients* $\neq []$ **using** *compG.quotients-length length* by auto
have $\text{quots}\mathfrak{H}\text{notempty}$: *compH.quotients* $\neq []$ **using** *compH.quotients-length lengthHbig* by auto

— Instantiate truncated composition series since they are used for both cases
interpret $\mathfrak{H}\text{butlast}$: *composition-series* $\mathfrak{H}Pm$ take m \mathfrak{H} **using** *compH.composition-series-prefix-closed m'(1,2) Hpm-def* by auto
interpret $\mathfrak{G}\text{butlast}$: *composition-series* $\mathfrak{G}Pn$ take n \mathfrak{G} **using** *compG.composition-series-prefix-closed n'(1,2) GPn-def* by auto
have ltaken : $n = \text{length} (\text{take } n \mathfrak{G})$ **using** *length-take n'(2)* by auto
have ltakem : $m = \text{length} (\text{take } m \mathfrak{H})$ **using** *length-take m'(2)* by auto
show *?thesis*
proof (*cases* $\mathfrak{H} ! (m - 1) = \mathfrak{G} ! (n - 1)$)
— If $\mathfrak{H} ! (l - 1) = \mathfrak{G} ! 1$, everything is simple...
case *True*
— The last quotients of \mathfrak{G} and \mathfrak{H} are equal.
have lasteq : *last compG.quotients = last compH.quotients*
proof –
from length **have** lg : $\text{length } \mathfrak{G} - 1 - 1 + 1 = \text{length } \mathfrak{G} - 1$ by (*metis Suc-diff-1 Suc-eq-plus1 n'(1) n-def*)
from $\text{length}\mathfrak{H}\text{big}$ **have** lh : $\text{length } \mathfrak{H} - 1 - 1 + 1 = \text{length } \mathfrak{H} - 1$ by (*metis Suc-diff-1 Suc-eq-plus1 <0 < m> m-def*)
have *last compG.quotients = G Mod G ! (n - 1)* **using** *length compG.last-quotient*
unfolding n -def by auto
also have $\dots = G \text{ Mod } \mathfrak{H} ! (m - 1)$ **using** *True* by *simp*
also have $\dots = \text{last compH.quotients}$ **using** *lengthHbig compH.last-quotient*
unfolding m -def by auto
finally show *?thesis* .
qed
from ltaken **have** ind : *mset (map group.iso-class Gbutlast.quotients) = mset (map group.iso-class Hbutlast.quotients)*
using $1(1)$ *True n'(5) grpGPn.is-group finGbl Gbutlast.is-composition-series Hbutlast.is-composition-series* **unfolding** $\mathfrak{G}Pn$ -def $\mathfrak{H}Pm$ -def by *metis*

```

    have mset (map group.iso-class comp $\mathfrak{G}$ .quotients)
      = mset (map group.iso-class (butlast comp $\mathfrak{G}$ .quotients @ [last
comp $\mathfrak{G}$ .quotients])) by (simp add: quot $\mathfrak{G}$ notempty)
    also have ... = mset (map group.iso-class ( $\mathfrak{G}$ butlast.quotients @ [last (comp $\mathfrak{G}$ .quotients)]))
using comp $\mathfrak{G}$ .quotients-butlast length unfolding n-def  $\mathfrak{G}$ Pn-def by auto
    also have ... = mset ((map group.iso-class  $\mathfrak{G}$ butlast.quotients) @ [group.iso-class
(last (comp $\mathfrak{G}$ .quotients))]) by auto
    also have ... = mset (map group.iso-class  $\mathfrak{G}$ butlast.quotients) + {# group.iso-class
(last (comp $\mathfrak{G}$ .quotients)) #} by auto
    also have ... = mset (map group.iso-class  $\mathfrak{H}$ butlast.quotients) + {# group.iso-class
(last (comp $\mathfrak{G}$ .quotients)) #} using ind by simp
    also have ... = mset (map group.iso-class  $\mathfrak{H}$ butlast.quotients) + {# group.iso-class
(last (comp $\mathfrak{H}$ .quotients)) #} using lasteq by simp
    also have ... = mset ((map group.iso-class  $\mathfrak{H}$ butlast.quotients) @ [group.iso-class
(last (comp $\mathfrak{H}$ .quotients))]) by auto
    also have ... = mset (map group.iso-class ( $\mathfrak{H}$ butlast.quotients @ [last (comp $\mathfrak{H}$ .quotients)]))
by auto
    also have ... = mset (map group.iso-class (butlast comp $\mathfrak{H}$ .quotients @ [last
comp $\mathfrak{H}$ .quotients])) using length $\mathfrak{H}$ big comp $\mathfrak{H}$ .quotients-butlast unfolding m-def  $\mathfrak{H}$ Pm-def
by auto
    also have ... = mset (map group.iso-class comp $\mathfrak{H}$ .quotients) using ap-
pend-butlast-last-id quot $\mathfrak{H}$ notempty by simp
    finally show ?thesis .
next
case False
define  $\mathfrak{H}$ PmInt $\mathfrak{G}$ Pn where  $\mathfrak{H}$ PmInt $\mathfrak{G}$ Pn = G(|carrier :=  $\mathfrak{H}$  ! (m - 1)  $\cap$   $\mathfrak{G}$ 
! (n - 1)|)
interpret  $\mathfrak{G}$ Pnmax: max-normal-subgroup  $\mathfrak{G}$  ! (n - 1) G unfolding n-def
by (metis add-lessD1 diff-diff-add n'(3) add.commute one-add-one 1(3)
comp $\mathfrak{G}$ .snd-to-last-max-normal)
interpret  $\mathfrak{H}$ Pmmax: max-normal-subgroup  $\mathfrak{H}$  ! (m - 1) G unfolding m-def
by (metis add-lessD1 diff-diff-add m'(3) add.commute one-add-one 1(3)
comp $\mathfrak{H}$ .snd-to-last-max-normal)
have  $\mathfrak{H}$ PmnormG:  $\mathfrak{H}$  ! (m - 1)  $\triangleleft$  G using comp $\mathfrak{H}$ .normal-series-snd-to-last
m'(4) unfolding m-def by auto
have  $\mathfrak{G}$ PnnormG:  $\mathfrak{G}$  ! (n - 1)  $\triangleleft$  G using comp $\mathfrak{G}$ .normal-series-snd-to-last
n'(6) unfolding n-def by auto
have  $\mathfrak{H}$ Pmint $\mathfrak{G}$ PnnormG:  $\mathfrak{H}$  ! (m - 1)  $\cap$   $\mathfrak{G}$  ! (n - 1)  $\triangleleft$  G using  $\mathfrak{H}$ PmnormG
 $\mathfrak{G}$ PnnormG by (rule comp $\mathfrak{G}$ .normal-subgroup-intersect)
have Intnorm $\mathfrak{G}$ Pn:  $\mathfrak{H}$  ! (m - 1)  $\cap$   $\mathfrak{G}$  ! (n - 1)  $\triangleleft$   $\mathfrak{G}$ Pn using  $\mathfrak{G}$ PnnormG
 $\mathfrak{H}$ PmnormG Int-lower2 unfolding  $\mathfrak{G}$ Pn-def
by (metis comp $\mathfrak{G}$ .normal-restrict-supergroup comp $\mathfrak{G}$ .normal-series-subgroups
comp $\mathfrak{G}$ .normal-subgroup-intersect n'(4))
then interpret grp $\mathfrak{G}$ PnMod $\mathfrak{H}$ Pmint $\mathfrak{G}$ Pn: group  $\mathfrak{G}$ Pn Mod  $\mathfrak{H}$  ! (m - 1)  $\cap$   $\mathfrak{G}$ 
! (n - 1) by (rule normal.factorgroup-is-group)
have Intnorm $\mathfrak{H}$ Pm:  $\mathfrak{H}$  ! (m - 1)  $\cap$   $\mathfrak{G}$  ! (n - 1)  $\triangleleft$   $\mathfrak{H}$ Pm using  $\mathfrak{H}$ PmnormG
 $\mathfrak{G}$ PnnormG Int-lower2 Int-commute unfolding  $\mathfrak{H}$ Pm-def
by (metis comp $\mathfrak{G}$ .normal-restrict-supergroup comp $\mathfrak{G}$ .normal-subgroup-intersect
comp $\mathfrak{H}$ .normal-series-subgroups m'(6))

```

then interpret $grp\mathfrak{H}PmMod\mathfrak{H}Pmint\mathfrak{G}Pn$: $group\ \mathfrak{H}Pm\ Mod\ \mathfrak{H}!(m-1) \cap \mathfrak{G}!(n-1)$ **by** (*rule normal.factorgroup-is-group*)

— Show that the second to last entries are not contained in each other.

have $not\mathfrak{H}PmSub\mathfrak{G}Pn$: $\neg(\mathfrak{H}!(m-1) \subseteq \mathfrak{G}!(n-1))$ **using** $\mathfrak{H}Pm$ -*max.max-normal* $\mathfrak{G}Pn$ *normG False[symmetric]* $\mathfrak{G}Pn$ *max.proper* **by** *simp*

have $not\mathfrak{G}PnSub\mathfrak{H}Pm$: $\neg(\mathfrak{G}!(n-1) \subseteq \mathfrak{H}!(m-1))$ **using** $\mathfrak{G}Pn$ -*max.max-normal* $\mathfrak{H}Pm$ *normG False* $\mathfrak{H}Pm$ *max.proper* **by** *simp*

— Show that $G Mod\ \mathfrak{H}!(m-1) \cap \mathfrak{G}!(n-1)$ is a simple group.

have $\mathfrak{H}PmSubSetmult$: $\mathfrak{H}!(m-1) \subseteq \mathfrak{H}!(m-1) \langle \# \rangle_G \mathfrak{G}!(n-1)$

using $\mathfrak{G}Pn$ *max.subgroup-axioms* $\mathfrak{H}Pm$ *normG second-isomorphism-grp.H-contained-in-set-mult*

second-isomorphism-grp-axioms-def second-isomorphism-grp-def **by** *blast*

have $\mathfrak{G}PnSubSetmult$: $\mathfrak{G}!(n-1) \subseteq \mathfrak{H}!(m-1) \langle \# \rangle_G \mathfrak{G}!(n-1)$

by (*metis* $\mathfrak{G}Pn$ *max.subset* $\mathfrak{G}Pn$ *normG* $\mathfrak{H}Pm$ *SubSetmult* $\mathfrak{H}Pm$ *max.max-normal* $\mathfrak{H}Pm$ *max.subgroup-axioms* $\mathfrak{H}Pm$ *normG*

comp \mathfrak{G} *.normal-subgroup-set-mult-closed* *comp* \mathfrak{G} *.set-mult-inclusion*)

have $\mathfrak{G}!(n-1) \neq (\mathfrak{H}!(m-1) \langle \# \rangle_G \mathfrak{G}!(n-1))$ **using** $\mathfrak{H}Pm$ *SubSetmult* *not* $\mathfrak{H}Pm$ *Sub* $\mathfrak{G}Pn$ **by** *auto*

hence *set-multG*: $(\mathfrak{H}!(m-1) \langle \# \rangle_G \mathfrak{G}!(n-1)) = carrier\ G$

by (*metis* $\mathfrak{G}Pn$ *SubSetmult* $\mathfrak{G}Pn$ *max.max-normal* $\mathfrak{G}Pn$ *normG* $\mathfrak{H}Pm$ *normG* *comp* \mathfrak{G} *.normal-subgroup-set-mult-closed*)

then obtain φ **where** $\varphi \in iso(\mathfrak{G}Pn\ Mod\ (\mathfrak{H}!(m-1) \cap \mathfrak{G}!(n-1)))$
($G(\backslash carrier := carrier\ G) Mod\ \mathfrak{H}!(m-1)$)

by (*metis* *second-isomorphism-grp.normal-intersection-quotient-isom* $\mathfrak{H}Pm$ -*normG*

$\mathfrak{G}Pn$ -*def* $\mathfrak{G}Pn$ *max.subgroup-axioms second-isomorphism-grp-axioms-def second-isomorphism-grp-def*)

hence φ : $\varphi \in iso(\mathfrak{G}Pn\ Mod\ (\mathfrak{H}!(m-1) \cap \mathfrak{G}!(n-1)))$ ($G Mod\ \mathfrak{H}!(m-1)$) **by** *auto*

then obtain $\varphi 2$ **where** $\varphi 2$: $\varphi 2 \in iso(G Mod\ \mathfrak{H}!(m-1))$ ($\mathfrak{G}Pn Mod\ (\mathfrak{H}!(m-1) \cap \mathfrak{G}!(n-1))$)

using *group.iso-set-sym* *grp* $\mathfrak{G}PnMod\mathfrak{H}Pmint\mathfrak{G}Pn$ *.is-group* **by** *auto*

moreover **have** *simple-group* ($G(\backslash carrier := \mathfrak{H}!(m-1+1)) Mod\ \mathfrak{H}!(m-1)$) **using** *comp* \mathfrak{H} *.simplefact* $m'(3)$ **by** *simp*

hence *simple-group* ($G Mod\ \mathfrak{H}!(m-1)$) **using** *comp* \mathfrak{H} *.last* *last-conv-nth* *comp* \mathfrak{H} *.notempty* $m'(5)$ **by** *fastforce*

ultimately **have** *simple* $\mathfrak{G}PnModInt$: *simple-group* ($\mathfrak{G}Pn Mod\ (\mathfrak{H}!(m-1) \cap \mathfrak{G}!(n-1))$)

using *simple-group.iso-simple* *grp* $\mathfrak{G}PnMod\mathfrak{H}Pmint\mathfrak{G}Pn$ *.is-group* **by** *auto*

interpret $grpGMod\mathfrak{H}Pm$: $group(G Mod\ \mathfrak{H}!(m-1))$ **by** (*metis* $\mathfrak{H}Pm$ *normG* *normal.factorgroup-is-group*)

— Show analogues of the previous statements for $\mathfrak{H}!(m-1)$ instead of $\mathfrak{G}!(n-1)$.

have $\mathfrak{H}PmSubSetmult'$: $\mathfrak{H}!(m-1) \subseteq \mathfrak{G}!(n-1) \langle \# \rangle_G \mathfrak{H}!(m-1)$

by (*metis* $\mathfrak{G}Pn$ *normG* $\mathfrak{H}Pm$ *SubSetmult* *comp* \mathfrak{G} *.commut-normal* $\mathfrak{H}Pm$ *normG* *normal-imp-subgroup*)

have $\mathfrak{G}PnSubSetmult'$: $\mathfrak{G}!(n-1) \subseteq \mathfrak{G}!(n-1) \langle \# \rangle_G \mathfrak{H}!(m-1)$
by (*metis* $\mathfrak{H}PmnormG$ *normal-imp-subgroup* $\mathfrak{G}PnSubSetmult$ $\mathfrak{G}PnormG$ *comp* \mathfrak{G} .*commut-normal*)
have $\mathfrak{H}!(m-1) \neq (\mathfrak{G}!(n-1)) \langle \# \rangle_G (\mathfrak{H}!(m-1))$ **using** $\mathfrak{G}PnSubSetmult'$ *not* $\mathfrak{G}PnSub\mathfrak{H}Pm$ **by** *auto*
hence *set-multG*: $(\mathfrak{G}!(n-1)) \langle \# \rangle_G (\mathfrak{H}!(m-1)) = \text{carrier } G$
using $\mathfrak{H}Pmmax.max-normal$ $\mathfrak{G}PnormG$ *comp* \mathfrak{G} .*normal-subgroup-set-mult-closed* $\mathfrak{H}PmSubSetmult'$ $\mathfrak{H}PmnormG$ **by** *blast*
from *set-multG* **obtain** ψ **where**
 $\psi \in \text{iso } (\mathfrak{H}Pm \text{ Mod } (\mathfrak{G}!(n-1) \cap \mathfrak{H}!(m-1))) (G(\text{carrier} := \text{carrier } G) \text{ Mod } \mathfrak{G}!(n-1))$
by (*metis* $\mathfrak{H}Pm-def$ $\mathfrak{H}PmnormG$ *second-isomorphism-grp-axioms-def* *second-isomorphism-grp-def* *second-isomorphism-grp.normal-intersection-quotient-isom* $\mathfrak{G}PnormG$ *normal-imp-subgroup*)
hence ψ : $\psi \in \text{iso } (\mathfrak{H}Pm \text{ Mod } (\mathfrak{H}!(m-1) \cap (\mathfrak{G}!(n-1)))) (G(\text{carrier} := \text{carrier } G) \text{ Mod } \mathfrak{G}!(n-1))$ **using** *Int-commute* **by** *metis*
then obtain ψ_2 **where**
 ψ_2 : $\psi_2 \in \text{iso } (G \text{ Mod } \mathfrak{G}!(n-1)) (\mathfrak{H}Pm \text{ Mod } (\mathfrak{H}!(m-1) \cap \mathfrak{G}!(n-1)))$
using *group.iso-set-sym* *grp* $\mathfrak{H}Pm\text{Mod}\mathfrak{H}Pmint\mathfrak{G}Pn.is-group$ **by** *auto*
moreover have *simple-group* $(G(\text{carrier} := \mathfrak{G}!(n-1+1)) \text{ Mod } \mathfrak{G}!(n-1))$ **using** *comp* \mathfrak{G} .*simplefact* $n'(3)$ **by** *simp*
hence *simple-group* $(G \text{ Mod } \mathfrak{G}!(n-1))$ **using** *comp* \mathfrak{G} .*last* *last-conv-nth* *comp* \mathfrak{G} .*notempty* $n'(7)$ **by** *fastforce*
ultimately have *simple* $\mathfrak{H}Pm\text{ModInt}$: *simple-group* $(\mathfrak{H}Pm \text{ Mod } (\mathfrak{H}!(m-1) \cap \mathfrak{G}!(n-1)))$
using *simple-group.iso-simple* *grp* $\mathfrak{H}Pm\text{Mod}\mathfrak{H}Pmint\mathfrak{G}Pn.is-group$ **by** *auto*
interpret *grpGMod* $\mathfrak{G}Pn$: *group* $(G \text{ Mod } \mathfrak{G}!(n-1))$ **by** (*metis* $\mathfrak{G}PnormG$ *normal.factorgroup-is-group*)

— Instantiate several composition series used to build up the equality of quotient multisets.

define \mathfrak{K} **where** $\mathfrak{K} = \text{remdups-adj } (\text{map } ((\cap) (\mathfrak{H}!(m-1))) \mathfrak{G})$
define \mathfrak{L} **where** $\mathfrak{L} = \text{remdups-adj } (\text{map } ((\cap) (\mathfrak{G}!(n-1))) \mathfrak{H})$
interpret \mathfrak{K} : *composition-series* $\mathfrak{H}Pm$ \mathfrak{K} **using** *comp* \mathfrak{G} .*intersect-normal* $1(3)$ $\mathfrak{H}PmnormG$ **unfolding** $\mathfrak{K}-def$ $\mathfrak{H}Pm-def$ **by** *auto*
interpret \mathfrak{L} : *composition-series* $\mathfrak{G}Pn$ \mathfrak{L} **using** *comp* \mathfrak{H} .*intersect-normal* $1(3)$ $\mathfrak{G}PnormG$ **unfolding** $\mathfrak{L}-def$ $\mathfrak{G}Pn-def$ **by** *auto*

— Apply the induction hypothesis on $\mathfrak{G}butlast$ and \mathfrak{L}

from $n'(2)$ **have** *Suc* $(\text{length } (\text{take } n \mathfrak{G})) \leq \text{length } \mathfrak{G}$ **by** *auto*
hence *msets* $\mathfrak{G}butlast\mathfrak{L}$: *mset* $(\text{map } \text{group.iso-class } \mathfrak{G}butlast.\text{quotients}) = \text{mset } (\text{map } \text{group.iso-class } \mathfrak{L}.\text{quotients})$
using $1.hyps$ *grp* $\mathfrak{G}Pn.is-group$ *finGbl* $\mathfrak{G}butlast.is-composition-series$ $\mathfrak{L}.is-composition-series$ **by** *metis*
hence *length* \mathfrak{L} : $n = \text{length } \mathfrak{L}$ **using** $\mathfrak{G}butlast.quotients-length$ $\mathfrak{L}.quotients-length$

length-map size-mset ltaken by metis
hence $\text{length } \mathcal{L}'$: $\text{length } \mathcal{L} > 1$ $\text{length } \mathcal{L} - 1 > 0$ $\text{length } \mathcal{L} - 1 \leq \text{length } \mathcal{L}$
using $n'(6)$ *length by auto*
have $\text{Inteq}\mathcal{L}\text{sndlast}$: $\mathfrak{H}! (m - 1) \cap \mathfrak{G}! (n - 1) = \mathcal{L}! (\text{length } \mathcal{L} - 1 - 1)$
proof –
have $\text{length } \mathcal{L} - 1 - 1 + 1 < \text{length } \mathcal{L}$ **using** $\text{length}\mathcal{L}'$ **by auto**
moreover **have** KGnotempty : $(\text{map } ((\cap) (\mathfrak{G}! (n - 1))) \mathfrak{H}) \neq \square$ **using**
comp \mathfrak{H} .*notempty* **by** (*metis Nil-is-map-conv*)
ultimately obtain i **where** i : $i + 1 < \text{length } (\text{map } ((\cap) (\mathfrak{G}! (n - 1))) \mathfrak{H})$
 $\mathcal{L}! (\text{length } \mathcal{L} - 1 - 1) = (\text{map } ((\cap) (\mathfrak{G}! (n - 1))) \mathfrak{H})! i$ $\mathcal{L}! (\text{length } \mathcal{L} - 1 - 1 + 1) = (\text{map } ((\cap) (\mathfrak{G}! (n - 1))) \mathfrak{H})! (i + 1)$
using *remdups-adj-obtain-adjacency* **unfolding** \mathcal{L} -*def* **by force**
hence $\mathcal{L}! (\text{length } \mathcal{L} - 1 - 1) = \mathfrak{H}! i \cap \mathfrak{G}! (n - 1)$ $\mathcal{L}! (\text{length } \mathcal{L} - 1 - 1 + 1) = \mathfrak{H}! (i + 1) \cap \mathfrak{G}! (n - 1)$ **by auto**
hence $\mathcal{L}! (\text{length } \mathcal{L} - 1) = \mathfrak{H}! (i + 1) \cap \mathfrak{G}! (n - 1)$ **using** $\text{length}\mathcal{L}'(2)$
by (*metis Suc-diff-1 Suc-eq-plus1*)
hence $\mathfrak{G}Pn\text{sub}\mathfrak{H}Pm$: $\mathfrak{G}! (n - 1) \subseteq \mathfrak{H}! (i + 1)$ **using** \mathcal{L} .*last* \mathcal{L} .*notempty*
last-conv-nth **unfolding** $\mathfrak{G}Pn$ -*def* **by auto**
from $i(1)$ **have** $i + 1 < m + 1$ **unfolding** m -*def* **by auto**
moreover **have** $\neg (i + 1 \leq m - 1)$ **using** *comp* \mathfrak{H} .*entries-mono* $m'(6)$
not $\mathfrak{G}Pn\text{Sub}\mathfrak{H}Pm$ $\mathfrak{G}Pn\text{sub}\mathfrak{H}Pm$ **by fastforce**
ultimately **have** $m - 1 = i$ **by auto**
with i **show** *?thesis* **by auto**
qed
hence $\mathcal{L}\text{sndlast}$: $\mathfrak{H}Pm\text{Int}\mathfrak{G}Pn = (\mathfrak{G}Pn(\text{carrier} := \mathcal{L}! (\text{length } \mathcal{L} - 1 - 1)))$
unfolding $\mathfrak{H}Pm\text{Int}\mathfrak{G}Pn$ -*def* $\mathfrak{G}Pn$ -*def* **by auto**
then interpret $\mathcal{L}\text{butlast}$: *composition-series* $\mathfrak{H}Pm\text{Int}\mathfrak{G}Pn$ *take* $(\text{length } \mathcal{L} - 1)$ \mathcal{L} **using** $\text{length}\mathcal{L}'$ \mathcal{L} .*composition-series-prefix-closed* **by metis**
from $\langle \text{length } \mathcal{L} > 1 \rangle$ **have** $\text{quots}\mathcal{L}\text{notempty}$: \mathcal{L} .*quotients* $\neq \square$ **unfolding**
 \mathcal{L} .*quotients-def* **by auto**

– Apply the induction hypothesis on $\mathcal{L}\text{butlast}$ and $\mathfrak{K}\text{butlast}$

have $\text{length } \mathfrak{K} > 1$
proof (*rule ccontr*)
assume $\neg \text{length } \mathfrak{K} > 1$
with \mathfrak{K} .*notempty* **have** $\text{length } \mathfrak{K} = 1$ **by** (*metis One-nat-def Suc-lessI length-greater-0-conv*)
hence *carrier* $\mathfrak{H}Pm = \{1_{\mathfrak{H}Pm}\}$ **using** \mathfrak{K} .*composition-series-length-one*
 \mathfrak{K} .*composition-series-triv-group* **by auto**
hence *carrier* $\mathfrak{H}Pm = \{1_G\}$ **unfolding** $\mathfrak{H}Pm$ -*def* **by auto**
hence *carrier* $\mathfrak{H}Pm \subseteq \mathfrak{G}! (n - 1)$ **using** $\mathfrak{G}Pn\text{max}$.*is-subgroup subgroup.one-closed* **by auto**
with *not* $\mathfrak{H}Pm\text{Sub}\mathfrak{G}Pn$ **show** *False* **unfolding** $\mathfrak{H}Pm$ -*def* **by auto**
qed
hence $\text{length}\mathfrak{K}'$: $\text{length } \mathfrak{K} - 1 > 0$ $\text{length } \mathfrak{K} - 1 \leq \text{length } \mathfrak{K}$ **by auto**
have $\text{Inteq}\mathfrak{K}\text{sndlast}$: $\mathfrak{H}! (m - 1) \cap \mathfrak{G}! (n - 1) = \mathfrak{K}! (\text{length } \mathfrak{K} - 1 - 1)$
proof –
have $\text{length } \mathfrak{K} - 1 - 1 + 1 < \text{length } \mathfrak{K}$ **using** $\text{length}\mathfrak{K}'$ **by auto**
moreover **have** KGnotempty : $(\text{map } ((\cap) (\mathfrak{H}! (m - 1))) \mathfrak{G}) \neq \square$ **using**

comp \mathfrak{G} .*notempty* **by** (*metis Nil-is-map-conv*)
ultimately obtain i **where** $i: i + 1 < \text{length} (\text{map} ((\cap) (\mathfrak{H} ! (m - 1))))$
 \mathfrak{G})
 $\mathfrak{K} ! (\text{length} \mathfrak{K} - 1 - 1) = (\text{map} ((\cap) (\mathfrak{H} ! (m - 1)))) \mathfrak{G} ! i \mathfrak{K} ! (\text{length} \mathfrak{K} - 1 - 1 + 1) = (\text{map} ((\cap) (\mathfrak{H} ! (m - 1)))) \mathfrak{G} ! (i + 1)$
using *remdups-adj-obtain-adjacency* **unfolding** \mathfrak{K} -*def* **by** *force*
hence $\mathfrak{K} ! (\text{length} \mathfrak{K} - 1 - 1) = \mathfrak{G} ! i \cap \mathfrak{H} ! (m - 1) \mathfrak{K} ! (\text{length} \mathfrak{K} - 1 - 1 + 1) = \mathfrak{G} ! (i + 1) \cap \mathfrak{H} ! (m - 1)$ **by** *auto*
hence $\mathfrak{K} ! (\text{length} \mathfrak{K} - 1) = \mathfrak{G} ! (i + 1) \cap \mathfrak{H} ! (m - 1)$ **using** *length $\mathfrak{K}'(1)$*
by (*metis Suc-diff-1 Suc-eq-plus1*)
hence $\mathfrak{H}Pm\text{sub}\mathfrak{G}Pn: \mathfrak{H} ! (m - 1) \subseteq \mathfrak{G} ! (i + 1)$ **using** \mathfrak{K} .*last* \mathfrak{K} .*notempty*
last-conv-nth **unfolding** $\mathfrak{H}Pm$ -*def* **by** *auto*
from $i(1)$ **have** $i + 1 < n + 1$ **unfolding** n -*def* **by** *auto*
moreover **have** $\neg (i + 1 \leq n - 1)$ **using** *comp* \mathfrak{G} .*entries-mono* $n'(2)$
not $\mathfrak{H}Pm\text{Sub}\mathfrak{G}Pn$ $\mathfrak{H}Pm\text{sub}\mathfrak{G}Pn$ **by** *fastforce*
ultimately **have** $n - 1 = i$ **by** *auto*
with i **show** *?thesis* **by** *auto*
qed
have *composition-series* ($G(\text{carrier} := \mathfrak{K} ! (\text{length} \mathfrak{K} - 1 - 1))$) (*take* ($\text{length} \mathfrak{K} - 1$) \mathfrak{K})
using *length \mathfrak{K}'* \mathfrak{K} .*composition-series-prefix-closed* **unfolding** $\mathfrak{H}Pm\text{Int}\mathfrak{G}Pn$ -*def*
 $\mathfrak{H}Pm$ -*def* **by** *fastforce*
then **interpret** $\mathfrak{R}\text{butlast}: \text{composition-series } \mathfrak{H}Pm\text{Int}\mathfrak{G}Pn$ (*take* ($\text{length} \mathfrak{K} - 1$) \mathfrak{K}) **using** *Inteq \mathfrak{R} sndlast* **unfolding** $\mathfrak{H}Pm\text{Int}\mathfrak{G}Pn$ -*def* **by** *auto*
from *finGbl* **have** *finInt*: *finite* (*carrier* $\mathfrak{H}Pm\text{Int}\mathfrak{G}Pn$) **unfolding** $\mathfrak{H}Pm\text{Int}\mathfrak{G}Pn$ -*def*
 $\mathfrak{G}Pn$ -*def* **by** *simp*
moreover **have** *Suc* ($\text{length} (\text{take} (\text{length} \mathfrak{L} - 1) \mathfrak{L})$) $\leq \text{length } \mathfrak{G}$ **using**
length \mathfrak{L} **unfolding** n -*def* **using** $n'(2)$ **by** *auto*
ultimately **have** *multisets $\mathfrak{R}\mathfrak{L}$ butlast*: *mset* (*map group.iso-class* $\mathfrak{L}\text{butlast.quotients}$)
 $= \text{mset} (\text{map group.iso-class } \mathfrak{R}\text{butlast.quotients})$
using *1.hyps* $\mathfrak{L}\text{butlast.is-group}$ $\mathfrak{R}\text{butlast.is-composition-series}$ $\mathfrak{L}\text{butlast.is-composition-series}$
by *auto*
hence $\text{length} (\text{take} (\text{length} \mathfrak{K} - 1) \mathfrak{K}) = \text{length} (\text{take} (\text{length} \mathfrak{L} - 1) \mathfrak{L})$
using $\mathfrak{R}\text{butlast.quotients-length}$ $\mathfrak{L}\text{butlast.quotients-length}$ *length-map size-mset*
by *metis*
hence $\text{length} (\text{take} (\text{length} \mathfrak{K} - 1) \mathfrak{K}) = n - 1$ **using** *length \mathfrak{L}* $n'(1)$ **by** *auto*
hence *length \mathfrak{K}* : $\text{length } \mathfrak{K} = n$ **by** (*metis Suc-diff-1* \mathfrak{K} .*notempty* *butlast-conv-take*
length-butlast length-greater-0-conv $n'(1)$)

— Apply the induction hypothesis on \mathfrak{K} and $\mathfrak{H}\text{butlast}$

from *Inteq \mathfrak{R} sndlast* **have** $\mathfrak{R}\text{sndlast}: \mathfrak{H}Pm\text{Int}\mathfrak{G}Pn = (\mathfrak{H}Pm(\text{carrier} := \mathfrak{K} ! (\text{length} \mathfrak{K} - 1 - 1)))$ **unfolding** $\mathfrak{H}Pm\text{Int}\mathfrak{G}Pn$ -*def* $\mathfrak{H}Pm$ -*def* \mathfrak{K} -*def* **by** *auto*
from *length \mathfrak{K}* **have** *Suc* ($\text{length } \mathfrak{K}$) $\leq \text{length } \mathfrak{G}$ **using** $n'(2)$ **by** *auto*
hence *multisets $\mathfrak{H}\text{butlast}\mathfrak{K}$* : *mset* (*map group.iso-class* $\mathfrak{H}\text{butlast.quotients}$) $=$
mset (*map group.iso-class* \mathfrak{K} .*quotients*)
using *1.hyps* *grp $\mathfrak{H}Pm$.is-group* *finHbl* $\mathfrak{H}\text{butlast.is-composition-series}$ \mathfrak{K} .*is-composition-series*
by *metis*
hence *length \mathfrak{K}* : $m = \text{length } \mathfrak{K}$ **using** $\mathfrak{H}\text{butlast.quotients-length}$ \mathfrak{K} .*quotients-length*
length-map size-mset *ltakem* **by** *metis*

hence $\text{length } \mathfrak{K} > 1 \text{ length } \mathfrak{K} - 1 > 0 \text{ length } \mathfrak{K} - 1 \leq \text{length } \mathfrak{K}$ **using** $m'(4)$
 $\text{length} \mathfrak{H}$ **big by auto**

hence $\text{quots} \mathfrak{K} \text{notempty}$: $\mathfrak{K}.\text{quotients} \neq []$ **unfolding** $\mathfrak{K}.\text{quotients-def}$ **by auto**

interpret $\mathfrak{R}\text{butlastadd} \mathfrak{G}Pn$: *composition-series* $\mathfrak{G}Pn$ (take (length $\mathfrak{K} - 1$) \mathfrak{R})
 $@ [\mathfrak{G} ! (n - 1)]$

using $\text{grp} \mathfrak{G}Pn.\text{composition-series-extend}$ $\mathfrak{R}\text{butlast}.\text{is-composition-series simple} \mathfrak{G}Pn \text{ModInt Intnorm} \mathfrak{G}Pn$

unfolding $\mathfrak{G}Pn\text{-def}$ $\mathfrak{H}Pm \text{Int} \mathfrak{G}Pn\text{-def}$ **by auto**

interpret $\mathfrak{L}\text{butlastadd} \mathfrak{H}Pm$: *composition-series* $\mathfrak{H}Pm$ (take (length $\mathfrak{L} - 1$) \mathfrak{L})
 $@ [\mathfrak{H} ! (m - 1)]$

using $\text{grp} \mathfrak{H}Pm.\text{composition-series-extend}$ $\mathfrak{L}\text{butlast}.\text{is-composition-series simple} \mathfrak{H}Pm \text{ModInt Intnorm} \mathfrak{H}Pm$

unfolding $\mathfrak{H}Pm\text{-def}$ $\mathfrak{H}Pm \text{Int} \mathfrak{G}Pn\text{-def}$ **by auto**

— Prove equality of those composition series.

have $\text{mset} (\text{map group.iso-class } \text{comp} \mathfrak{G}.\text{quotients})$

$= \text{mset} (\text{map group.iso-class} ((\text{butlast } \text{comp} \mathfrak{G}.\text{quotients}) @ [\text{last } \text{comp} \mathfrak{G}.\text{quotients}])))$ **using** $\text{quots} \mathfrak{G} \text{notempty}$ **by simp**

also have $\dots = \text{mset} (\text{map group.iso-class} (\mathfrak{G}\text{butlast}.\text{quotients} @ [G \text{Mod } \mathfrak{G} ! (n - 1)]))$

using $\text{comp} \mathfrak{G}.\text{quotients-butlast}$ $\text{comp} \mathfrak{G}.\text{last-quotient length}$ **unfolding** $n\text{-def}$ $\mathfrak{G}Pn\text{-def}$ **by auto**

also have $\dots = \text{mset} (\text{map group.iso-class} ((\text{butlast } \mathfrak{L}.\text{quotients}) @ [\text{last } \mathfrak{L}.\text{quotients}])) + \{\# \text{group.iso-class} (G \text{Mod } \mathfrak{G} ! (n - 1)) \# \}$

using $\text{multisets} \mathfrak{G}\text{butlast} \mathfrak{L} \text{quots} \mathfrak{L} \text{notempty}$ **by simp**

also have $\dots = \text{mset} (\text{map group.iso-class} (\mathfrak{L}\text{butlast}.\text{quotients} @ [\mathfrak{G}Pn \text{Mod } \mathfrak{H} ! (m - 1) \cap \mathfrak{G} ! (n - 1)])) + \{\# \text{group.iso-class} (G \text{Mod } \mathfrak{G} ! (n - 1)) \# \}$

using $\mathfrak{L}.\text{quotients-butlast}$ $\mathfrak{L}.\text{last-quotient} \langle \text{length } \mathfrak{L} > 1 \rangle$ $\mathfrak{L}\text{sndlast Inteq} \mathfrak{L}\text{sndlast}$ **unfolding** $n\text{-def}$ **by auto**

also have $\dots = \text{mset} (\text{map group.iso-class } \mathfrak{R}\text{butlast}.\text{quotients}) + \{\# \text{group.iso-class} (\mathfrak{G}Pn \text{Mod } \mathfrak{H} ! (m - 1) \cap \mathfrak{G} ! (n - 1)) \# \} + \{\# \text{group.iso-class} (G \text{Mod } \mathfrak{G} ! (n - 1)) \# \}$

using $\text{multisets} \mathfrak{R}\mathfrak{L}\text{butlast}$ **by simp**

also have $\dots = \text{mset} (\text{map group.iso-class } \mathfrak{R}\text{butlast}.\text{quotients}) + \{\# \text{group.iso-class} (G \text{Mod } \mathfrak{H} ! (m - 1)) \# \} + \{\# \text{group.iso-class} (\mathfrak{H}Pm \text{Mod } \mathfrak{H} ! (m - 1) \cap \mathfrak{G} ! (n - 1)) \# \}$

using $\varphi \psi 2 \text{iso-classes-iff}$ $\text{grp} \mathfrak{G}Pn \text{Mod} \mathfrak{H}Pm \text{Int} \mathfrak{G}Pn.\text{is-group}$ $\text{grp} G \text{Mod} \mathfrak{H}Pm.\text{is-group}$ $\text{grp} G \text{Mod} \mathfrak{G}Pn.\text{is-group}$ $\text{grp} \mathfrak{H}Pm \text{Mod} \mathfrak{H}Pm \text{Int} \mathfrak{G}Pn.\text{is-group}$

by metis

also have $\dots = \text{mset} (\text{map group.iso-class } \mathfrak{R}\text{butlast}.\text{quotients}) + \{\# \text{group.iso-class} (\mathfrak{H}Pm \text{Mod } \mathfrak{H} ! (m - 1) \cap \mathfrak{G} ! (n - 1)) \# \} + \{\# \text{group.iso-class} (G \text{Mod } \mathfrak{H} ! (m - 1)) \# \}$

by simp

also have $\dots = \text{mset} (\text{map group.iso-class} ((\text{butlast } \mathfrak{R}.\text{quotients}) @ [\text{last } \mathfrak{R}.\text{quotients}]))) + \{\# \text{group.iso-class} (G \text{Mod } \mathfrak{H} ! (m - 1)) \# \}$

using $\mathfrak{R}.\text{quotients-butlast}$ $\mathfrak{R}.\text{last-quotient} \langle \text{length } \mathfrak{R} > 1 \rangle$ $\mathfrak{R}\text{sndlast Inteq} \mathfrak{R}\text{sndlast}$ **unfolding** $m\text{-def}$ **by auto**

also have $\dots = \text{mset} (\text{map group.iso-class } \mathfrak{H}\text{butlast}.\text{quotients}) + \{\# \text{group.iso-class}$

```

(G Mod  $\mathfrak{H} ! (m - 1) \#$ )
  using multisets  $\mathfrak{H}$  butlast  $\mathfrak{K}$  quots  $\mathfrak{K}$  notempty by simp
  also have ... = mset (map group.iso-class ((butlast comp  $\mathfrak{H}$ .quotients) @ [last comp  $\mathfrak{H}$ .quotients]))
  using comp  $\mathfrak{H}$ .quotients-butlast comp  $\mathfrak{H}$ .last-quotient length  $\mathfrak{H}$  big unfolding
m-def  $\mathfrak{H}$  Pm-def by auto
  also have ... = mset (map group.iso-class comp  $\mathfrak{H}$ .quotients) using quots  $\mathfrak{H}$  notempty
by simp
  finally show ?thesis .
qed
qed
qed

```

As a corollary, we see that the composition series of a fixed group all have the same length.

corollary (in *jordan-hoelder*) *jordan-hoelder-size*:

shows *length* $\mathfrak{G} = \text{length } \mathfrak{H}$

proof –

have *length* $\mathfrak{G} = \text{length comp}$ \mathfrak{G} .*quotients* + 1 by (*metis comp* \mathfrak{G} .*quotients-length*)

also have ... = *size* (*mset* (*map group.iso-class comp* \mathfrak{G} .*quotients*)) + 1 by (*metis length-map size-mset*)

also have ... = *size* (*mset* (*map group.iso-class comp* \mathfrak{H} .*quotients*)) + 1

using *jordan-hoelder-multisets is-group finite is-composition-series comp* \mathfrak{H} .*is-composition-series* by *metis*

also have ... = *length comp* \mathfrak{H} .*quotients* + 1 by (*metis size-mset length-map*)

also have ... = *length* \mathfrak{H} by (*metis comp* \mathfrak{H} .*quotients-length*)

finally show *?thesis*.

qed

end

References

[Ran05] Stuart Rankin. The jordan-hölder theorem, 2005.

[vR14] Jakob von Raumer. Secondary sylow theorems. *Archive of Formal Proofs*, January 2014. http://isa-afp.org/entries/Secondary_Sylow.shtml, Formal proof development.