The Jordan-Hölder Theorem

Jakob von Raumer

February 23, 2021

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 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 The Second Isomorphism Theorem for Groups 2
  1.1 Preliminaries .............................................. 2

2 Preliminary lemmas 7

3 More Facts about Subgroups 7

4 Facts about Normal Subgroups 9

5 Flattening the type of group carriers 18

6 Simple Groups 21

7 Facts about maximal normal subgroups 23

8 Normal series and Composition series 25
  8.1 Preliminaries ............................................... 25
  8.2 Normal Series ............................................. 27
  8.3 Composition Series ...................................... 32

9 Isomorphism Classes of Groups 46
10 The Jordan-Hölder Theorem

theory SndIsomorphismGrp
import
  HOL—Algebra.Coset
  Secondary-Sylow.SubgroupConjugation
begin

1 The Second Isomorphism Theorem for Groups

1.1 Preliminaries

lemma (in group) triv-subgroup:
  shows subgroup \{1\} G
unfolding subgroup-def by auto

lemma (in group) triv-normal-subgroup:
  shows \{1\} \triangleleft G
unfolding normal-def normal-axioms-def l-coset-def r-coset-def
using is-group triv-subgroup by auto

lemma (in group) normal-restrict-supergroup:
  assumes SsubG: subgroup S G
  assumes Nnormal: N \triangleleft G
  assumes N \subseteq S
  shows N \triangleleft (G\{carrier := S\})
proof –
interpret Sgrp: group G\{carrier := S\}
using SsubG by (rule subgroup-imp-group)
show \?thesis
proof(rule Sgrp.normalI)
  show subgroup N (G\{carrier := S\}) using assms is-group by (metis subgroup-of-subset normal-inv-iff)
next
  from SsubG have S \subseteq carrier G by (rule subgroup.subset)
  thus \(\forall x\in\text{carrier} (G\{\text{carrier} := S\}). N \#> G\{\text{carrier} := S\} x = x <\# G\{\text{carrier} := S\})\)
N
  using Nnormal unfolding normal-def normal-axioms-def l-coset-def r-coset-def
by fastforce
qed
qed

As this is maybe the best place this fits in: Factorizing by the trivial subgroup is an isomorphism.

lemma (in group) trivial-factor-iso:
  shows the-elem \in iso (G Mod \{1\}) G
proof –
  have group-hom G G (\lambda x. x) unfolding group-hom-def group-hom-axioms-def
  hom-def using is-group by simp
moreover have \((\lambda x. x)'\) carrier \(G = carrier G\) by simp
moreover have kernel \(G\) \(G (\lambda x. x) = \{1\}\) unfolding kernel-def by auto
ultimately show \(?\)thesis using group-hom.FactGroup-iso-set by force
qed

And the dual theorem to the previous one: Factorizing by the group itself gives the trivial group

**lemma (in group) self-factor-iso:**
shows \((\lambda X. the-elem ((\lambda x. 1) \cdot X)) \in iso (G Mod (carrier G)) (G\{ carrier := \{1\} })\)
proof –
  have group \((G\{carrier := \{1\} })\) by (metis subgroup-imp-group triv-subgroup)
  hence group-hom \(G\{carrier := \{1\} })\) \((\lambda x. 1)\) unfolding group-hom-def
  group-hom-axioms-def hom-def using is-group by auto
  moreover have \((\lambda x. 1)'\) carrier \(G = carrier (G\{carrier := \{1\} })\) by auto
  moreover have kernel \(G\{carrier := \{1\} })\) \((\lambda x. 1) = carrier G\) unfolding kernel-def by auto
  ultimately show \(?\)thesis using group-hom.FactGroup-iso-set by force
qed

This theory provides a proof of the second isomorphism theorems for groups. The theorems consist of several facts about normal subgroups.

The first lemma states that whenever we have a subgroup \(S\) and a normal subgroup \(H\) of a group \(G\), their intersection is normal in \(G\)

**locale second-isomorphism-grp = normal +**
fixes \(S:: a\) set
assumes subgrpS:subgroup \(S G\)

**context second-isomorphism-grp**
begin

**interpretation groupS: group G\{carrier := \{S\}\}**
using subgrpS by (metis subgroup-imp-group)

**lemma normal-subgrp-intersection-normal:**
shows \(S \cap H \triangleleft (G\{carrier := \{S\}\})\)
proof(auto simp: groupS.normal-inv-iff)
  from subgrpS is-subgroup have \(\land x. x \in \{S, H\} \Rightarrow subgroup x G\) by auto
  hence subgroup \((\cap \{S, H\}) G\) using subgroups-Inter by blast
  hence subgroup \((S \cap H) G\) by auto
  moreover have \(S \cap H \subseteq S\) by simp
  ultimately show subgroup \((S \cap H) (G\{carrier := \{S\}\})\) using is-group subgroup subgroup-of-subset subgrpS by metis
next
fix \(g h\)
assume g: \(g \in S\) and hH:h \(\in H\) and hS:h \(\in S\) 
  from \(g h H\) subgrpS show \(g \otimes h \otimes inv_{G\{carrier := \{S\}\}} g \in H\) by (metis inv-op-closed2 subgroup.mem-carrier m-inv-consistent)
lemma normal-set-mult-subgroup:
  shows subgroup (H <#> S) G
proof (rule subgroupI)
  show H <#> S ⊆ carrier G by (metis setmult-subset-G subgroup subset subgroupS subset)
next
  have 1 ∈ H 1 ∈ S using is-subgroup subgroupS subgroup.one-closed by auto
  hence 1 ⊗ 1 ∈ H <#> S unfolding set-mult-def by blast
  thus H <#> S ≠ {} by auto
next
  fix g
  assume g : g ∈ H <#> S
  then obtain h s where h · h ∈ H and s · s ∈ S and g · h · s unfolding set-mult-def by auto
  hence s ∈ carrier G by (metis subgroup mem-carrier subgroupS)
  with g · h · s obtain h' where h' · h' ∈ H and g = s ⊗ h' using coset-eq unfolding r-coset-def l-coset-def by auto
  with s have inv g = (inv h') ⊗ (inv s) by (metis inv-mult-group mem-carrier subgroup mem-carrier subgroupS)
  moreover from h' · s subgroupS have inv h' ∈ H inv s ∈ S using subgroup m-inv-closed m-inv-closed by auto
  ultimately show inv g ∈ H <#> S unfolding set-mult-def by auto
next
  fix g g'
  assume g : g ∈ H <#> S and h · g' ∈ H <#> S
  then obtain h h' s s' where hh' ss' · h ∈ H h' · H s ∈ S s' ∈ S and g = h · s unfolding set-mult-def by auto
  hence g ⊗ g' = (h ⊗ s) ⊗ (h' ⊗ s') by metis
  also from hh' ss' have inG. : h ∈ carrier G h' ∈ carrier G s ∈ carrier G s' ∈ carrier G using subgroupS mem-carrier subgroup mem-carrier by force+
  hence (h ⊗ s) ⊗ (h' ⊗ s') = h · (s ⊗ h') ⊗ s' using m-assoc by auto
  also from hh' ss' have inG obtain h'' where h'' · h'' ∈ H and s ⊗ h' = h'' ⊗ s unfolding coset-eq unfolding r-coset-def l-coset-def by fastforce
  hence h ⊗ (s ⊗ h') ⊗ s' = h ⊗ (h'' ⊗ s) ⊗ s' by simp
  also from h'' inG have ... = (h ⊗ h'') ⊗ (s ⊗ s') using m-assoc mem-carrier by auto
  finally have g ⊗ g' = h ⊗ h'' ⊗ (s ⊗ s').
  moreover with h'' hh' ss' have ... ∈ H <#> S unfolding set-mult-def using subgroupS subgroup.m-closed by fastforce
  ultimately show g ⊗ g' ∈ H <#> S by simp
qed

lemma oneH : 1 ∈ H by (metis is-subgroup subgroup one-closed)
lemma $H$-contained-in-set-mult:
  shows $H \subseteq H <\#> S$
proof auto
  have $1 \in S$ by (metis subgroup.one-closed subgrps)
  fix $x$
  assume $x : x \in H$
  with $\{1 \in S\}$ have $x \otimes 1 \in H <\#> S$ unfolding set-mult-def by force
  with $x$ show $x \in H <\#> S$ by (metis mem-carrier r-one)
qed

lemma $S$-contained-in-set-mult:
  shows $S \subseteq H <\#> S$
proof auto
  fix $s$
  assume $s : s \in S$
  with oneH have $1 \otimes s \in H <\#> S$ unfolding set-mult-def by force
  with $s$ show $s \in H <\#> S$ using subgrps subgroup.mem-carrier l-one by force
qed

lemma normal-intersection-hom:
  shows $\text{group-hom}((G\langle\cdot\rangle | \text{carrier} := S)) ((G\langle\cdot\rangle | \text{carrier} := H <\#> S)) \text{Mod } H \langle \lambda g. H <\#> g \rangle$
proof (auto del: equalityI simp: group-hom-def group-hom-axioms-def hom-def groupS.is-group)
  have $gr : \text{group}((G\langle\cdot\rangle | \text{carrier} := H <\#> S))$ by (metis normal-set-mult-subgroup subgroup-imp-group)
  moreover have $H \subseteq H <\#> S$ by (rule $H$-contained-in-set-mult)
  moreover have $\text{subgroup}(H <\#> S) G$ by (metis normal-set-mult-subgroup)
  ultimately have $H <\langle G\langle\cdot\rangle | \text{carrier} := H <\#> S\rangle \text{Mod } H)$ by (metis inv-op-closed2 is-subgroup normal-inv-iff)
  with $gr$ show $\text{group}((G\langle\cdot\rangle | \text{carrier} := H <\#> S)) \text{Mod } H)$ by (metis normal_factorgroup-is-group)
next
  fix $g$
  assume $g : g \in S$
  with subgrps have $1 \otimes g \in H <\#> S$ unfolding set-mult-def by fastforce
  with $g$ have $g \in H <\#> S$ by (metis l-one subgroup.mem-carrier subgrps)
  thus $H <\#> g \in \text{carrier}((G\langle\cdot\rangle | \text{carrier} := H <\#> S)) \text{Mod } H)$ unfolding Fact-Group-def RCOSETS-def r-coset-def by auto
next
  show $\forall x \ y. [x \in S; y \in S] \Longrightarrow H <\#> x \otimes y = H <\#> x <\#> (H <\#> y)$
    using normal.rcos-sum normal-axioms subgroup.mem-carrier subgrps by fast-force
qed

lemma normal-intersection-hom-kernel:
  shows $\text{kernel}((G\langle\cdot\rangle | \text{carrier} := S)) ((G\langle\cdot\rangle | \text{carrier} := H <\#> S)) \text{Mod } H)$ $\lambda g. H <\#> g = H \cap S$
proof –
have kernel \((G|\text{carrier := } S|) \mod H) (\lambda g \cdot \text{H #> g})\) unfolding kernel-def by auto
also have \(\ldots = \{ g \in S. H #> g = H \}\) unfolding FactGroup-def by auto
also have \(\ldots = \{ g \in S. g \in H \}\) by (metis coset-eq is-subgroup lcoset-join2 rcos-self subgroup.mem-carrier subgrpS)
also have \(\ldots = \{ g \in S. H #> g\}\) unfolding subgroup.mem-carrier subgrpS

also have \(\ldots = H \cap S\) by auto
finally show \(\text{thesis}\).

qed

lemma normal-intersection-hom-surj:
shows \((\lambda g. \text{H #> g}')\text{carrier (}(G|\text{carrier := } S|) \mod H)\)
proof auto
fix \(g\)
assume \(g \in S\)
hence \(g \in H <<\#>> S\) using S-contained-in-set-mult by auto
thus \(H #> g \in \text{carrier (}(G|\text{carrier := } H <<\#>> S|) \mod H)\) unfolding FactGroup-def RCOSETS-def r-coset-def by auto
next
fix \(x\)
assume \(x \in \text{carrier (}(G|\text{carrier := } H <<\#>> S|) \mod H)\)
then obtain \(h s\) where \(h:h \in H\) and \(s:s \in S\) and \(x = H #> (h \otimes s)\)
unfolding FactGroup-def RCOSETS-def r-coset-def set-mult-def by auto
hence \(x = (H #> h) #> s\) by (metis h s coset-mult-assoc mem-carrier subgroup.mem-carrier subgrpS subset)
also have \(\ldots = H #> s\) by (metis h is-group rcos-const)
finally have \(x = H #> s\).
with \(s\) show \(x \in (##>) H \cdot S\) by simp
qed

Finally we can prove the actual isomorphism theorem:

theorem normal-intersection-quotient-isom:
shows \((\lambda X. \text{the-elem ((}(\lambda g. H #> g) \cdot X)) \in \text{iso (}(G|\text{carrier := } S|) \mod (H \cap S))\)) unfolding normal-intersection-hom-kernel[symmetric] normal-intersection-hom normal-intersection-hom-surj by (metis group-hom.FactGroup-iso-set)

end

end

theory SubgroupsAndNormalSubgroups
imports
Secondary-Sylow.SndSylow
SndIsomorphismGrp
HOL-Algebra.Coset

6
2 Preliminary lemmas

A group of order 1 is always the trivial group.

lemma (in group) order-one-triv-iff:
  shows (order G = 1) = (carrier G = {1})
proof
  assume order:order G = 1
  then obtain x where x:carrier G = {x} unfolding order-def by (auto simp add: card-Suc-eq)
  hence 1 = x using one-closed by auto
  with x show carrier G = {1} by simp
next
  assume carrier G = {1}
  thus order G = 1 unfolding order-def by auto
qed

lemma (in group) finite-pos-order:
  assumes finite:finite (carrier G)
  shows 0 < order G
proof
  from one-closed finite show ?thesis unfolding order-def by (metis card-gt-0-iff subgroup-nonempty subgroup-self)
qed

lemma iso-order-closed:
  assumes φ ∈ iso G H
  shows order G = order H
using assms
unfolding order-def iso-def by (metis (no-types) bij-betw-same-card mem-Collect-eq)

3 More Facts about Subgroups

lemma (in subgroup) subgroup-of-restricted-group:
  assumes subgroup U (G[ carrier := H])
  shows U ⊆ H
using assms subgroup.subset by force

lemma (in subgroup) subgroup-of-subgroup:
  assumes group G
  assumes subgroup U (G[ carrier := H])
  shows subgroup U G
proof
  from assms(2) have U ⊆ H by (rule subgroup-of-restricted-group)
  thus U ⊆ carrier G by (auto simp:subset)
next
fix \( x \) \( y \)

have \( a : x \otimes y = x \otimes G \) \( \text{carrier} \):= \( H \) \( y \) by simp

assume \( x \in U \) \( y \in U \)

with \( \text{assms} \) a show \( x \otimes y \in U \) by \( \text{metis subgroup.m-closed} \)

next

have \( 1 \) \( G \) \( \text{carrier} \):= \( H \) \( = 1 \) by simp

with \( \text{assms} \) show \( 1 \in U \) by \( \text{metis subgroup.one-closed} \)

next

have subgroup \( H \) \( G \).

fix \( x \)

assume \( x \in U \)

with \( \text{assms}(2) \) have

\[ \text{inv}(G) \text{carrier} := H \] \( x \in U \) by \( \text{metis subgroup.m-inv-closed} \)

moreover from \( \text{assms} \) \( x \in U \) have \( x \in H \) by \( \text{metis in-mono subgroup-of-restricted-group} \)

with \( \text{assms}(1) \) \( \text{subgroup} \) \( H \) \( G \) : have \( \text{inv}(G) \text{carrier} := H \) \( x = \text{inv} x \) by \( \text{rule} \) \( \text{group.m-inv-consistent} \)

ultimately show \( \text{inv} x \in U \) by simp

qed

Being a subgroup is preserved by surjective homomorphisms

**Lemma (in subgroup)** surj-hom-subgroup:

assumes \( \varphi : \text{group-hom} \) \( G \) \( F \)

assumes \( \varphi \text{surj} : \varphi' \text{carrier} \) \( (carrier \) \( G \)) \( = \) carrier \( F \)

shows subgroup \( \varphi' \) \( H \) \( F \)

**Proof**

from \( \varphi \text{surj} \) show img-subset: \( \varphi' \) \( H \subseteq \text{carrier} \) \( F \) unfolding isom-def bij-betw-def by auto

next

fix \( f f' \)

assume \( h : f \in \varphi' \) \( H \) \( \text{and} \) \( h' : f' \in \varphi' \) \( H \)

with \( \varphi \text{surj} \) obtain \( g g' \) \( \text{where} \) \( g : g \in H \) \( f = \varphi g \) and \( g' : g' \in H \) \( f' = \varphi g' \) by auto

hence \( g \otimes_G g' \in H \) by \( \text{metis m-closed} \)

hence \( \varphi \) \( g \otimes_G g' \in \varphi' \) \( H \) by simp

with \( g g' \) \( \varphi \) \( \text{show} \) \( f \otimes_P f' \in \varphi' \) \( H \) using \( \text{group-hom.hom-mult} \) by fastforce

next

have \( \varphi 1 \in \varphi' \) \( H \) by auto

with \( \varphi \) show \( 1_P \in \varphi' \) \( H \) by \( \text{metis group-hom.hom-one} \)

next

fix \( f \)

assume \( f : f \in \varphi' \) \( H \)

then obtain \( g \) \( \text{where} \) \( g : g \in H \) \( f = \varphi g \) by auto

hence \( \text{inv} g \in H \) by auto

hence \( \varphi \) \( (\text{inv} g) \in \varphi' \) \( H \) by auto

with \( \varphi g \) subset show \( \text{inv} P f \in \varphi' \) \( H \) using \( \text{group-hom.hom-inv} \) by fastforce

qed

... and thus of course by isomorphisms of groups.

**Lemma** iso-subgroup:

assumes groups: \( \text{group} \) \( G \) \( \text{group} \) \( F \)
assumes \( HG: \text{subgroup} \ G \ G \)
assumes \( \varphi: \varphi \in \text{iso} \ G \ F \)
shows subgroup \((\varphi \cdot H) \ F\)

\[\text{proof} -\]
from groups \( \varphi \) have group-hom \( G \ F \ \varphi \) unfolding group-hom-def group-hom-axioms-def
iso-def by auto

moreover from \( \varphi \) have \( \varphi \cdot (\text{carrier} \ G) = \text{carrier} \ F \) unfolding iso-def bij-betw-def
by simp
moreover note \( HG \)
ultimately show \( \text{thesis} \) by (metis subgroup.surj-hom-subgroup)
qed

An isomorphism restricts to an isomorphism of subgroups.

\[\text{lemma iso-restrict:} -\]
assumes groups group \( G \ group \ F \)
assumes \( HG: \text{subgroup} \ G \ G \)
assumes \( \varphi: \varphi \in \text{iso} \ G \ F \)
shows \((\text{restrict} \ \varphi \ H) \in \text{iso} \ (G\{\text{carrier} := H\}) \ (F\{\text{carrier} := \varphi \cdot H\})\)
unfolding iso-def hom-def bij-betw-def inj-on-def
proof auto

fix \( g \ h \)
assume \( g \in H \ h \in H \)
hence \( g \in \text{carrier} \ G \ h \in \text{carrier} \ G \) by (metis HG subgroup.mem-carrier)+
thus \( \varphi \cdot (g \otimes \ G \ h) = \varphi \ g \otimes \ F \ \varphi \ h \) using \( \varphi \) unfolding iso-def hom-def by auto
next
fix \( g \ h \)
assume \( g \in H \ h \in H \ g \otimes \ G \ h \notin H \)
hence False using \( HG \) unfolding subgroup-def by auto
thus undefined = \( \varphi \ g \otimes \ F \ \varphi \ h \) by auto
next
fix \( g \ h \)
assume \( g: \ g \in H \ and \ h: \ h \in H \) and eq: \( \varphi \ g = \varphi \ h \)
hence \( g \in \text{carrier} \ G \ h \in \text{carrier} \ G \) by (metis HG subgroup.mem-carrier)+
with eq show \( g = h \) using \( \varphi \) unfolding iso-def bij-betw-def inj-on-def by auto
qed

The intersection of two subgroups is, again, a subgroup

\[\text{lemma (in group) subgroup-intersect:} -\]
assumes subgroup \( H \ G \)
assumes subgroup \( H' \ G \)
shows subgroup \((H \cap H') \ G\)
using assms unfolding subgroup-def by auto

\[\text{4 Facts about Normal Subgroups}\]

\[\text{lemma (in normal) is-normal:} -\]
shows \( H \triangleleft G \)
by (metis coset-eq is-subgroup normalI)
Being a normal subgroup is preserved by surjective homomorphisms.

**Lemma (in normal) surj-hom-normal-subgroup:**
- Assumes $\varphi: \text{group-hom } G \to F$
- Assumes $\varphi$ surj: $\varphi'(\text{carrier } G) = \text{carrier } F$
- Shows $(\varphi' \triangleleft F) \triangleleft F$

**Proof** (rule group.normalI)
- From $\varphi$ show group $F$ unfolding group-hom-def group-hom-axioms-def by simp
- Next from $\varphi$ surj show subgroup $(\varphi' \triangleleft F)$ by (rule surj-hom-subgroup)
- Next show $\forall x \in \text{carrier } F$. $\varphi' x = x <\#_F \varphi' \triangleleft F$

**Proof**
- Fix $f$
- Assume $f: f \in \text{carrier } F$
- With $\varphi$ surj obtain $g$ where $g: g \in \text{carrier } G \Rightarrow f = \varphi \circ g$ by auto
- Hence $\varphi' \triangleleft F = \varphi' \triangleleft F = \varphi' \circ g$ by simp
- Also have $\ldots = (\lambda x. (\varphi \circ x) \otimes_F (\varphi g))' \varphi' \triangleleft F$ unfolding r-coset-def image-def by auto
- Also have $\ldots = (\lambda x. \varphi (g \otimes x))' \varphi' \triangleleft F$ unfolding l-coset-def image-def by auto
- Also have $\ldots = (\lambda x. (\varphi g) \otimes_F (\varphi x))' \varphi' \triangleleft F$ unfolding l-coset-def image-def by auto
- Finally show $\varphi' \triangleleft F = f = f <\#_F \varphi' \triangleleft F$.

**Qed**

Being a normal subgroup is preserved by group isomorphisms.

**Lemma iso-normal-subgroup:**
- Assumes groups: group $G$ group $F$
- Assumes $H: H \triangleleft G$
- Assumes $\varphi: \varphi \in \text{iso } G \to F$
- Shows $(\varphi' \triangleleft F) \triangleleft F$

**Proof**
- From groups $\varphi$ have group-hom $G \to F$ unfolding group-hom-def group-hom-axioms-def iso-def by auto
- Moreover from $\varphi$ have $\varphi'(\text{carrier } G) = \text{carrier } F$ unfolding iso-def bij-betw-def by simp
- Moreover note $H G$
- Ultimately show $\exists \triangleleft F$ using normal.surj-hom-normal-subgroup by metis

**Qed**

The trivial subgroup is a subgroup:

**Lemma (in group) triv-subgroup:**
- Shows subgroup $\{1\} \triangleleft G$
unfolding subgroup-def by auto

The cardinality of the right cosets of the trivial subgroup is the cardinality of the group itself:

**lemma** (in group) card-rcosets-triv:
  assumes finite (carrier G)
  shows card (rcosets {1}) = order G

**proof** –
  have subgroup {1} G by (rule triv-subgroup)
  with assms have card (rcosets {1}) * card {1} = order G
  using lagrange by blast
  thus ?thesis by (auto simp: card-Suc-eq)
qed

The intersection of two normal subgroups is, again, a normal subgroup.

**lemma** (in group) normal-subgroup-intersect:
  assumes M \triangleleft G and N \triangleleft G
  shows M \cap N \triangleleft G

using assms subgroup-intersect is-group normal-inv-iff by simp

The set product of two normal subgroups is a normal subgroup.

**lemma** (in group) setmult-lcos-assoc:
  \[ [H \subseteq carrier G; K \subseteq carrier G; x \in carrier G] \implies (x \lt# H) \lt# K = x \lt# (H \lt# K) \]
by (force simp add: l-coset-def set-mult-def m-assoc)

**lemma** (in group) normal-subgroup-set-mult-closed:
  assumes M \triangleleft G and N \triangleleft G
  shows M \lt# N \triangleleft G

**proof** (rule normalI)
  from assms show subgroup (M \lt# N) G
    using second-isomorphism-grp.normal-set-mult-subgroup normal-imp-subgroup
    unfolding second-isomorphism-grp-def second-isomorphism-grp-axioms-def by force
next
  show \forall x \in carrier G. M \lt# N \lt# x = x \lt# (M \lt# N)
  ** proof
    fix x
    assume x:x \in carrier G
    have M \lt# N \lt# x = M \lt# (N \lt# x) by (metis assms(1,2) normal-inv-iff
      setmult-reos-assoc subgroup.subset x)
    also have \ldots = M \lt# (x \lt# N) by (metis assms(2) normal.coset-eq x)
    also have \ldots = (M \lt# x) \lt# N by (metis assms(1,2) normal-imp-subgroup
      rcos-asso-lcos subgroup.subset x)
    also have \ldots = (x \lt# M) \lt# N by (metis assms(1) normal.coset-eq x)
    also have \ldots = x \lt# (M \lt# N) by (metis assms(1,2) normal-imp-subgroup
      setmult-lcos-assoc subgroup.subset x)
    finally show M \lt# N \lt# x = x \lt# (M \lt# N),
  qed
The following is a very basic lemma about subgroups: If restricting the carrier of a group yields a group it’s a subgroup of the group we’ve started with.

**lemma** (in group) restrict-group-imp-subgroup:
- **assumes** $H \subseteq \text{carrier}\ G$ group ($G[\text{carrier} := H]$)
- **shows** subgroup $H G$

**proof**
- **from** assms(1) show $H \subseteq \text{carrier}\ G$.
- **next**
  - fix $x y$
  - assume $x \in H \land y \in H$
  - hence $x \in \text{carrier}\ G[\text{carrier} := H]$ by auto
  - with assms(2) show $x \odot y \in H$ using assms(2) group.is-monoid monoid.m-closed by fastforce
- **next**
  - show $1 \in H$ using assms(2) group.is-monoid monoid.one-closed by fastforce
  - **next**
    - fix $x$
    - assume $x \in H$
    - hence $x \in \text{carrier}\ G[\text{carrier} := H]$ by auto
    - hence $\text{inv}\ G[\text{carrier} := H] x \in \text{carrier}\ G[\text{carrier} := H]$ using assms(2) group.inv-closed by fastforce
    - hence $\text{inv}\ G[\text{carrier} := H] x \in \text{carrier}\ G$ using $x$ assms(1) by auto
    - moreover have $\text{inv}\ G[\text{carrier} := H] x \odot x = 1$ using assms(2) group.l-inv $x$ by fastforce
    - moreover have $x \in \text{carrier}\ G$ using $x$ assms(1) by auto
    - ultimately have $\text{inv}\ G[\text{carrier} := H] x = \text{inv}\ x$ using inv-equality[symmetric]
      
      by auto
      
    - thus $\text{inv}\ x \in H$ using assms(2) group.inv-closed $x$ by fastforce

**qed**

A subgroup relation survives factoring by a normal subgroup.

**lemma** (in group) normal-subgroup-factorize:
- **assumes** $N \triangleleft G$ and $N \subseteq H$ and subgroup $H G$
- **shows** subgroup $(\text{rcosets}\ G[\text{carrier} := H] N) (G \text{ Mod } N)$

**proof** –
- **interpret** $G\text{ModN} :\ G \text{ Mod } N$ using assms(1) by (rule normal фактогрупп-ис-групп)
- have $N \triangleleft G[\text{carrier} := H]$ using assms by (metis normal-restrict-supergroup)
- hence $\text{grpHN} :\ G[\text{carrier} := H] \text{ Mod } N$ by (rule normal фактогрупп-ис-групп)
- have $(\langle\#\rangle G[\text{carrier} := H]) = (\lambda U K. (\bigcup h \in U. \bigcup k \in K. \{ h \odot G[\text{carrier} := H]\ k\}))$ using set-mult-def by metis
  - moreover have $\ldots = (\lambda U K. (\bigcup h \in U. \bigcup k \in K. \{ h \odot G\}))$ by auto
  - moreover have $(\langle\#\rangle) = (\lambda U K. (\bigcup h \in U. \bigcup k \in K. \{ h \odot k\}))$ using set-mult-def by metis
  - ultimately have $(\langle\#\rangle G[\text{carrier} := H]) = (\langle\#\rangle G)$ by simp

12
A normality relation survives factoring by a normal subgroup.

**Lemma** (in group) normality-factorization:

**Assumes** \( NG : N < G \) and \( NH : N \subseteq H \) and \( HG : H < G \)

**Shows** \( (\text{rcosets}_{G\langle\text{carrier} := H\rangle} N) \triangleleft (G \text{ Mod } N) \)

**Proof** —

**From** \( \text{assms}(1) \) **Interpret** \( G\text{ModN} : \text{group } G \text{ Mod } N \) by \( (\text{metis normal_factorgroup_is-group}) \)

**Show** \( \text{?thesis} \)

**Proof** (**auto simp**: \( G\text{ModN.normal-inv-iff} \))

**From** \( \text{assms} \) **Show** subgroup \( (\text{rcosets}_{G\langle\text{carrier} := H\rangle} N) \) \( (G \text{ Mod } N) \) using

normal-imp-subgroup normal-subgroup-factorize by **force**

**Next**

**Fix** \( U \) \( V \)

**Assume** \( U : U \in \text{carrier } (G \text{ Mod } N) \) and \( V : V \in \text{rcosets}_{G\langle\text{carrier} := H\rangle} N \)

**Then obtain** \( g \) where \( g : g \in \text{carrier } G \) \( U = N \triangleleft g \) unfolding **FactGroup-def**

**RCOSETS-def** by **auto**

**From** \( V \) **Obtain** \( h \) where \( h : h \in H \) \( V = N \triangleleft h \) unfolding **FactGroup-def**

**RCOSETS-def** \( \text{r-coset-def} \) by **auto**

**Hence** \( hG : h \in \text{carrier } G \) using \( HG \) normal-imp-subgroup subgroup.mem-carrier by **force**

**Hence** \( ghG : g \otimes h \in \text{carrier } G \) using \( g \) m-closed by **auto**

**From** \( g \) \( h \) **Have** \( g \otimes h \otimes \text{inv } g \in H \) using \( HG \) normal-inv-iff by **auto**

**Moreover** **Have** \( U \triangleleft V \triangleleft inv_{G \text{ Mod } N} U = N \triangleleft (g \otimes h \otimes \text{inv } g) \)

**Proof** —

**From** \( g \) \( U \) **Have** \( inv_{G \text{ Mod } N} U = N \triangleleft inv \) using \( NG \) normal.inv-FactGroup

**Normal.rcos-inv** by **fastforce**

**Hence** \( U \triangleleft V \triangleleft inv_{G \text{ Mod } N} U = (N \triangleleft inv \) using \( g \) \( h \) by **simp**

**Also have** \( ... = N \triangleleft (g \otimes h) \triangleleft inv \) using \( g \) \( hG \) **NG**

normal.rcos-sum by **force**

**Also have** \( ... = N \triangleleft (g \otimes h \otimes inv \) using \( g \) inv-closed \( ghG \) **NG**

normal.rcos-sum by **force**

**Finally** **Show** \( \text{?thesis} \).

**Qed**

**Ultimately** **Show** \( U \triangleleft V \triangleleft inv_{G \text{ Mod } N} U \in \text{rcosets}_{G\langle\text{carrier} := H\rangle} N \)

unfolding **RCOSETS-def** \( \text{r-coset-def} \) by **auto**

**Qed**

Factoring by a normal subgroups yields the trivial group iff the subgroup is the whole group.
lemma (in normal) fact-group-trivial-iff:
  assumes finite (carrier G)
  shows (carrier (G Mod H) = \{1 \_ G Mod H\}) = (H = carrier G)
proof
  assume carrier (G Mod H) = \{1 \_ G Mod H\}
  moreover with assms lagrange have order (G Mod H) * card H = order G
  unfolding FactGroup-def order-def using is-subgroup by force
  ultimately have card H = order G unfolding order-def by auto
  thus H = carrier G using subgroup.subset is-subgroup assms card-subset-eq
unfolding order-def
  by metis
next
  from assms have ordgt0:order G > 0 unfolding order-def by (metis subgroup.finite-imp-card-positive subgroup-self)
  assume H = carrier G
  hence card H = order G unfolding order-def by simp
  with assms is-subgroup lagrange have card (rcosets H) * order G = order G by metis
  with ordgt0 have card (rcosets H) = 1 by (metis mult-eq-self-implies-10 mult.commute neg0-conv)
  hence order (G Mod H) = 1 unfolding order-def FactGroup-def by auto
  thus carrier (G Mod H) = \{1 \_ G Mod H\} using factorgroup-is-group by (metis group.order-one-triv-iff)
qed

Finite groups have finite quotients.
lemma (in normal) factgroup-finite:
  assumes finite (carrier G)
  shows finite (rcosets H)
using assms unfolding RCOSETS-def by auto

The union of all the cosets contained in a subgroup of a quotient group acts as a representation for that subgroup.
lemma (in normal) factgroup-subgroup-union-char:
  assumes subgroup A (G Mod H)
  shows (\bigcup A) = \{x \in carrier G. H \#> x \in A\}
proof
  show \bigcup A \subseteq \{x \in carrier G. H \#> x \in A\}
  proof
    fix x
    assume x:x \in \bigcup A
    then obtain a where a:a \in A x \in a by auto
    with assms have xx:x \in carrier G using subgroup.subset unfolding FactGroup-def RCOSETS-def r-coset-def by force
    from assms a obtain y where y:y \in carrier G a = H \#> y using subgroup.subset unfolding FactGroup-def RCOSETS-def by force
    with a have x \in H \#> y by simp
    hence H \#> y = H \#> x using y-is-subgroup repr-independence by auto
    with y(\2) a(\1) have H \#> x \in A by auto
with xx show \( x \in \{ x \in \text{carrier } G. \ H \triangleright x \in A \} \) by simp

qed

next

show \( \{ x \in \text{carrier } G. \ H \triangleright x \in A \} \subseteq \bigcup A \)

proof

fix \( x \)

assume \( x : x \in \{ x \in \text{carrier } G. \ H \triangleright x \in A \} \)

hence \( xx : x \in \text{carrier } G. H \nleq x \in A \) by auto

moreover have \( x \in H \triangleright x \) by (metis is-subgroup rcos-self xx (1))

ultimately show \( x \in \bigcup A \) by auto

qed

qed

lemma (in normal) factgroup-subgroup-union-subgroup:

assumes subgroup \( A \ (G \mod H) \)

shows subgroup \((\bigcup A) G\)

proof –

have subgroup \( \{ x \in \text{carrier } G. \ H \triangleright x \in A \} G \)

proof

show \( \{ x \in \text{carrier } G. \ H \triangleright x \in A \} \subseteq \text{carrier } G \) by auto

next

fix \( x \ y \)

assume \( x \in \{ x \in \text{carrier } G. \ H \triangleright x \in A \} \) and \( y \in \{ x \in \text{carrier } G. \ H \triangleright x \in A \} \)

hence \( xy : x \in \text{carrier } G. H \nleq x \in A \) by auto

hence \( xyG : x \otimes y \in \text{carrier } G \) by (metis m-closed)

from assms \( x \ y \) have \( (H \triangleright x) \nleq (H \triangleright y) \in A \) using subgroup.m-closed unfolding FactGroup-def by fastforce

hence \( H \triangleright (x \otimes y) \in A \) by (metis rcos-sum x (1) y (1))

with \( xyG \) show \( x \otimes y \in \{ x \in \text{carrier } G. \ H \triangleright x \in A \} \) by simp

next

have \( H \triangleright 1 \in A \) using assms subgroup.one-closed unfolding FactGroup-def by (metis coset-mult-one monoid.select-convs (2) subset)

with assms one-closed show \( 1 \in \{ x \in \text{carrier } G. \ H \triangleright x \in A \} \) by simp

next

fix \( x \)

assume \( x \in \{ x \in \text{carrier } G. \ H \triangleright x \in A \} \)

hence \( xx : x \in \text{carrier } G. H \nleq x \in A \) by auto

hence \( invx : inv x \in \text{carrier } G \) using inv-closed by simp

from assms \( x \) have set-inv \( (H \triangleright x) \in A \) using subgroup.m-inv-closed by (metis inv-FactGroup subgroup.mem-carrier)

hence \( H \triangleright (inv x) \in A \) by (metis rcos-inv x (1))

with \( invx \) show \( inv x \in \{ x \in \text{carrier } G. \ H \triangleright x \in A \} \) by simp

qed

with assms factgroup-subgroup-union-char show \(?thesis \) by auto

qed

lemma (in normal) factgroup-subgroup-union-normal:

assumes \( A \triangleleft (G \mod H) \)
shows $\bigcup A \triangleleft G$

proof –

have $\{ x \in \text{carrier } G, H \triangleright x \in A \} \triangleleft G$

unfolding normal-def normal-axioms-def

proof auto

from assms show subgroup $\{ x \in \text{carrier } G, H \triangleright x \in A \} \triangleleft G$

by (metis (full-types) factgroup-subgroup-union-char factgroup-subgroup-union-subgroup normal-imp-subgroup)

next

interpret Anormal: normal $A (G \mod H)$ using assms by simp

fix $x$ $y$

assume $x \cdot x \in \text{carrier } G \ y \in \{ x \in \text{carrier } G, H \triangleright x \in A \} \triangleright x$

then obtain $x' \ 	ext{where} \ x' \in \{ x \in \text{carrier } G, H \triangleright x \in A \} \ y = x' \odot x$

unfolding r-coset-def by auto

hence $x' \cdot x' \in \text{carrier } G \ H \triangleright x' \in A$ by auto

from $x(1)$ have $Hx: H \triangleright x \in \text{carrier } (G \mod H)$ unfolding FactGroup-def

RCOSETS-def by force

with $x'$ have $(\inv G \mod H (H \triangleright x)) \odot G \mod H (H \triangleright x') \odot G \mod H (H \triangleright x) \in A$ using Anormal.inv-op-closed1 by auto

hence $(\set-inv (H \triangleright x)) \# (H \triangleright x') \# (H \triangleright x) \in A$ using inv-FactGroup Hx unfolding FactGroup-def by auto

hence $(H \triangleright (\inv x)) \# (H \triangleright x') \# (H \triangleright x) \in A$ using $x(1)$ by (metis rcos-inv)

hence $(H \triangleright (\inv x \odot x')) \# (H \triangleright x) \in A$ by (metis inv-closed rcos-sum $x'(1) \ x(1)$)

hence $H \triangleright (\inv x \odot x' \odot x) \in A$ by (metis inv-closed m-closed rcos-sum $x'(1) \ x(1)$)

moreover have $\inv x \odot x' \odot x \in \text{carrier } G$ using $x' x$ by (metis inv-closed m-closed)

ultimately have $\inv x \odot x' \odot x \in \{ x \in \text{carrier } G, H \triangleright x \in A \}$ by auto

hence xcoset: $x \odot (\inv x \odot x' \odot x) \in x \# \{ x \in \text{carrier } G, H \triangleright x \in A \}$

unfolding l-coset-def using $x(1)$ by auto

have $x \odot (\inv x \odot x' \odot x) = (x \odot \inv x) \odot x' \odot x$ by (metis Units-eq Units-inv-Units m-assoc m-closed $x'(1) \ x(1)$)

also have $\ldots = x' \odot y$ by (metis l-one r-inv $x'(1) \ x(1)$)

also have $\ldots = y$ by (metis y $x' \odot x$)

finally have $x \odot (\inv x \odot x' \odot x) = y$, with xcoset show $y \in x \# \{ x \in \text{carrier } G, H \triangleright x \in A \}$ by auto

next

interpret Anormal: normal $A (G \mod H)$ using assms by simp

fix $x$ $y$

assume $x \cdot x \in \text{carrier } G \ y \in x \# \{ x \in \text{carrier } G, H \triangleright x \in A \}$

then obtain $x' \ 	ext{where} \ x' \in \{ x \in \text{carrier } G, H \triangleright x \in A \} \ y = x \odot x'$

unfolding l-coset-def by auto

hence $x' \cdot x' \in \text{carrier } G \ H \triangleright x' \in A$ by auto

from $x(1)$ have inex: $x \in \text{carrier } G$ by (rule inv-closed)

hence Hinex: $H \triangleright (\inv x) \in \text{carrier } (G \mod H)$ unfolding FactGroup-def

RCOSETS-def by force

with $x'$ have $(\inv G \mod H (H \triangleright \inv x)) \odot G \mod H (H \triangleright x') \odot G \mod H$
\[(H \not\supset \text{inv } x) \in A \text{ using } \text{inv } \text{normal.inv-op-closed} \text{ by } \text{auto}\]

hence \((\text{set-inv } (H \not\supset \text{inv } x)) \triangleleft H \not\supset (H \not\supset \text{inv } x) \in A \text{ using } \text{inv.FactGroup}\text{ H inv unfolding FactGroup-def by } \text{auto}\]

hence \((H \not\supset \text{inv } x) \triangleleft H \not\supset (H \not\supset \text{inv } x) \in A \text{ using } \text{inv.FactGroup}\text{ H inv unfolding FactGroup-def by } \text{auto}\]

\[
\text{hence } (H \not\supset x) \triangleleft (H \not\supset x') \triangleleft (H \not\supset \text{inv } x) \in A \text{ by } (\text{metis inv-inv} x(1))
\]

\[
\text{hence } (H \not\supset (x \otimes x')) \triangleleft (H \not\supset \text{inv } x) \in A \text{ by } (\text{metis rcos-sum } x'(1) x(1))
\]

moreover have \(x \otimes x' \otimes \text{inv } x \in \text{carrier } G\) using \(x x'\) by (metis inv-inv unfolded m-closed m-closed)

ultimately have \(x \otimes x' \otimes \text{inv } x \in \{x \in \text{carrier } G. H \not\supset x \in A\}\) by auto

hence \(\text{xosomal}(x \otimes x' \otimes \text{inv } x) \otimes x \in \{x \in \text{carrier } G. H \not\supset x \in A\} \not\supset x\)

unfolding r-coset-def using \(\text{inv } x\) by auto

have \((x \otimes x' \otimes \text{inv } x) \otimes x = (x \otimes x') \otimes (\text{inv } x \otimes x)\) by (metis Units-eq

Units-inv-Units m-assoc m-closed \(x'(1) x(1))\)

also have \(\ldots = x \otimes x'\) using \(x(1) l-\text{inv } x'(1) m\)-closed r-one by auto

also have \(\ldots = y\) by (metis \(y = x \otimes x')\)

finally have \(x \otimes x' \otimes \text{inv } x \otimes x = y.\)

with \(\text{xosomal}\) show \(y \in \{x \in \text{carrier } G. H \not\supset x \in A\} \not\supset x\) by auto

qed

with \(\text{assms}\) show \(?\text{thesis}\) by (metis (full-types) factgroup-subgroup-union-char normal-imp-subgroup)

qed

lemma (in normal) factgroup-subgroup-union-factor:

assumes subgroup \(A\) (\(G \text{ Mod } H\))

shows \(A = \text{rcosets}\{\text{carrier} := \bigcup A\} H\)

proof –

have \(A = \text{rcosets}\{\text{carrier} := \{x \in \text{carrier } G. H \not\supset x \in A\}\} H\)

proof auto

fix \(U\)

assume \(\text{U}\in A\)

then obtain \(x'\) where \(x':x' \in \text{carrier } G U = H \not\supset x'\) using \(\text{assms}\) subgroup subset unfolding FactGroup-def RCOSETS-def by force

with \(U\) have \(H \not\supset x' \in A\) by simp

with \(x'\) show \(U \in \text{rcosets}\{\text{carrier} := \{x \in \text{carrier } G. H \not\supset x \in A\}\} H\) unfolding RCOSETS-def r-coset-def by auto

next

fix \(U\)

assume \(U \in \text{rcosets}\{\text{carrier} := \{x \in \text{carrier } G. H \not\supset x \in A\}\}\) \(H\)

then obtain \(x'\) where \(x':x' \in \{x \in \text{carrier } G. H \not\supset x \in A\} U = H \not\supset x'\)

unfolding RCOSETS-def r-coset-def by auto

hence \(x' \in \text{carrier } G H \not\supset x' \in A\) by auto

with \(x'\) show \(U \in A\) by simp

qed

with \(\text{assms}\) show \(?\text{thesis}\) using factgroup-subgroup-union-char by auto

17
5 Flattening the type of group carriers

Flattening here means to convert the type of group elements from 'a set to 'a. This is possible whenever the empty set is not an element of the group.

definition flatten where
flatten (G:('a set, 'b) monoid-scheme) rep = ⟨carrier=(rep ' (carrier G)),
monoid.mult=(λ x y. rep ((the-inv-into (carrier G) rep x) ⊗_G (the-inv-into (carrier G) rep y)),
one=rep 1_G⟩

lemma flatten-set-group-hom:
  assumes group:group G
  assumes inj:inj-on rep (carrier G)
  shows rep ∈ hom G (flatten G rep)
unfolding hom-def
proof auto
  fix g
  assume g:g ∈ carrier G
  thus rep g ∈ carrier (flatten G rep) unfolding flatten-def by auto
next
  fix g h
  assume g:g ∈ carrier G and h:h ∈ carrier G
  hence rep g ∈ carrier (flatten G rep) rep h ∈ carrier (flatten G rep) unfolding flatten-def by auto
  hence rep g ⊗_flatten G rep h
  = rep (the-inv-into (carrier G) rep (rep g) ⊗_G the-inv-into (carrier G) rep (rep h)) unfolding flatten-def by auto
  also have ... = rep (g ⊗_G h) using inj g h by (metis the-inv-into-f-f)
  finally show rep (g ⊗_G h) = rep g ⊗_flatten G rep h.
qed

lemma flatten-set-group:
  assumes group:group G
  assumes inj:inj-on rep (carrier G)
  shows group (flatten G rep)
proof (rule groupI)
  fix x y
  assume x:x ∈ carrier (flatten G rep) and y:y ∈ carrier (flatten G rep)
  define g h
  where g = the-inv-into (carrier G) rep x
  and h = the-inv-into (carrier G) rep y
  hence x ⊗_flatten G rep y = rep (g ⊗_G h) unfolding flatten-def by auto
moreover from g-def h-def have g ∈ carrier G h ∈ carrier G
  using inj x y the-inv-into unfolding flatten-def by (metis partial-object.select-convs(1)
subset-refl)+
  hence g ⊗_G h ∈ carrier G by (metis group group.is-monoid monoid.m-closed)
hence rep \((g \otimes_G h)\) \(\in\) carrier (flatten G rep) unfolding flatten-def by simp
ultimately show \(x \otimes_{\text{flatten G rep}} y\) \(\in\) carrier (flatten G rep) by simp
next
show 1_{\text{flatten G rep}} \(\in\) carrier (flatten G rep) unfolding flatten-def by (simp add: group group.is-monoid)
next
fix \(x\) \(y\) \(z\)
assume \(x\):\(x\) \(\in\) carrier (flatten G rep) and \(y\):\(y\) \(\in\) carrier (flatten G rep) and \(z\):\(z\) \(\in\) carrier (flatten G rep)
define \(g\) \(h\) \(k\)
where \(g = \text{the-inv-into (carrier G) rep}\ x\)
and \(h = \text{the-inv-into (carrier G) rep}\ y\)
and \(k = \text{the-inv-into (carrier G) rep}\ z\)
hence \(x \otimes_{\text{flatten G rep}} y \otimes_{\text{flatten G rep}} z\) \(=\) \((\text{rep } (g \otimes_G h)) \otimes \text{flatten G rep } z\)
unfolding flatten-def by auto
also have \(\ldots\) \(=\) \(\text{rep (the-inv-into (carrier G) rep } (g \otimes_G h) \otimes_G k)\) using k-def unfolding flatten-def by auto
also from g-def h-def k-def have ghkG:\(g\) \(\in\) carrier G \(h\) \(\in\) carrier G \(k\) \(\in\) carrier G
using inj \(x\) \(y\) \(z\) the-inv-into onto unfolding flatten-def by fastforce+
hence ghkG:ghk \(\in\) carrier G and hkk: \(h \otimes_G k\) \(\in\) carrier G by (metis group group.is-monoid monoid.m-closed)+
hence \(\text{rep (the-inv-into (carrier G) rep } (g \otimes_G h)) \otimes_G k\) \(=\) \(\text{rep } ((g \otimes_G h) \otimes_G k)\)
unfolding flatten-def using inj the-inv-into f-f by fastforce
also have \(\ldots\) \(=\) \(\text{rep } (g \otimes_G h) \otimes_G k\) using group group.is-monoid ghkG monoid.m-assoc by fastforce
also have \(\ldots\) \(=\) \(x \otimes_{\text{flatten G rep}} (h \otimes_G k)\) unfolding g-def flatten-def
using hk inj the-inv-into f-f by fastforce
also have \(\ldots\) \(=\) \(x \otimes_{\text{flatten G rep}} (y \otimes_{\text{flatten G rep}} z)\) unfolding h-def k-def
flatten-def using x y by force
finally show \(x \otimes_{\text{flatten G rep}} y \otimes_{\text{flatten G rep}} z\) \(=\) \(x \otimes_{\text{flatten G rep}} (y \otimes_{\text{flatten G rep}} z)\).
next
fix \(x\)
assume \(x\):\(x\) \(\in\) carrier (flatten G rep)
define \(g\) where \(g = \text{the-inv-into (carrier G) rep } x\)
hence gG:ghG \(\in\) carrier G using inj x unfolding flatten-def using the-inv-into
by force
have 1G: \(1_G\) \(\in\) (carrier G) by (simp add: group group.is-monoid)
hence the-inv-into (carrier G) \(\text{rep } (1_{\text{flatten G rep}})\) \(=\) \(1_G\) unfolding flatten-def
using the-inv-into f-f inj by force
hence 1_{\text{flatten G rep}} \otimes_{\text{flatten G rep}} x = \text{rep } (1_G \otimes_G g)\) unfolding flatten-def g-def by simp
also have \(\ldots\) \(=\) \(\text{rep } g\) using gG group by (metis group.is-monoid monoid.l-one)
also have \(\ldots\) \(=\) \(x\) unfolding g-def using inj x f-the-inv-into f unfolding
flatten-def by force
finally show 1_{\text{flatten G rep}} \otimes_{\text{flatten G rep}} x = x.
next
from group inj have hom:rep ∈ hom G (flatten G rep) using flatten-set-group-hom
by auto
fix x
assume x:x ∈ carrier (flatten G rep)
define g where g = the-inv-into (carrier G) rep x
hence gG: g ∈ carrier G using inj x unfolding flatten-def using the-inv-into
by force
hence invG: invG g ∈ carrier G by (metis group group.inv-closed)
hence rep (invG g) ⊗ flatten G rep x = rep (invG g) ⊗ flatten G rep (rep g)
  unfolding g-def using f-the-inv-into-f inj x unfolding flatten-def by fastforce
hence rep (invG g) ⊗ flatten G rep x = 1 using invG gG by (metis group-group.inv
hence rep (invG g) ⊗ flatten G rep x = 1 using invG gG by (metis group-group.inv
hence rep (invG g) ⊗ flatten G rep x = 1 by auto
ultimately show ∃y ∈ carrier (flatten G rep). y ⊗ flatten G rep x = 1 by auto
qed

lemma (in normal) flatten-set-group-mod-inj:
shows inj-on (λU. SOME g. g ∈ U) (carrier (G Mod H))
proof (rule inj-onI)
fix U V
assume U: U ∈ carrier (G Mod H) and V: V ∈ carrier (G Mod H)
then obtain g h where g: U = H #> g g ∈ carrier G and h: V = H #> h h
  in carrier G
  unfolding FactGroup-def RCOSETS-def by auto
hence notempty: U ≠ {} and V ≠ {} by (metis empty-iff is-subgroup rcos-self+)
assume (SOME g. g ∈ U) = (SOME g. g ∈ V)
with notempty have (SOME g. g ∈ U) ∈ U ∩ V by (metis IntI ex-in-conv
somet)
thus U = V by (metis Int-iff g h is-subgroup repr-independence)
qed

lemma (in normal) flatten-set-group-mod:
shows group (flatten (G Mod H) (λU. SOME g. g ∈ U))
using factorygroup-is-group flatten-set-group-mod-inj by (rule flatten-set-group)

lemma (in normal) flatten-set-group-mod-iso:
shows (λU. SOME g. g ∈ U) ∈ iso (G Mod H) (flatten (G Mod H) (λU. SOME g. g ∈ U))
unfolding iso-def bij-betw-def
apply (auto)
apply (metis flatten-set-group-mod-inj factorygroup-is-group flatten-set-group-hom)
apply (rule flatten-set-group-mod-inj)
unfolding flatten-def apply (auto)
theory SimpleGroups
imports
  SubgroupsAndNormalSubgroups
  Secondary-Sylow.SndSylow
  SndIsomorphismGrp
begin

6 Simple Groups

locale simple-group =
  group +
  assumes order-gt-one: order G > 1
  assumes no-real-normal-subgroup: \( \forall H. H \triangleleft G \implies (H = \text{carrier } G \lor H = \{1\}) \)

lemma (in simple-group) is-simple-group: simple-group G by (rule simple-group-axioms)

Simple groups are non-trivial.

lemma (in simple-group) simple-not-triv: carrier G \neq \{1\} using order-gt-one unfolding order-def by auto

Every group of prime order is simple

lemma (in group) prime-order-simple:
  assumes prime: prime (order G)
  shows simple-group G
proof
  from prime show 1 < order G unfolding prime-nat-iff by auto
next
  fix H
  assume H \triangleleft G
  hence HG: subgroup H G unfolding normal-def by simp
  hence card H \leq order G by (rule card-subgrp-ded)
  with prime have card H = 1 \lor card H = order G unfolding prime-nat-iff by simp
  thus H = \text{carrier } G \lor H = \{1\}
proof
  assume card H = 1
  moreover from HG have 1 \in H by (metis subgroup.one-closed)
  ultimately show \?thesis by (auto simp: card-Suc-eq)
next
  assume card H = order G
  moreover from HG have H \subseteq \text{carrier } G unfolding subgroup-def by simp
  moreover from prime have card (carrier G) > 1 unfolding order-def prime-nat-iff ..
  hence finite (carrier G) by (auto simp:card-ge-0-finite)

ultimately show \( \textit{thesis}\) unfolding order-def by (metis card-subset-eq)

qed

Being simple is a property that is preserved by isomorphisms.

\begin{verbatim}
lemma (in simple-group) iso-simple:
  assumes H:group H
  assumes iso: \( \varphi \in \text{iso } G\ H \)
  shows simple-group H
unfolding simple-group-def simple-group-axioms-def using assms(1)
proof (auto del: equalityI)
from iso have order G = order H unfolding iso-def order-def using bij-betw-same-card
by auto
with order-gt-one show Suc 0 < order H by simp
next
  have inv-iso:(inv-into (carrier G) \( \varphi \)) \( \in \text{iso } G\ G \) using iso
    by (simp add: iso-set-sym)
  fix \( N \)
  assume NH: \( N < H \) and Nneq1: \( N \neq \{1_H\} \)
  then interpret Nnormal: normal N H by simp
  define M where \( M = (\text{inv-into } G \varphi) \cdot N \)
  hence MG: \( M < G \) using inv-iso NH H by (metis is-group iso-normal-subgroup)
  have surj: \( \varphi \cdot \text{carrier } G = \text{carrier } H \) using iso unfolding iso-def bij-betw-def
    by simp
  hence MN: \( \varphi \cdot M = N \) unfolding M-def using Nnormal,subset image-inv-into-cancel
    by metis
  moreover have \( M \neq \{1\} \)
    proof (rule notI)
      assume M = \( \{1\} \)
      hence \( \varphi \cdot M = \{1\} \) by (metis (full-types) image-empty image-insert)
      hence \( \varphi \cdot M = \{1_H\} \) by (metis (lifting) Nnormal,is-subgroup MN calculation singleton-iff subgroup.oneclosed)
      thus False using Nneq1 MN by simp
    qed
  hence M = carrier G using no-real-normal-subgroup MG by auto
  ultimately show \( N = \text{carrier } H \) using surj by simp
qed
\end{verbatim}

As a corollary of this: Factorizing a group by itself does not result in a simple group!

\begin{verbatim}
lemma (in group) self-factor-not-simple: \( \neg \) simple-group \( (G \Mod (\text{carrier } G)) \)
proof
  assume asm:simple-group \( (G \Mod (\text{carrier } G)) \)
  have group \( \langle \text{carrier := } \{1\} \rangle \) by (metis subgroup-imp-group triv-subgroup)
    with asm self-factor-iso simple-group.iso-simple have simple-group \( (G\langle \text{carrier := } \{1\} \rangle) \)
    by auto
  thus False using simple-group.simple-not-triv by force
qed
\end{verbatim}
7 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 factgroup-is-group allI impI disjCI)
      from finite factgroup-finite factgroup-is-group group finite-pos-order have $gt0: 0 < \text{card } (rcosets } H)$
        unfolding FactGroup-def order-def by force
      from maxH.proper finite have carrier $(G \text{ Mod } H) \neq \{1G \text{ Mod } H\}$ using fact-group-trivial-iff by auto
      hence $1 \neq \text{order } (G \text{ Mod } H)$ using factgroup-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:$A' \neq \{1G \text{ Mod } H\}$
    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 unfolding FactGroup-def by force
    hence $H \subseteq A$ unfolding A-def by auto
    hence $A1: H \triangleleft (G\{\text{carrier := } A\})$ using $A2$ is-normal by (metis is-subgroup maxH.max-normal normal-restrict-supergroup subgroup-self)
    have $A3: A' = rcosets G\{\text{carrier := } A\} H$
unfolding $A$-def using factgroup-subgroup-union-factor $A$'normal normal-imp-subgroup by auto
from $A1$ interpret normalHA: normal $H$ $(G\langle\text{carrier} := A\rangle)$ by metis
have $H \subseteq A$ using normalHA.is-subgroup subgroup.subset by force
with $A2$ have $A = H \lor A = \text{carrier } G$ using maxH.max-normal by auto
thus $A' = \text{carrier } (G \text{ Mod } H)$
proof
  assume $A = H$
  hence $\text{carrier } (G\langle\text{carrier} := A\rangle \text{ Mod } H) = \{1(G\langle\text{carrier} := A\rangle \text{ Mod } H)\}$
by (metis finite is-group normalHA.fact-group-trivial-iff normalHA.subgroup-self normalHA.subset subgroup-finite subgroup-of-restricted-group subgroup-of-subgroup subset-antisym)
also have $\ldots = \{1 G \text{ Mod } H\}$ unfolding FactGroup-def by auto
finally have $A' = \{1 G \text{ Mod } H\}$ using $A3$ unfolding FactGroup-def by simp
with $A$'nottriv show $?thesis$.
next
  assume $A = \text{carrier } G$
  hence $(G\langle\text{carrier} := A\rangle \text{ Mod } H) = G \text{ Mod } H$ by auto
  thus $A' = \text{carrier } (G \text{ Mod } H)$ using $A3$ unfolding FactGroup-def by simp
qed
qed
next
  assume simple:simple-group $(G \text{ Mod } H)$
  show max-normal-subgroup $H G$
  proof
    from simple have $\text{carrier } (G \text{ Mod } H) \neq \{1 G \text{ Mod } H\}$ unfolding simple-group-def
    simple-group-axioms-def order-def by auto
    with finite fact-group-trivial-iff show $H \neq \text{carrier } G$ by auto
next
  fix $A$
  assume $A:A < G A \neq H A \neq \text{carrier } G$
  show $\neg H \subseteq A$
  proof
    assume $HA:H \subseteq A$
    hence $H < (G\langle\text{carrier} := A\rangle)$ by (metis $A(1)$ inv-op-closed2 is-subgroup
    normal-inv-iff normal-restrict-supergroup)
    then interpret normalHA: normal $H$ $(G\langle\text{carrier} := A\rangle)$ by simp
    from finite have finiteA:finite $A$ using $A(1)$ normal-imp-subgroup by (metis
    subgroup-finite)
    have rcosets$(G\langle\text{carrier} := A\rangle) H < G \text{ Mod } H$ using normality-factorization
    is-normal HA $A(1)$ by auto
    with simple have rcosets$(G\langle\text{carrier} := A\rangle) H = \{1 G \text{ Mod } H\} \lor rcosets(G\langle\text{carrier} := A\rangle) H$
    $= \text{carrier } (G \text{ Mod } H)$
    unfolding simple-group-def simple-group-axioms-def by auto
    thus $\neg False$
    proof
      assume rcosets$G\langle\text{carrier} := A\rangle H = \{1 G \text{ Mod } H\}$
      hence rcosets$G\langle\text{carrier} := A\rangle H = \{1 (G\langle\text{carrier} := A\rangle) \text{ Mod } H\}$ unfolding
FactGroup-def by auto
with finite A have H = A using normal HA. fact-group-trivial-iff unfolding
FactGroup-def by auto
with A(2) show ?thesis by simp
next
assume AHGH: rcosets G \{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 ∈ carrier G
hence H #> x ∈ rcosets H unfolding RCOSETS-def by auto
with AHGH have H #> x ∈ rcosets G \{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 ∈ H #> x' by (metis is-subgroup rcos-self x)
hence x ∈ A #> x' using HA unfolding r-coset-def by auto
thus x ∈ 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
SimpleGroups
MaximalNormalSubgroups
begin

8 Normal series and Composition series

8.1 Preliminaries

A subgroup which is unique in cardinality is normal:

lemma (in group) unique-sizes-subgrp-normal:
assumes \( \text{fin:finite} \) (carrier \( G \))
assumes \( \exists! Q. \ Q \in \text{subgroups-of-size} \ q \)
shows \( (\text{THE } Q. \ Q \in \text{subgroups-of-size} \ q) \triangleleft G \)

proof –
from \( \text{assms} \) obtain \( Q \) where \( Q \in \text{subgroups-of-size} \ q \) by \( \text{auto} \)
define \( Q \) where \( Q = (\text{THE } Q. \ Q \in \text{subgroups-of-size} \ q) \)
with \( \text{assms} \) have \( Q \in \text{subgroups-of-size} \ q \) using \( \text{the1 by} \) \( \text{metis} \)
hence \( Q \in \text{subgroups-of-size} \ q \) and \( \text{card} Q = q \)
unfolding \( \text{subgroups-of-size-def} \)
by \( \text{auto} \)
from \( Q \) have \( Q \triangleleft G \)
proof (rule \( \text{normalI} \))
fix \( g \)
assume \( g : g \in \text{carrier} \ G \)

hence \( \text{inv} g : \text{inv} g \in \text{carrier} \ G \) by \( \text{metis inv-closed} \)
with \( \text{fin} \ Q \) size have \( \text{conjugation-action}\ q (\text{inv} g) \in \text{subgroups-of-size} \ q \)
by \( \text{metis conjugation-is-size-invariant} \)
with \( \text{g Q size} \) have \( \text{(inv g)} < (\# (Q \ # > g) = Q \) by \( \text{rule conj-wo-inv} \)
qed
with \( Q \)-def show \( \exists! \) \( ?\text{thesis by} \) \( \text{simp} \)
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) \( \text{pq-order-unique-subgrp} \):
assumes finite: \( \text{finite} \) (carrier \( G \))
assumes orderG: \( \text{order} \ G = q \ast p \)
assumes primep: \( \text{prime} \ p \) and \( \text{primeq} : \text{prime} \ q \) and \( \text{pq} : p < q \)
shows \( \exists! Q. \ Q \in \text{subgroups-of-size} \ q \)
proof –
from \( \text{primep primeq pq have} \; \text{ndqvdp:<-} (g \text{ dvd p}) \) by \( \text{metis less-not-refl3 prime-nat-iff} \)
define \( \text{calM where} \; \text{calM} = \{ s. \ s \subseteq \text{carrier} \ G \land \text{card} \ s = q ^ 1 \} \)
define \( \text{RelM where} \; \text{RelM} = \{ (N1, N2). \ N1 \in \text{calM} \land N2 \in \text{calM} \land (\exists g \in \text{carrier} \ G. \ N1 = N2 \ # > g) \} \)
interpret \( \text{syl: snd-sylow} \ G \ q \# 1 \ p \) \( \text{calM RelM} \)
unfolding \( \text{snd-sylow-def} \; \text{snd-sylow-axioms-def} \; \text{snd-sylow-axioms-def} \)
using \( \text{is-group primeq orderG} \; \text{finite ndqvdp} \) \( \text{calM-def RelM-def by} \) \( \text{auto} \)
obtain \( Q \) where \( Q \in \text{subgroups-of-size} \ q \) by \( \text{metis (lifting, mono-tags) mem-Collect-eq power-one-right subgroups-of-size-def syl.sylow-thm} \)
thus \( ?\text{thesis} \)
proof (rule \( \text{ex1I} \))
fix \( P \)
assume \( P : P \in \text{subgroups-of-size} \ q \)
have \( \text{card} (\text{subgroups-of-size} \ q) \mod q = 1 \) by \( \text{metis power-one-right syl.p-sylow-mod-p} \)

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

26
then have \( \text{card} \ (\text{subgroups-of-size } q) = p \lor \text{card} \ (\text{subgroups-of-size } q) = 1 \)
using \( \text{primep} \) by (auto simp add: prime-nat-iff)
ultimately have \( \text{card} \ (\text{subgroups-of-size } q) = 1 \) using \( p \)
by auto
with \( Q P \) show \( P = Q \) by (auto simp:card-Suc-eq)
qed

... And this unique subgroup is normal.

corollary (in group) \( \text{pq-order-subgrp-normal} \):
assumes \( \text{finite} \): \( \text{finite} \ (\text{carrier } G) \)
assumes \( \text{orderG} \): \( \text{order } G = q \ast p \)
assumes \( \text{primep} \): \( \text{prime } p \) and \( \text{primeq} \): \( \text{prime } q \) and \( \text{pq} \): \( p < q \)
shows \( (\text{THE } Q, Q \in \text{subgroups-of-size } q) < G \)
using \( \text{assms} \) by (metis \( \text{pq-order-unique-subgrp} \) \( \text{unique-sizes-subgrp-normal} \))

The trivial subgroup is normal in every group.

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

### 8.2 Normal Series

We define a normal series as a locale which fixes one group \( G \) and a list \( \mathcal{G} \) 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 \( \mathcal{G} \)
assumes notempty: \( \mathcal{G} \neq [] \)
assumes hd: \( \text{hd } \mathcal{G} = \{1\} \)
assumes last: \( \text{last } \mathcal{G} = \text{carrier } G \)
assumes normal: \( \forall i. i + 1 < \text{length } \mathcal{G} \Rightarrow (\mathcal{G} ! i) < G \langle(\text{carrier} := \mathcal{G} ! (i + 1)) \rangle \)

lemma (in normal-series) \( \text{is-normal-series} \): \( \text{normal-series } G \ \mathcal{G} \) 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) \( \text{trivial-normal-series} \):
shows \( \text{normal-series } G \langle\{1\}, \text{carrier } G\rangle \)
unfolding \( \text{normal-series-def} \) \( \text{normal-series-axioms-def} \)
using \( \text{is-group trivial-subgroup-is-normal} \) 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 $\mathcal{G} = 2$
shows $\mathcal{G} = \{1\}$, carrier $G$
proof (rule nth-equalityI)
  from assms show length $\mathcal{G} = \length \{1\}$, carrier $G$ by auto
next
  show $\mathcal{G} \setminus i = \{1\}$, carrier $G$ if $i < \length \mathcal{G}$ for $i$
  proof
    have $i = 0 \vee i = 1$ using that assms by auto
    thus $\mathcal{G} \setminus i = \{1\}$, carrier $G$ using $i$ by simp
  next
    assume $i : i = 1$
    with assms have $\mathcal{G} \setminus i = \{1\}$, carrier $G$ using last $i$ by simp
  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 ($G | \text{carrier} := H$)
assumes HG: $H \triangleleft G$
shows normal-series $G$ ($\mathcal{G} \setminus \text{carrier} := (\mathcal{G} \setminus \text{carrier}) \setminus i | \text{carrier} := \mathcal{G} \setminus \text{carrier} (i + 1)$)
proof
  from normal interpret normalH: normal-series ($G | \text{carrier} := H$) $\mathcal{G}$.
  from normalH.hd have hd $\mathcal{G} = \{1\}$ by simp
  with normalH.notempty have hdTriv:hd ($\mathcal{G} \setminus \text{carrier} = \{1\}$) by (metis hd-append2)
  show ?thesis unfolding normal-series-def normal-series-axioms-def using is-group
  proof auto
    fix $x$
    assume $x \in \text{hd} (\mathcal{G} \setminus \text{carrier} G)$
    with hdTriv show $x = 1$ by simp
  next
    fix $i$
    assume $i : i < \length \mathcal{G}$
    show ($\mathcal{G} \setminus \text{carrier} G$) ! $i < G | \text{carrier} := (\mathcal{G} \setminus \text{carrier} G) ! \text{Suc } i$
    proof (cases $i + 1 < \length \mathcal{G}$)
      case True
      with normalH.normal have $\mathcal{G} ! i < G | \text{carrier} := \mathcal{G} ! (i + 1)$ by auto
    qed
  qed
with \( i \) have \((\mathcal{H} \circ [\text{carrier } G]) \mid i < G\{\text{carrier} := \mathcal{H} \mid (i + 1)\}\) using nth-append by metis

with True show \((\mathcal{H} \circ [\text{carrier } G]) \mid i < G\{\text{carrier} := (\mathcal{H} \circ [\text{carrier } G]) \mid (\text{Suc } i)\}\) using nth-append Suc-eq-plus1 by metis

next

case False

with \( i \) have \( i: i + 1 = \text{length } \mathcal{H} \) by simp
rom \( i \) have \((\mathcal{H} \circ [\text{carrier } G]) \mid i = \mathcal{H} \mid i \) by (metis nth-append)

also from \( i: \text{normalH.nontempty have } \ldots = \text{last } \mathcal{H} \) by (metis add-diff-cancel-right' last-conv-nth)

also from \( \text{normalH.last have } \ldots = H \) by simp

finally have \((\mathcal{H} \circ [\text{carrier } G]) \mid i = H. \)

ultimately show \(?\text{thesis using HG by auto}\)

qed

All entries of a normal series for \( G \) are subgroups of \( G \).

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

shows \( i < \text{length } \mathcal{G} \Longrightarrow \text{subgroup } (\mathcal{G} \mid i) G \)

proof –

have \( i + 1 < \text{length } \mathcal{G} \Longrightarrow \text{subgroup } (\mathcal{G} \mid i) G \)

proof (induction length \( \mathcal{G} - (i + 2) \) arbitrary; \( i \))

case \( 0 \)

hence \( i: i + 2 = \text{length } \mathcal{G} \) by simp

hence \( ii: i + 1 = \text{length } \mathcal{G} - 1 \) by force

from \( i \) normal have \( \mathcal{G} \mid i < G\{\text{carrier} := \mathcal{G} \mid (i + 1)\} \) by auto

with \( ii \) last notempty show subgroup \((\mathcal{G} \mid i) G\) using last-conv-nth normal-imp-subgroup by fastforce

next

case \( (\text{Suc } k) \)
rom \( \text{Suc}(3) \) normal have \( i: \text{subgroup } (\mathcal{G} \mid i) (G\{\text{carrier} := \mathcal{G} \mid (i + 1)\}) \)

using normal-imp-subgroup by auto

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

with \( \text{Suc} \) have \( \text{subgroup } (\mathcal{G} \mid (i + 1)) G \) by simp

with \( i \) show subgroup \((\mathcal{G} \mid i) G\) by (metis is-group subgroup subgroup-of-subgroup)

qed

moreover have \( i + 1 = \text{length } \mathcal{G} \Longrightarrow \text{subgroup } (\mathcal{G} \mid i) G \)

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

ultimately show \( i < \text{length } \mathcal{G} \Longrightarrow \text{subgroup } (\mathcal{G} \mid i) G \) 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 \( \mathcal{G} \mid (\text{length } \mathcal{G} - 2) < G \)

proof (cases \( 2 \leq \text{length } \mathcal{G} \))

case False

29
with notempty have length:length G = 1 by (metis Suc-eq-plus1 leI length-0-conv less-2-cases plus-nat.add-0)

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

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

next

case True

hence (length G - 2) + 1 < length G by arith

with normal last have G ! (length G - 2) ≺ G[carrier := G ! (length G - 2) + 1)] by auto

have 1 + (1 + (length G - (1 + 1))) = length G

using True le-add-diff-inverse by presburger

then have G ! (length G - 2) ≺ G[carrier := G ! (length G - 1)]

by (metis G ! (length G - 2) ≺ G[carrier := G ! (length 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 ≤ length G and 0 < i

shows normal-series (G|carrier := G ! (i - 1))) (take i 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 ∈ subgroups-of-size q), carrier G]

proof –

define H where H = (THE H. H ∈ subgroups-of-size q)

with assms have HG:H ≪ G by (metis pq-order-subgrp-normal)

then interpret groupH: group G[carrier := H]

unfolding normal-def by (metis subgroup-imp-group)

have normal-series (G[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:
The list of quotient groups has one less entry than the series itself:

**Lemma (in normal-series) quotients-length:**

shows \( \text{length} \text{ quotients} + 1 = \text{length} G \)

**Proof**

- have \( \text{length} \text{ quotients} + 1 = \text{length} [0..<(\text{length} G) - 1] + 1 \) unfolding quotients-def by simp
- also have \( ... = (\text{length} G - 1) + 1 \) by (metis diff-zero length-upt)
- also with notempty have \( ... = \text{length} G \)
- by (simp add: ac-simps)
- finally show \(?thesis \).

**QED**

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

**Lemma (in normal-series) last-quotient:**

assumes \( \text{length} G > 1 \)

shows \( \text{last quotients} = G \text{ Mod } G ! (\text{length} G - 1 - 1) \)

**Proof**

- from assms have \( \text{lsimp:} \text{length} G - 1 - 1 + 1 = \text{length} G - 1 \) 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 \( ... = \text{quotients} ! (\text{length} G - 1 - 1) \) by (metis add-diff-cancel-left'
- unfolding quotients-length add.commute)
- also have \( ... = G \{\text{carrier} := G ! ((\text{length} G - 1 - 1) + 1)| Mod G ! (\text{length} G - 1 - 1) \)
- unfolding quotients-def using assms by auto
- also have \( ... = G \{\text{carrier} := G ! (\text{length} G - 1)| Mod G ! (\text{length} G - 1 - 1) \}
- using \( \text{lsimp by simp} \)
- also have \( ... = G \text{ Mod } G ! (\text{length} G - 1 - 1) \) using last last-conv-nth notempty
- by force
- finally show \(?thesis \).

**QED**
finally show \( \text{hd} \ (\text{map} \ (\cdot) \ \mathcal{G}) = \{1_H\} \).

next

show last (\text{map} \ (\cdot) \ \mathcal{G}) = \text{carrier} \ H \land (\forall \ i. \ \text{Suc} \ i < \text{length} \ \mathcal{G} \Rightarrow \mathcal{G} \ ! \ i \prec H(\text{carrier} := \mathcal{G} \ ! \ i))

proof (auto del: equalityI)

have last (\text{map} \ (\cdot) \ \mathcal{G}) = \Psi \ (\text{carrier} \ H) \ \text{using last last-map notempty by metis}

also have \ldots = \text{carrier} \ H \ \text{using iso unfolding iso-def bij-betw-def by simp}

finally show last (\text{map} \ (\cdot) \ \mathcal{G}) = \text{carrier} \ H.

next

fix \ i

assume \ i : \text{Suc} \ i < \text{length} \ \mathcal{G}

hence norm:G \ i < G(\text{carrier} := G \ ! \ i) \ \text{using normal by simp}

moreover have restrict \ (\mathcal{G} \ ! \ i) \in iso \ (G(\text{carrier} := G \ ! \ i))

(H(\text{carrier} := \Psi \ G \ ! \ i))

by (metis H i is-group iso iso-restrict normal-series-subgroups)

moreover have group \ (G(\text{carrier} := G \ ! \ i)) \ \text{by (metis i normal-series-subgroups subgroup-imp-group)}

moreover hence subgroup \ (G \ ! \ i) \ G \ \text{by (metis i normal-series-subgroups subgroup-imp-group)}

moreover hence subgroup \ (G \ ! \ i) \ H \ \text{by (metis H is-group iso iso-subgroup)}

hence group \ (H(\text{carrier} := \Psi \ G \ ! \ i)) \ \text{by (metis H subgroup subgroup-is-group)}

ultimately have restrict \ (G \ ! \ i) \in iso \ (G \ ! \ i \prec H(\text{carrier} := \Psi \ G \ ! \ i))

using is-group H iso-normal-subgroup by (auto cong del: image-cong-simp)

moreover from norm have \ G \ ! \ i \subseteq G \ ! \ i \ \text{unfolding normal-def subgroup-def by auto}

hence \{y. \ \exists x \in G \ ! \ i. \ y = (if x \in G \ ! \ i \ then \ \Psi \ x \ else \ undefined)\} = \{y. \ \exists x \in G \ ! \ i. \ y = \Psi \ x\} \ \text{by auto}

ultimately show \ \Psi \ G \ ! \ i \prec H(\text{carrier} := \Psi \ G \ ! \ i) \ \text{unfolding restrict-def image-def by auto}

qed

qed

8.3 Composition Series

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

locale composition-series = normal-series +

assumes simplefact: \( \forall i. \ i + 1 < \text{length} \ \mathcal{G} \Rightarrow \text{simple-group} \ (G(\text{carrier} := \mathcal{G} \ ! \ (i + 1))) \ \text{Mod} \ \mathcal{G} \ ! \ i) \)

lemma (in composition-series) is-composition-series:

shows \ composition-series \ G \ \mathcal{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} \ \mathcal{G} = 1) = (\mathcal{G} = \{[1]\})

32
proof
assume length $\mathcal{G} = 1$
with hd have length $\mathcal{G} = length \{1\} \land (\forall i < length \mathcal{G}. \mathcal{G} ! i = \{1\} ! i)$ using
hd-conv-nth notempty by force
thus $\mathcal{G} = \{1\}$ using list-eq-iff-nth-eq by blast
next
assume $\mathcal{G} = \{1\}$
thus length $\mathcal{G} = 1$ by simp
qed

lemma (in composition-series) composition-series-triv-group:
shows (carrier $G = \{1\} = (\mathcal{G} = \{1\})$
proof
assume $G$:carrier $G = \{1\}$
have length $\mathcal{G} = 1$
proof (rule ccontr)
assume length $\mathcal{G} \neq 1$
with notempty have length:length $\mathcal{G} \geq 2$ 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 := $\mathcal{G}$
! $1$) Mod $\{1\}$ by force
moreover have $SG$:subgroup ($\mathcal{G} ! 1$) $G$ using length normal-series-subgroups
by auto
hence group ($G$carrier := $\mathcal{G} ! 1$) by (metis subgroup-imp-group)
ultimately have simple-group ($G$carrier := $\mathcal{G} ! 1$) using group.trivial-factor-iso
simple-group.iso-simple by fastforce
moreover from $SG$ $G$ have carrier ($G$carrier := $\mathcal{G} ! 1$) = $\{1\}$ unfolding
subgroup-def by auto
ultimately show False using simple-group.simple-not-triv by force
qed
thus $\mathcal{G} = \{1\}$ by (metis composition-series-length-one)
next
assume $\mathcal{G} = \{1\}$
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 $i + 1 < length \mathcal{G}$
assumes $i > 0$
shows $\mathcal{G} ! i \neq \{1\}$
proof
from assms have $(i - 1) + 1 < length \mathcal{G}$ by simp
hence simple:simple-group ($G$carrier := $\mathcal{G} ! ((i - 1) + 1)$) Mod $\mathcal{G} ! (i - 1)$)
using simplefact by auto
assume $i:G ! i = \{1\}$
moreover from assms have $(i - 1) + 1 = i$ by auto
ultimately have $G$carrier := $\mathcal{G} ! ((i - 1) + 1)$) Mod $\mathcal{G} ! (i - 1) = G$carrier

33
A composition series of a simple group always is its trivial one.

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

**shows** (simple-group G = (G = {1}, carrier G))

**proof**

**assume** simple: simple-group G

**have** length G > 1

**(rule ccontr)**

**assume** ¬ (length G ≤ 2)

**hence** length G = 1 by (simp add: Suc-leI antisym notempty)

**hence** carrier G = {1} using (metis composition-series-length-one composition-series-triv-group)

**hence** order G = 1 unfolding order-def by auto

**(rule disjE)**

**assume** G k = carrier G

**hence** G = G using notempty last last-conv-nth k-def

**(by (metis Nat.add-diff-assoc Nat.diff-cancel ¬ length G ≤ 2 add.commute nat-le-linear one-add-one))**

**from normal ksmall have** G k < G[carrier := G ! (k + 1)]

**(by simp)**

**from simplefact ksmall have** simplek: simple-group (G[carrier := G ! (k + 1)])

**(Mod G ! k) by simp**

**from simplefact ksmall have** simplek': simple-group (G[carrier := G ! ((k - 1) + 1)])

**(Mod G ! (k - 1)) by auto**

**have** G ! k < G using carrier k-def gt2 normal ksmall by force

**(with simp)**

**thus** False

**(proof (rule disjE))**

**assume** G ! k = carrier G

**hence** G[carrier := G ! (k + 1)]

**(Mod G ! k = G Mod (carrier G) using**
Two consecutive elements in a composition series are distinct.

**lemma** (in `composition-series`) entries-distinct:
- **assumes** finite:finite (carrier G)
- **assumes** i:i + 1 < length Ø
- **shows** Ø ! i ≠ Ø ! (i + 1)
- **proof**
  - **from** finite **have** finite (Ø ! (i + 1))
  - **using** i normal-series-subgroups subgroup subset rev-finite-subset by metis
  - **hence** fin:finite (carrier (G!carrier := Ø ! (i + 1))) by auto
  - **from** i **have** norm:Ø ! i < (G!carrier := Ø ! (i + 1)) by (rule normal)
  - **assume** Ø ! i = Ø ! (i + 1)
  - **hence** Ø ! i = carrier (G!carrier := Ø ! (i + 1)) by auto
  - **hence** carrier ((G!carrier := (Ø ! (i + 1)))) Mod (Ø ! i) = \{1_G!carrier := Ø ! (i + 1))\} Mod Ø ! i
    - **using** norm fin normal факт-group-trivial-iff by metis
    - **hence** ¬ simple-group ((G!carrier := (Ø ! (i + 1)))) Mod (Ø ! 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 ∈ 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 = (THE H. H ∈ subgroups-of-size q)
    - **from** assms have exi:∃!Q. Q ∈ (subgroups-of-size q) by (auto simp: pq-order-unique-subgrp)
      - **hence** Hsize:H ∈ subgroups-of-size q unfolding H-def using the1' by metis
      - **hence** HsubG:subgroup H G unfolding subgroups-of-size-def by auto
    - **then interpret** Hgroup: group G!carrier := H by (metis subgroup-imp-group)
      - **fix** i
        - **assume** i < Suc (Suc 0)
        - **hence** i = 0 ∨ i = 1 by auto
        - **thus** simple-group ((G!carrier := [H, carrier G] ! i)) Mod \{1\}, H, carrier G! i)
proof
  assume i:i = 0
  from Hsize have orderH:order (G\{carrier := H\}) = q unfolding subgroups-of-size-def order-def by simp
  hence order (G\{carrier := H\} Mod \{1\}) = q unfolding FactGroup-def using card-rcosets-triv order-def
  by (metis Hgroup.card-rcosets-triv HsubG finite monoid.cases-scheme monoid.select-convs(2) partial-object.select-convs(1) partial-object.update-convs(1) subgroup-finite)
  have normal \{1\} (G\{carrier := H\}) by (metis Hgroup.is-group Hgroup.normal-inv-iff HsubG group.trivial-subgroup-is-normal is-group singleton-iff subgroup.one-closed subgroup.subgroup-of-subgroup)
  hence group (G\{carrier := H\} Mod \{1\}) by (metis normal.factorgroup-is-group)
  with orderH primeq have simple-group (G\{carrier := H\} Mod \{1\}) by (metis order (G\{carrier := H\} Mod \{1\}) = q \ group.prime-order-simple)
  with i show \?thesis by simp
next
  assume i:i = 1
  from assms exi have H < 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 \$i and 0 < i
  shows composition-series (G\{carrier := \$i ! (i - 1)\}) (take i \$i)
unfolding composition-series-def composition-series-axioms-def
proof auto
  from assms show normal-series (G\{carrier := \$i ! (i - Suc 0)\}) (take i \$i) by (metis One-nat-def normal-series-prefix-closed)
next
  fix j
  assume j:Suc j < length \$Suc j < i
  with simplefact show simple-group (G\{carrier := \$i ! Suc j\} Mod \$i ! 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 \$
shows \( \text{simple-group } (G[\text{carrier} := \mathcal{G} \cup \{1\}]) \)

proof –

- from assms interpret compTake: composition-series \( G[\text{carrier} := \mathcal{G} \cup \{1\}] \) take \( 2 \mathcal{G} \) by (metis add-diff-cancel-right' composition-series-prefix-closed one-add-one zero-less-numeral)
- from assms have length (take 2 \( \mathcal{G} \)) = 2 by (metis add-diff-cancel-right' append-take-drop-id diff-diff-cancel length-append length-drop)
  hence (take 2 \( \mathcal{G} \)) = \((\{1\}(G[\text{carrier} := \mathcal{G} \cup \{1\}]), \text{carrier } (G[\text{carrier} := \mathcal{G} \cup \{1\}])\)) by (rule compTake.length-two-unique)
- thus \( \text{thesis } \) 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.

\textbf{lemma (in composition-series) composition-snd-simple-iff:}
 assumes \( i < \text{length } \mathcal{G} \)
 shows \( \text{simple-group } (G[\text{carrier} := \mathcal{G} \cup \{i\}]) = (i = 1) \)

proof
  assume simp:i: \( \text{simple-group } (G[\text{carrier} := \mathcal{G} \cup \{i\}]) \)
  hence \( i \neq \{1\} \) 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 \( G[\text{carrier} := \mathcal{G} \cup \{i\}] \) take (Suc \( i \)) \( \mathcal{G} \)
  using assms composition-series-prefix-closed by (metis diff-Suc-1 less-eq-Suc-le zero-less-Suc)
  from simp:i have (take (Suc \( i \)) \( \mathcal{G} \)) = \((\{1\}_G[\text{carrier} := \mathcal{G} \cup \{i\}], \text{carrier } (G[\text{carrier} := \mathcal{G} \cup \{i\}])\))
    by (metis compTake.composition-series-simple-group)
  hence length (take (Suc \( i \)) \( \mathcal{G} \)) = 2 by auto
  hence min (length \( \mathcal{G} \)) (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 \( 2 \leq \text{length } \mathcal{G} \) by simp
  with \( i \) show \( \text{simple-group } (G[\text{carrier} := \mathcal{G} \cup \{i\}]) \)
    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 \textit{maximal} normal subgroup.

\textbf{lemma (in composition-series) snd-to-last-max-normal:}
 assumes finite:finite (\text{carrier } G)
 assumes length:length \( \mathcal{G} > 1 \)
 shows \( \text{max-normal-subgroup } (\mathcal{G} \cup \text{length } \mathcal{G} - 2) G \)

unfolding max-normal-subgroup-def max-normal-subgroup-axioms-def

proof (auto del: equalityI)
  show \( \mathcal{G} \cup \text{length } \mathcal{G} - 2 < G \) by (rule normal-series-snd-to-last)

next
\[
\text{define } G' \text{ where } G' = \mathcal{G} ! (\text{length } \mathcal{G} - 2) \\
\text{from length have length21:length } \mathcal{G} - 2 + 1 = \text{length } \mathcal{G} - 1 \text{ by arith} \\
\text{from length have length } \mathcal{G} - 2 + 1 < \text{length } \mathcal{G} \text{ by arith} \\
\text{with simple\_fact have simple\_group } (G[\text{carrier := } \mathcal{G} ! ((\text{length } \mathcal{G} - 2) + 1)]) \\
\text{Mod } G' \text{ unfolding } G'\text{-def by auto} \\
\text{with length21 have simple\_last:simple\_group } (G \text{ Mod } G') \text{ using last notempty last\_conv\_nth by fastforce} \\
\{ \\
\text{assume snd\_to\_last\_eq} G' = \text{carrier } G \\
\text{hence carrier } (G \text{ Mod } G') = \{1_{G \text{ Mod } G'}\} \\
\text{using normal\_series\_snd\_to\_last finite normal\_fact\_group\_trivial\_iff unfolding } \\
G'\text{-def by metis} \\
\text{with snd\_to\_last\_eq have } \sim \text{simple\_group } (G \text{ Mod } G') \text{ by (metis self\_factor\_not\_simple)} \\
\text{with simple\_last show False unfolding } G'\text{-def by auto} \\
\} \\
\{ \\
\text{have } G'G' < G \text{ unfolding } G'\text{-def by (rule normal\_series\_snd\_to\_last)} \\
\text{fix } J \\
\text{assume } J\_J < G \text{ J } \neq G' \text{ J } \neq \text{carrier } G \text{ G' } \subseteq J \\
\text{hence } JG'G' \text{rcosets } (G[\text{carrier := } J]) \text{ G' } \subseteq G \text{ Mod } G' \text{ using normal\_\_factorization normal\_series\_snd\_to\_last unfolding } G'\text{-def by auto} \\
\text{from } G'G'J(1,J) \text{ have } G'J \text{G' } \subseteq (G[\text{carrier := } J]) \text{ by (metis normal\_imp\_subgroup normal\_\_restrict\_supergroup)} \\
\text{from finite J(1) have finJ\_finite } J \text{ by (auto simp: normal\_imp\_subgroup subgroup\_finite)} \\
\text{from } JG'G' \text{ simple\_last have rcosets } G[\text{carrier := } J] \text{ G' } \subseteq \{1_{G \text{ Mod } G'}\} \lor \\
\text{rcosets } G[\text{carrier := } J] \text{ G' } = \text{carrier } (G \text{ Mod } G') \\
\text{unfolding simple\_group\_def simple\_group\_axioms\_def by auto} \\
\text{thus False} \\
\text{proof} \\
\text{assume rcosets } G[\text{carrier := } J] \text{ G' } = \{1_{G \text{ Mod } G'}\} \\
\text{hence rcosets } G[\text{carrier := } J] \text{ G' } = \{1_{G[\text{carrier := } J]} \text{ Mod } G'} \text{ unfolding FactGroup\_def by simp} \\
\text{hence } G' = J \text{ using } G'J \text{ finJ normal\_fact\_group\_trivial\_iff unfolding FactGroup\_def by fastforce} \\
\text{with J(2) show False by simp} \\
\text{next} \\
\text{assume facts\_eq:rcosets } G[\text{carrier := } J] \text{ G' } = \text{carrier } (G \text{ Mod } G') \\
\text{have } J = \text{carrier } G \\
\text{proof} \\
\text{show } J \subseteq \text{carrier } G \text{ using J(1) normal\_imp\_subgroup subgroup\_subset by fastforce} \\
\text{next} \\
\text{show carrier } G \subseteq J \\
\text{proof} \\
\text{fix } x \\
\text{assume } xx \in \text{carrier } G \\
\text{hence } G' \#> x \in \text{carrier } (G \text{ Mod } G') \text{ unfolding FactGroup\_def}
RCOSETS-def by auto  
  hence G' #> x ∈ rcosets\(_G(\text{carrier} := J)\) G' using facts-eq by auto  
  then obtain j where j: j ∈ J G' #> x = G' #> j unfolding RCOSETS-def  
r-coset-def by force  
  hence x ∈ G' #> j using G'G normal-imp-subgroup x repr-independenceD  
by fastforce  
  then obtain g' where g': g' ∈ G' x = g' ⊗ j unfolding r-coset-def  
by auto  
  hence g' ∈ J using G'J normal-imp-subgroup subgroup subset by force  
with g'(\(\underline{2}\)) j(\(\underline{1}\)) show x ∈ J using J(\(\underline{1}\)) normal-imp-subgroup subset  
by force  
with g'(\(\underline{2}\)) j(\(\underline{1}\)) show x ∈ J using J(\(\underline{3}\)) normal-imp-subgroup m-closed  
by fastforce  
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 \(xs \neq []\) have lenxs:length xs > 0 by simp  
from xs have rem:remdups-adj (x # xs) = (if x = y then remdups-adj (y # xs')  
  else x # remdups-adj (y # xs')) using remdups-adj.simps(3) by auto  
show thesis  
proof (cases x = y)  
case True  
  with rem xs have rem2:remdups-adj (x # xs) = remdups-adj xs by auto  
  with Cons(3) have i + 1 < length (remdups-adj xs) by simp  
with Cons.IH lenxs obtain k where j:k + 1 < length xs remdups-adj xs! i =  
  xs! k  
  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:remdups-adj (x # xs) = x # remdups-adj xs by auto  
show thesis

39
proof (cases i)
case 0
  have 0 + 1 < length (x # xs) using lenxs by auto
  moreover have remdups-adj (x # xs)! i = (x # xs)! 0
  proof –
    have remdups-adj (x # xs)! i = (x # remdups-adj (y # xs'))! 0 using xs
  end
  using rem2 0 by simp
  also have ... = x by simp
  also have ... = (x # xs)! 0 by simp
  finally show ?thesis.
  qed
  moreover have remdups-adj (x # xs)! (i + 1) = (x # remdups-adj (y # xs'))! 1
  proof –
    have remdups-adj (x # xs)! (i + 1) = (x # remdups-adj (y # xs'))! 1
  using xs rem2 0 by simp
  also have ... = remdups-adj (y # xs')! 0 by simp
  also have ... = (y # (remdups (y # xs')))! 0 by (metis nth-Cons'
  remdups-adj-Cons-alt)
  also have ... = y by simp
  also have ... = (x # xs)! (0 + 1) unfolding xs by simp
  finally show ?thesis.
  qed
ultimately show thesis by (rule Cons.prems(1))
next
  case (Suc k)
  with Cons(3) have k + 1 < length (remdups-adj (x # xs)) - 1 by auto
  also have ... ≤ length (remdups-adj xs) + 1 - 1 by (metis One-nat-def le_refl
  list.size(4) rem2)
  also have ... = length (remdups-adj xs) by simp
  finally have k + 1 < length (remdups-adj xs).
  with Cons.IH lenxs obtain j where j+1 < length xs remdups-adj xs ! k
  = xs ! j
    remdups-adj xs ! (k + 1) = xs ! (j + 1) by auto
  from j(1) have Suc j + 1 < length (x # xs) by simp
  moreover have remdups-adj (x # xs)! i = (x # xs)! (Suc j)
  proof –
    have remdups-adj (x # xs)! i = (x # remdups-adj xs)! i using rem2 by simp
    also have ... = (remdups-adj xs)! k using Suc by simp
    also have ... = xs ! j using j(2).
    also have ... = (x # xs)! (Suc j) by simp
    finally show ?thesis.
  qed
  moreover have remdups-adj (x # xs)! (i + 1) = (x # remdups-adj xs)! (i + 1)
  proof –
    have remdups-adj (x # xs)! (i + 1) = (x # remdups-adj xs)! (i + 1) using
    rem2 by simp
    also have ... = (remdups-adj xs)! (k + 1) using Suc by simp
    also have ... = xs ! (j + 1) using j(3).
also have \ldots = (x \# xs) ! (Suc j + 1) by simp

finally show \&thesis.
qed

ultimately show \&thesis by (rule Cons.prems(1))

qed

qed

lemma hd-remdups-adj[simp]: hd (remdups-adj xs) = hd xs

by (induction xs rule: remdups-adj.induct) simp-all

lemma remdups-adj-adjacent:
      Suc i < length (remdups-adj xs) \implies remdups-adj ! i \neq remdups-adj ! 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 | carrier := K) (remdups-adj (map (\lambda H. K \cap H) \varnothing))

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

apply (auto simp only: conjI del: equalityI)

proof -
  show group (G | 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) \varnothing) = []
  hence map ((\cap) K) \varnothing = [] by (metis remdups-adj-Nil-iff)
  hence \varnothing = [] 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 \varnothing = {1} \# tl \varnothing using notempty hd by (metis list.sel(1,3) neq-Nil-conv)
  hence map ((\cap) K) \varnothing = map ((\cap) K) ({1} \# tl \varnothing) by simp
  hence remdups-adj (map ((\cap) K) \varnothing) = remdups-adj ((K \cap {1}) \# (map ((\cap) K) (tl \varnothing))) by simp

also have \ldots = (K \cap {1}) \# tl (remdups-adj ((K \cap {1}) \# (map ((\cap) K) (tl \varnothing)))) by simp

finally have hd (remdups-adj (map ((\cap) K) \varnothing)) = K \cap {1} using list.sel(1)

by metis

thus hd (remdups-adj (map ((\cap) K) \varnothing)) = \{1\}_{G | 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 $\mathfrak{G} = (\text{carrier } G) \neq tl (\text{rev } \mathfrak{G})$ by (metis list.sel(1,3) last last-rev
neq-Nil-cone notempty rev-is-Nil-cone rev-rev-ident)

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

hence rev:rev (map ((\cap) K) $\mathfrak{G}) = (K \cap (\text{carrier } G)) \neq (map ((\cap) K) (tl (\text{rev } \mathfrak{G})))$ by (metis
s imp)

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

by (metis hd-rev map-is-Nil-conv notempty remdups-adj-Nil-iff)
also have \ldots = hd (remdups-adj (rev (map ((\cap) K) $\mathfrak{G}$))) by (metis remdups-adj-rev)
also have \ldots = hd (remdups-adj ((K \cap (\text{carrier } G)) \neq (map ((\cap) K) (tl (\text{rev } \mathfrak{G}))) by (metis list.sel(I) remdups-adj-Cons-alt)
also have \ldots = K using KG normal-imp-subgroup subgroup subset by force

finally show last (remdups-adj (map ((\cap) K) $\mathfrak{G}$)) = carrier (G[|carrier := K|])
by auto

next
— The induction step, using the second isomorphism theorem for groups.

fix $j$
assume $ji + 1 < length (\text{remdups-adj} (\text{map} ((\cap) K) \mathfrak{G}))$

have KGnotempty:(map ((\cap) K) $\mathfrak{G}) \neq []$ using notempty by (metis Nil-is-map-cone)

with $j$ obtain $i$ where $i + 1 < length (\text{map} ((\cap) K) \mathfrak{G})$

(remdups-adj (map ((\cap) K) $\mathfrak{G}$)) ! $j = (\text{map} ((\cap) K) \mathfrak{G}) ! i$

using remdups-adj-obtain-adjacency by force

from $i(I)$ have $i':i + 1 < length \mathfrak{G}$ by (metis length-map)

hence $\text{GiS}i':\mathfrak{G} : i \subseteq \mathfrak{G} : (i + I)$ 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 GSi KG i' normal-series-subgroups have GSiKnormGSi:G! (i + 1) \cap K < G[|carrier := G ! (i + 1)|]

using second-isomorphism-grp.normal-subgrp-intersection-normal

unfolding second-isomorphism-grp-def second-isomorphism-grp-axioms-def by auto

with GSi have $\mathfrak{G} ! (i + 1) \cap K < G[|carrier := G ! (i + 1)|]

by (metis group.normal-subgroup-intersect group.subgroup-imp-group i' is-group

hence $K \cap (\mathfrak{G} ! i \cap G ! (i + I)) < G[|carrier := G ! (i + I)|]$ by (metis
inf-commute inf-left-commute)

hence KGnormGSi:K \cap G ! i < G[|carrier := G ! (i + I)|] using GSi' by (metis
le-iff-inf)

moreover have $K \cap G ! i \subseteq K \cap G ! (i + I)$ using GSi' by auto

moreover have groupGSi:group (G[|carrier := G ! (i + 1)|]) using i nor-
mal-series-subgroups subgroup-imp-group by auto
  moreover have subKGSiGSi:subgroup (K ∩ G! (i + 1)) (G[carrier := G! (i + 1)]) by (metis GSiKnormGSi inf-sup-ac! normal-imp-subgroup)
  ultimately have fstgoal.K ∩ G! i ⊆ G[carrier := G! (i + 1), carrier := K ∩ G! (i + 1)]
    using group.normal-restrict-supergroup by force
  thus remdups-adj (map ((∩) K) Σ)! j < G[carrier := K, carrier := remdups-adj (map ((∩) K) Σ) ! (i + 1)]
    using i by auto
  from simplefact have Gisimple:simple-group (G[carrier := G! (i + 1)]) Mod G i i using i' by simp
    hence Gimax:max-normal-subgroup (G! i) (G[carrier := G! (i + 1)])
      using normal.max-normal-simple-quotient GSi finGSi by force
  from GSiKnormGSi GSi have G! i < G[carrier := G! (i + 1)] ! (i + 1) ∩ K < G[carrier := G! (i + 1)]
    using groupGsi group.normal-subgroup-set-mult-closed set-mult-consistent by fastforce
  hence G! i < # G! (i + 1) ∩ K < G[carrier := G! (i + 1)] unfolding
    set-mult-def by auto
  hence G! i < # K ∩ G! (i + 1) < G[carrier := G! (i + 1)] using inf-commute by metis
  moreover have G! i ≤ G! i < # G[carrier := G! (i + 1)] K ∩ G! (i + 1)
    unfolding second-isomorphism-grp-def second-isomorphism-grp-axioms-def
    using subKGSiGSi GSi normal-imp-subgroup by fastforce
  hence G! i ≤ G! i < # K ∩ G! (i + 1) unfolding set-mult-def by auto
  ultimately have KGdisj:G! i < # K ∩ G! (i + 1) = G! i ∨ G! i i < # K ∩ G! (i + 1) = G! i (i + 1)
    using Gimax unfolding max-normal-subgroup-def max-normal-subgroup-axioms-def by auto
  obtain ϕ where ϕ ∈ iso (G[carrier := K ∩ G! (i + 1)]) Mod (G! i ∩ (K ∩ G! (i + 1)))
    (G[carrier := G! i < # G[carrier := G! (i + 1)] K ∩ G! (i + 1)])
    Mod G! i i
      using second-isomorphism-grp.normal-intersection-quotient-isom
      unfolding second-isomorphism-grp-def second-isomorphism-grp-axioms-def
      using GSi subKGSiGSi Gnormal-imp-subgroup by fastforce
  hence ϕ ∈ iso (G[carrier := K ∩ G! (i + 1)]) Mod (K ∩ G! (i + 1) ∩ G! i)
    (G[carrier := G! i < # G[carrier := G! (i + 1)] K ∩ G! (i + 1)])
    Mod G! i i
      by (metis inf-commute)
  hence ϕ ∈ iso (G[carrier := K ∩ G! (i + 1)]) Mod (K ∩ (G! (i + 1) ∩ G! i))
    (G[carrier := G! i < # G[carrier := G! (i + 1)] K ∩ G! (i + 1)])
    Mod G! i i
      by (metis Int-assoc)
  hence ϕ ∈ iso (G[carrier := K ∩ G! (i + 1)]) Mod (K ∩ G! i)
    (G[carrier := G! i < # G[carrier := G! (i + 1)] K ∩ G! (i + 1)])

43
Mod $\emptyset \cap i$  

by (metis GiSi' Int-absorb2 Int-commute)  

hence $\varphi \varphi \in$ iso $(G\langle$carrier $:= K \cap \emptyset ! (i + 1)\rangle)$ Mod $(K \cap \emptyset ! i)$  

(G\langle carrier := \emptyset ! i \notin\# K \cap \emptyset ! (i + 1)\rangle) Mod \emptyset ! i  

unfolding set-mult-def by auto  

from fstgoal have KGsiKGi\$group:group $(G\langle$carrier $:= K \cap \emptyset ! (i + 1)\rangle)$ Mod $(K \cap \emptyset ! i)$ using normal_factor\$group-is-group by auto  

from KGdisj show simple\$group $(G\langle$carrier $:= K$, carrier $:= \text{remdups-adj (map (\langle i\rangle K) \emptyset) ! (j + 1)\rangle)$ Mod remdups-adj (map (\langle i\rangle K) \emptyset) ! j  

proof auto  

have groupGi:group $(G\langle$carrier $:= \emptyset ! i)\rangle$ using i' normal\$series\$sub\$groups subgroup-imp\$group by auto  

assume $\emptyset ! i \notin\# K \cap \emptyset ! \Succ i = \emptyset ! i$  

with $\varphi$ have $\varphi \in$ iso $(G\langle$carrier $:= K \cap \emptyset ! (i + 1)\rangle)$ Mod $(K \cap \emptyset ! i)$  

(G\langle carrier := \emptyset ! i \rangle Mod \emptyset ! i) by auto  

moreover obtain $\psi$ where $\psi \in$ iso $(G\langle$carrier $:= \emptyset ! i)\rangle$ Mod (carrier $(G\langle$carrier $:= \emptyset ! i)\rangle))$ (G\langle carrier := \{1\}$\langle$carrier $:= \emptyset ! i)\rangle)  

using group_self\$factor-iso groupGi by force  

ultimately obtain $\pi$ where $\pi \in$ iso $(G\langle$carrier $:= K \cap \emptyset ! (i + 1)\rangle)$ Mod $(K \cap \emptyset ! i)$  

(G\langle carrier := \{1\}\rangle)  

using iso-set-trans by fastforce  

hence order $(G\langle$carrier $:= K \cap \emptyset ! (i + 1)\rangle)$ Mod $(K \cap \emptyset ! i)$  

= order $(G\langle$carrier $:= \{1\}\rangle)$ by (metis iso-order-closed)  

hence order $(G\langle$carrier $:= K \cap \emptyset ! (i + 1)\rangle)$ Mod $(K \cap \emptyset ! i)$  

= 1 unfolding order-def by auto  

hence carrier $(G\langle$carrier $:= K \cap \emptyset ! (i + 1)\rangle)$ Mod $(K \cap \emptyset ! i)$  

= 1$G\langle$carrier $:= K \cap \emptyset ! (i + 1)\rangle) Mod (K \cap \emptyset ! i)$  

using group_order\$one\$triv\$iff KGsiKGi\$group by blast  

moreover from fstgoal have $K \cap \emptyset ! i \triangleleft G\langle$carrier $:= K \cap \emptyset ! (i + 1)\rangle)$ by auto  

moreover from finGsi have finite (carrier $(G\langle$carrier $:= K \cap \emptyset ! (i + 1)\rangle))$ by auto  

ultimately have $K \cap \emptyset ! i = \text{carrier}$ $(G\langle$carrier $:= K \cap \emptyset ! (i + 1)\rangle)$ by (metis normal\$factor\$group-trivial-iff)  

hence (remdups-adj (map (\langle i\rangle K) \emptyset) ! j = (remdups-adj (map (\langle i\rangle K) \emptyset) ! (j + 1) using i by auto  

with $j$ have False using remdups-adj\$adjacent KGnotempty Suc-eq\$plus1 by metis  

thus simple\$group $(G\langle$carrier $:= \text{remdups-adj (map (\langle i\rangle K) \emptyset) ! Suc j\rangle)$ Mod remdups-adj (map (\langle i\rangle K) \emptyset) ! j)$.  

next  

assume $\emptyset ! i \notin\# K \cap \emptyset ! \Succ i = \emptyset ! \Succ i$  

moreover with $\varphi$ have $\varphi \in$ iso $(G\langle$carrier $:= K \cap \emptyset ! (i + 1)\rangle)$ Mod $(K \cap \emptyset ! i)$ (G\langle carrier := \emptyset ! i \rangle Mod \emptyset ! i) by auto  

then obtain $\varphi'$ where $\varphi' \in$ iso $(G\langle$carrier $:= \emptyset ! (i + 1)\rangle)$ Mod $\emptyset ! i$ (G\langle carrier := K \cap \emptyset ! (i + 1)\rangle) Mod (K \cap \emptyset ! i)  

using KGsiKGi\$group group\$set\$sym by auto  

with Gsimple KGsiKGi\$group have simple\$group $(G\langle$carrier $:= K \cap \emptyset ! (i + 1)\rangle)$ Mod $(K \cap \emptyset ! i)$ by (metis simple\$group\$iso\$simple)
with \( i \) show simple-group \((G\langle \text{carrier} := \text{remdups-adj} (\map (\cap) K) \rangle \setminus \text{Suc} \ j)\) Mod \( \text{remdups-adj} (\map (\cap) K) \setminus j \) by auto
qed

qed

lemma (in group) composition-series-extend:
 assumes composition-series \((G\langle \text{carrier} := H \rangle) \setminus j \)
 assumes simple-group \((G \text{ Mod } H) H < G \)
 shows composition-series \(G (\setminus @ \text{carrier } G) \setminus j \)
 unfolding composition-series-def composition-series-axioms-def
 proof auto
 from assms(1) interpret \( \text{comp}\$: composition-series \(G\langle \text{carrier} := H \rangle \setminus j \) .
 show normal-series \(G (\setminus @ \text{carrier } G) \setminus j \) using assms(3) \( \text{comp}\$. \text{is-normal-series}
 by (metis normal-series-extend)
 fix \( i \)
 assume \( i ; i < \text{length } \setminus j \)
 show simple-group \((G\langle \text{carrier} := (\setminus @ \text{carrier } G) \setminus j ) \setminus \text{Suc} \ i \) Mod \( (\setminus @ \text{carrier } G) \setminus i \)
 proof (cases \( i = \text{length } \setminus j - 1 \))
 case True
 hence \( (\setminus @ \text{carrier } G) ! \) \( \text{Suc} \ i = \text{carrier } G \) by (metis \( i \) \( \text{diff-Suc-1} \) \( \text{lessE} \) \( \text{nth-append-length} \))
 moreover have \( (\setminus @ \text{carrier } G) ! \) \( i = \setminus j \) \( \text{by} \) (metis \( \text{butlast-snoc} \) \( i \) \( \text{nth-butlast} \))
 hence \( (\setminus @ \text{carrier } G) ! \) \( i = H \) using True \( \text{last-conv-nth comp}\$. \text{notempty} \)
 comp\$. \text{last} by auto
 ultimately show \( \text{thesis} \) using assms(2) by auto
 next
 case False
 hence \( \text{Suc} \ i < \text{length } \setminus j \) using \( i \) by auto
 hence \( (\setminus @ \text{carrier } G) ! \) \( \text{Suc} \ i = \setminus j \) \( \text{by} \) (metis \( \text{nth-append} \) \( \text{by} \) metis)
 moreover from \( i \) have \( (\setminus @ \text{carrier } G) ! \) \( i = \setminus j \) \( i \) \( \text{by} \) \( \text{nth-append} \) \( \text{by} \) \( \text{metis} \)
 ultimately show \( \text{thesis} \) using \( \text{Suc} \ i < \text{length } \setminus j \) \( \text{comp}\$. \text{simplefact} \) by auto
 qed
 qed

lemma (in composition-series) entries-mono:
 assumes \( i \leq j ; j < \text{length } \setminus G \)
 shows \( \setminus G ! \) \( i \subseteq \setminus G ! \) \( j \)
 using assms proof (induction \( j - i \) arbitrary; \( i \) \( j \))
 case 0
 hence \( i = j \) by auto
 thus \( \setminus G ! \) \( i \subseteq \setminus G ! \) \( j \) by auto
 next
 case \( \text{Suc} \ k i j \)
 hence \( i ; i + (\text{Suc} \ k) = j ; i + 1 < \text{length } \setminus G \) by auto
 hence \( i ; i + 1 \leq j \) by auto
 have \( \setminus G ! \) \( i \subseteq \setminus G ! (i + 1) \) using \( i' \) \( \text{normal normal-imp-subgroup subgroup subset} \) by force

45
moreover have \( j - (i + 1) = k j < \text{length } \mathcal{G} \) using Suc assms by auto
hence \( \mathcal{G} ! (i + 1) \subseteq \mathcal{G} ! j \) using Suc \( \text{ij} \) by auto
ultimately show \( \mathcal{G} ! i \subseteq \mathcal{G} ! j \) by simp
qed

theory GroupIsoClasses
imports
  HOL-Algebra.Coset
begin

9 Isomorphism Classes of Groups

We construct a quotient type for isomorphism classes of groups.

typedef 'a group = {G :: 'a monoid. group G}
proof
  show \( \forall a. \{\text{carrier} = \{a\}, \text{mult} = (\lambda x y. x), \text{one} = a\} \in \{G. \text{group } G\} \)
  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 ϕ ψ where ϕ ∈ iso (Rep-group G) (Rep-group H) ψ ∈ iso (Rep-group H) (Rep-group I)
    unfolding group-iso-rel-def by auto
  then obtain π where π ∈ iso (Rep-group G) (Rep-group I)
    using iso-set-trans by blast
  thus group-iso-rel G I unfolding group-iso-rel-def by auto
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 (∃ϕ. ϕ ∈ iso G H) = (group.iso-class G = 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 (∃ϕ. ϕ ∈ iso G H) = (∃ϕ. ϕ ∈ iso (monoid.truncate G) (monoid.truncate H))
    unfolding iso-def hom-def monoid.truncate-def by auto
  also have ... = group-iso-rel (Abs-group (monoid.truncate G)) (Abs-group (monoid.truncate H))
    unfolding group-iso-rel-def using groups group.Abs-group-inverse by (metis mem-Collect-eq)
  also have ... = (group.iso-class G = group.iso-class H) using group.iso-class-def
  assim group-iso-class.abs-eq-iff by metis
  finally show ?thesis.
qed

end

theory JordanHolder
imports
  CompositionSeries
  MaximalNormalSubgroups
  HOL-Library.Multiset
  GroupIsoClasses
begin
10 The Jordan-Hölder Theorem

locale jordan-holder = group
+ comp$f$: composition-series $G \triangleright$
+ comp$\Phi$: composition-series $G \vartriangleright$ for $\triangleright$ and $\vartriangleright$
+ 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 $\mathcal{S}$ = $\mathcal{H}$
  shows butlast quotients = normal-series.quotients ($G$[carrier := $\mathcal{S}$] ! (length $\mathcal{S}$ - 1 - 1))) (take (length $\mathcal{S}$ - 1) $\mathcal{S}$)
proof (rule nth-equality1)
  define $n$ where $n$ = length $\mathcal{S}$ - 1
  hence $n$ = length (take $n$ $\mathcal{S}$) $\triangleright$ 0 $\triangleright$ 0 < length $\mathcal{S}$ using assms notempty by auto
  interpret normal$\mathcal{S}$ butlast: normal-series ($G$[carrier := $\mathcal{S}$] ! (length $\mathcal{S}$ - 1))) take $n$ $\mathcal{S}$
  using normal-series-prefix-closed ($n$ > 0) ($n$ < length $\mathcal{S}$) by auto
  have length (butlast quotients) = length quotients - 1 by (metis length-butlast)
  also have ... = length $\mathcal{S}$ - 1 - 1 by (metis add-diff-cancel-right' quotients-length)
  also have ... = length (take $n$ $\mathcal{S}$) - 1 by (metis $n$ = length (take $n$ $\mathcal{S}$) $\triangleright$ n-def)
  also have ... = length normal$\mathcal{S}$ butlast.quotients by (metis normal$\mathcal{S}$ butlast.quotients-length diff-add-inverse2)
  finally show length (butlast quotients) = length normal$\mathcal{S}$ butlast.quotients .
  have $\forall$ i<length (butlast quotients). butlast quotients ! i = normal$\mathcal{S}$ butlast.quotients ! i
  proof auto
  fix i
  assume i:i < length quotients - Suc 0
  hence i':i < length $\mathcal{S}$ - 1 $\triangleright$ i < $n$ i + 1 < $n$ unfolding n-def using quotients-length by auto
  from i have butlast quotients ! i = quotients ! i by (metis One-nat-def length-butlast nth-butlast)
  also have ... = $G$[carrier := $\mathcal{S}$] ! (i + 1)] $\mathcal{S}$ ! i unfolding quotients-def
  using i'(1) by auto
  also have ... = $G$[carrier := (take $n$ $\mathcal{S}$) ! (i + 1)] $\mathcal{S}$ ! i using
  i'(2,3) nth-take by metis
  also have ... = normal$\mathcal{S}$ butlast.quotients ! i unfolding normal$\mathcal{S}$ butlast.quotients-def
  using i' by fastforce
  finally show butlast (normal-series.quotients $G$ $\mathcal{S}$) ! i = normal-series.quotients ($G$[carrier := $\mathcal{S}$] ! (length $\mathcal{S}$ - Suc 0))] (take $n$ $\mathcal{S}$) ! i by auto
  qed
  thus $\forall$. i. i < length (butlast quotients)
              $\triangleright$ butlast quotients ! i
              = normal-series.quotients ($G$[carrier := $\mathcal{S}$] ! (length $\mathcal{S}$ - 1 - 1)))
  (take (length $\mathcal{S}$ - 1) $\mathcal{S}$) ! 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-Hölder Multisets):**

** Assumes:**
- Group $G$
- Finite ($\text{carrier } G$)
- Composition series $G \triangleleft G$

** Shows:**
$$\text{mset } (\text{map group.iso-class } (\text{normal-series.quotients } G \triangleleft G)) = \text{mset } (\text{map group.iso-class } (\text{normal-series.quotients } G \triangleleft H))$$

** Using:**
- Assms

** Proof:**
  - **(induction length $G$ arbitrary; $G \triangleleft H$ G rule: full-nat-induct)**
    - **Then interpret** $\text{compG}: \text{composition-series } G \triangleleft G$ by simp
    - **From** 1 **interpret** $\text{compH}: \text{composition-series } G \triangleleft H$ by simp
    - **From** 1 **interpret** $\text{grpG}: \text{group } G$ by simp
    - **Show:** ?case
    - **Proof:**
      - **(cases length $G \leq 2)**
      - **Next**
        - **Case True**
          - **Hence** $\text{length } G = 0 \lor \text{length } G = 1 \lor \text{length } G = 2$ by arith
          - **With** $\text{compG}.\text{notempty have } \text{length } G = 1 \lor \text{length } G = 2$ by simp
          - **Thus** ?thesis
        - **Proof:**
          - **(auto simp del: mset-map)**
            - **First trivial case:** $G$ is the trivial group.
              - **Assume** $\text{length } G = \text{Suc } 0$
              - **Hence** $\text{length:} \text{length } G = 1$ by simp
              - **Hence** $\text{length } G + 1 = \text{length } G$ by auto
            - **Moreover from** $\text{length have } \text{charG}: G = [\{1_G\}]$ by (metis $\text{compG}.\text{composition-series-length-one})$
              - **Hence** $\text{carrier } G = \{1_G\}$ by (metis $\text{compG}.\text{composition-series-triv-group})$
            - **With** $\text{length charG have } G = H$ using $\text{compH}.\text{composition-series-triv-group}$
              - **By simp**
                - **Thus** ?thesis by simp
            - **Next**
              - **Second trivial case:** $G$ is simple.
                - **Assume** $\text{length } G = 2$
                - **Hence** $\text{charG} = \{1_G\}$, $\text{carrier } G$ by (metis $\text{compG}.\text{length-two-unique})$
                - **Hence** $\text{simple:} \text{simple-group } G$ by (metis $\text{compG}.\text{composition-series-simple-group})$
                  - **Hence** $H = \{1_G\}$, $\text{carrier } G$ using $\text{compH}.\text{composition-series-simple-group}$
              - **By auto**
                - **With** $\text{char have } G = H$ by simp
                - **Thus** ?thesis by simp
              - **Qed**
            - **Next**
              - **Case False**
                - **Non-trivial case:** $G$ has length at least 3.
                  - **Hence** $\text{length:} \text{length } G \geq 3$ by simp
                    - **First we show that $H$ must have a length of at least 3.
hence \( \sim \) simple-group \( G \) using \( \text{comp} \text{G}.\text{composition-series-simple-group} \) by auto

hence \( \text{裕} \neq \{ 1_G \} \) \( \text{carrier} \ G \) using \( \text{comp}\text{裕}.\text{composition-series-simple-group} \) by auto

hence length \( \text{裕} \neq 2 \) using \( \text{comp}\text{裕}.\text{length-two-unique} \) by auto

moreover from length \( \text{have} \) \( \text{carrier} \ G \neq \{ 1_G \} \) using \( \text{comp} \text{G}.\text{composition-series-length-one} \)

\( \text{comp} \text{G}.\text{composition-series-trie-group} \) by auto

hence length \( \text{裕} \neq 1 \) using \( \text{comp}\text{裕}.\text{composition-series-length-one} \) \( \text{comp}\text{裕}.\text{composition-series-trie-group} \) by auto

moreover from \( \text{comp}\text{裕}.\text{notempty have} \) length \( \text{裕} \neq 0 \) by simp

ultimately have length\( \text{裕}\text{big:length} \ \text{裕} \geq 3 \) using \( \text{comp}\text{裕}.\text{notempty} \) by arith

define \( m \) where \( m = \text{length} \ \text{裕} - 1 \)

define \( n \) where \( n = \text{length} \ \text{裕} - 1 \)

from length\( \text{裕}\text{big have} \) \( m \cdot m > 0 \) \( m < \text{length} \ \text{裕} \) \( (m - 1) \) \( 1 < \text{length} \ \text{裕} \) \( m - 1 = \text{length} \ \text{裕} - 2 \) \( m - 1 + 1 = \text{length} \ \text{裕} - 1 \)

unfolding \( m\text{-def} \) by auto

from length \( \text{have} \) \( n' ; n > 0 \) \( n < \text{length} \ \text{裕} \) \( (n - 1) \) \( 1 < \text{length} \ \text{裕} \) \( n - 1 = \text{length} \ \text{裕} - 2 \) \( n - 1 + 1 = \text{length} \ \text{裕} - 1 \)

unfolding \( n\text{-def} \) by auto

define \( \text{裕}\text{Pn where} \text{裕}\text{Pn} = G[\text{carrier :=} \text{裕} \setminus (n - 1)] \)

define \( \text{裕}\text{Pm where} \text{裕}\text{Pm} = G[\text{carrier :=} \text{裕} \setminus (m - 1)] \)

then interpret \( \text{.grp\text{裕}\text{Pm: group} \text{裕}\text{Pm}\text{unfolding} \text{裕}\text{Pm}\text{def using} n' \text{by} (\text{metis compG.normal-series-subgroups compG.subgroup-imp-group})} \)

interpret \( \text{grp\text{裕}\text{Pm: group} \text{裕}\text{Pm}\text{unfolding} \text{裕}\text{Pm}\text{def using} m' \text{compG.normal-series-subgroups} 1(2) \text{group.subgroup-imp-by force}} \)

have \( \text{finGbl:finite (carrier \text{裕}\text{Pn}) using} \langle n - 1 < \text{length} \ \text{裕} \setminus 1(3) \text{unfolding} \text{裕}\text{Pn}\text{def using} \text{compG.normal-series-subgroups \text{compG.subgroup-finite by auto}} \)

have \( \text{finHbl:finite (carrier \text{裕}\text{Pm}) using} \langle m - 1 < \text{length} \ \text{裕} \setminus 1(3) \text{unfolding} \text{裕}\text{Pm}\text{def using} \text{compG.normal-series-subgroups \text{compG.subgroup-finite by auto}} \)

have \( \text{quot-Gnotempty:compG.quotients \neq} \setminus \text{using compG.quotients-length length by auto} \)

have \( \text{quot-Gnotempty:compG.quotients \neq} \setminus \text{using compG.quotients-length length\text{裕}\text{big by auto}} \)

— Instantiate truncated composition series since they are used for both cases

interpret \( \text{裕}\text{bullast: composition-series} \text{裕}\text{Pm}\text{take} \ m \ \text{裕}\text{using} \text{compG.composition-series-prefix-closed} \ (m'1,2) \text{裕}\text{Pm-def by auto}} \)

interpret \( \text{裕}\text{bullast: composition-series} \text{裕}\text{Pn}\text{take} \ n \ \text{裕}\text{using} \text{compG.composition-series-prefix-closed} \ n'(1,2) \text{裕}\text{Pn-def by auto}} \)

have \( \text{ltaken:} n = \text{length (take} \ n \ \text{裕}) \text{using length-take} \ n'(2) \text{by auto} \)

have \( \text{ltakem:} m = \text{length (take} \ m \ \text{裕}) \text{using length-take} \ m'(2) \text{by auto} \)

show \( \)thesis

proof (cases \( \text{裕} ! \) \( (m - 1) = \text{裕} ! (n - 1) \))

— If \( \text{裕} ! (l - 1) = \text{裕} ! 1, \) everything is simple...

case True

— The last quotients of \( \text{裕} \) and \( \text{裕} \) are equal.

have \( \text{lastq: last compG.quotients = last compG.quotients} \)

proof

— from length \( \text{have} \ lg:length \ \text{裕} - 1 - 1 + 1 = \text{length} \ \text{裕} - 1 \text{by (metis Suc-diff-1 Suc-eq-plus1 n'(1) n-def) } \)}
from length big have lk: length $\mathcal{H} - 1 - 1 + 1 = length \mathcal{H} - 1$ by (metis Suc-diff-1 Suc-eq-plas $1 \langle 0 < m \rangle$ m-def)

have last comp $\mathcal{G}$.quotients = $G \ Mod \ \mathcal{G} ! (n - 1)$ using length comp $\mathcal{G}$.last-quotient unfolding n-def by auto

also have ... = $G \ Mod \ \mathcal{H} ! (m - 1)$ using True by simp

also have ... = last comp $\mathcal{G}$.quotients using length $\mathcal{H}$.big comp $\mathcal{H}$.last-quotient unfolding m-def by auto

finally show ?thesis .

qed

from letken have ind;mset (map group.iso-class $\mathcal{G}$butlast.quotients) = mset (map group.iso-class $\mathcal{H}$butlast.quotients) using 1(1) True $n'(5)$ grp$Pn$.is-group finGbl $\mathcal{G}$butlast.is-composition-series $\mathcal{H}$butlast.is-composition-series unfolding $\mathcal{G}$Pn-def $\mathcal{H}$Pm-def by metis

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

= mset (map group.iso-class (butlast comp $\mathcal{G}$.quotients @ [last comp $\mathcal{G}$.quotients])) by (simp add: quotientsnotempty)

also have ... = mset (map group.iso-class (butlast.quotients @ [last comp $\mathcal{G}$.quotients])) using comp $\mathcal{G}$.quotients-butlast length unfolding n-def $\mathcal{G}$Pn-def by auto

also have ... = mset (map group.iso-class $\mathcal{G}$butlast.quotients) @ [group.iso-class (last (comp $\mathcal{G}$.quotients))]) by auto

also have ... = mset (map group.iso-class $\mathcal{H}$butlast.quotients) + $\{ # \ group.iso-class (last (comp $\mathcal{G}$.quotients)) \}$ by auto

also have ... = mset (map group.iso-class $\mathcal{H}$butlast.quotients) + $\{ # \ group.iso-class (last (comp $\mathcal{G}$.quotients)) \}$ using ind by simp

also have ... = mset (map group.iso-class $\mathcal{H}$butlast.quotients) + $\{ # \ group.iso-class (last (comp $\mathcal{G}$.quotients)) \}$ using lasteq by simp

also have ... = mset (map group.iso-class $\mathcal{H}$butlast.quotients) @ [group.iso-class (last (comp $\mathcal{G}$.quotients))]) by auto

also have ... = mset (map group.iso-class $\mathcal{H}$butlast.quotients @ [last comp $\mathcal{G}$.quotients]) using append-butlast-last-id quotientsnotempty by simp

finally show ?thesis .

next

case False

define $\mathcal{H}PmInt\mathcal{G}Pn$ where $\mathcal{H}PmInt\mathcal{G}Pn = G\langle{\text{carrier := } \mathcal{H}}\rangle ! (m - 1) \cap \mathcal{G} ! (n - 1)\rangle$

interpret $\mathcal{G}$Pmax: max-normal-subgroup $\mathcal{G} ! (n - 1)$ $G$ unfolding n-def by (metis add-lessD1 diff-diff-add $n'(3)$ add.commute one-add-one $1(3)$ comp $\mathcal{G}$.snd-to-last-max-normal)

interpret $\mathcal{H}Pmax$: max-normal-subgroup $\mathcal{H} ! (m - 1)$ $G$ unfolding m-def by (metis add-lessD1 diff-diff-add $m'(3)$ add.commute one-add-one $1(3)$ comp $\mathcal{H}$.snd-to-last-max-normal)

have $\mathcal{H}Pmnnorm(G; \mathcal{H}) ! (m - 1) < G$ using comp $\mathcal{H}$.normal-series-snd-to-last $m'(4)$ unfolding m-def by auto
have \( \mathcal{S} P_{\text{norm}} G : \mathcal{S} ! (n - 1) \triangleleft G \) using \( \text{comp} \mathcal{S} . \text{normal-series-snd-to-last} \)
unfolding \( n \) def by auto
have \( \mathcal{S} P_{\text{min}} (\mathcal{S} P_{\text{norm}} G : \mathcal{S} ! (m - 1) \cap \mathcal{S} ! (n - 1) \triangleleft G \) using \( \mathcal{S} P_{\text{norm}} G \)
\( \mathcal{S} P_{\text{norm}} G \) by (rule \( \text{comp} \mathcal{S} . \text{normal-subgroup-intersect} \))
have \( \mathcal{S} P_{\text{norm}} G : \mathcal{S} ! (m - 1) \cap \mathcal{S} (n - 1) \triangleleft \mathcal{S} P_n \) using \( \mathcal{S} P_{\text{norm}} G \)
\( \mathcal{S} P_{\text{norm}} G \) Int-lower2 unfolding \( \mathcal{S} P_n \) def
by (metis \( \text{comp} \mathcal{S} . \text{normal-restrict-supergroup} \) \( \text{comp} \mathcal{S} . \text{normal-series-subgroups} \)
\( \text{comp} \mathcal{S} . \text{normal-subgroup-intersect} \) \( n \) (6))
then interpret \( \text{grp} \mathcal{S} P_{\text{mod}} \) \( \mathcal{S} P_{\text{min}} \mathcal{S} P_n \) : group \( \mathcal{S} P_n \) Mod \( \mathcal{S} ! (m - 1) \cap \mathcal{S} ! (n - 1) \) by (rule \( \text{normal-group} \) is-group)
have \( \mathcal{S} P_{\text{norm}} G : \mathcal{S} ! (m - 1) \cap \mathcal{S} ! (n - 1) \triangleleft \mathcal{S} P_m \) using \( \mathcal{S} P_{\text{norm}} G \)
\( \mathcal{S} P_{\text{norm}} G \) Int-lower2 Int-commute unfolding \( \mathcal{S} P_m \) def
by (metis \( \text{comp} \mathcal{S} . \text{normal-restrict-supergroup} \) \( \text{comp} \mathcal{S} . \text{normal-subgroup-intersect} \)
\( \text{comp} \mathcal{S} . \text{normal-series-subgroups} \) \( m \) (6))
then interpret \( \text{grp} \mathcal{S} P_{\text{mod}} \) \( \mathcal{S} P_{\text{min}} \mathcal{S} P_n \) : group \( \mathcal{S} P_m \) Mod \( \mathcal{S} ! (m - 1) \cap \mathcal{S} ! (n - 1) \) by (rule \( \text{normal-group} \) is-group)

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

have \( \text{not} \mathcal{S} P_{\text{min}} \mathcal{S} P_{\text{norm}} G : \neg (\mathcal{S} ! (m - 1) \subseteq \mathcal{S} ! (n - 1)) \) using \( \mathcal{S} P_{\text{max}} \)
\( \text{max-normal} \) \( \mathcal{S} P_{\text{norm}} G \) False \( \text{symmetric} \) \( \mathcal{S} P_{\text{max}} \) pro by simp
have \( \text{not} \mathcal{S} P_{\text{min}} \mathcal{S} P_{\text{norm}} G : \neg (\mathcal{S} ! (n - 1) \subseteq \mathcal{S} ! (m - 1)) \) using \( \mathcal{S} P_{\text{max}} \)
\( \text{max-normal} \) \( \mathcal{S} P_{\text{norm}} G \) False \( \mathcal{S} P_{\text{max}} \) pro by simp

— Show that \( \mathcal{S} P_{\text{mod}} \) \( \mathcal{S} ! (m - 1) \cap \mathcal{S} ! (n - 1) \) is a simple group.

have \( \mathcal{S} P_{\text{min}} \mathcal{S} P_{\text{norm}} G : \mathcal{S} ! (m - 1) \subseteq \mathcal{S} ! (n - 1) \triangleleft \mathcal{S} ! (m - 1) \) using \( \mathcal{S} P_{\text{max}} \)
\( \text{max-normal} \) \( \mathcal{S} P_{\text{norm}} G \) \( \text{normal} \) \( \mathcal{S} P_{\text{min}} \) group
\( \mathcal{S} P_{\text{norm}} G \) normal-imp-subgroup
unfolding \( \text{second-isomorphism-grp} \) \( H \)-contained-in-set-mult \( \mathcal{S} P_{\text{max}} \) is-normal
\( \mathcal{S} P_{\text{norm}} G \) normal-imp-subgroup
unfolding \( \text{second-isomorphism-grp} \) \( \text{def} \) \( \text{second-isomorphism-grp} \) \( \text{axioms} \)-def
\( \text{max-normal} \) \( \text{subgroup} \)-def by metis

have \( \mathcal{S} P_{\text{norm}} G : \mathcal{S} ! (n - 1) \subseteq \mathcal{S} ! (m - 1) \triangleleft \mathcal{S} ! (n - 1) \) using \( \mathcal{S} P_{\text{max}} \)
\( \text{max-normal} \) \( \mathcal{S} P_{\text{norm}} G \) \( \text{is-normal} \) \( \mathcal{S} P_{\text{norm}} G \) normal-imp-subgroup
unfolding \( \text{second-isomorphism-grp} \) \( \text{def} \) \( \text{second-isomorphism-grp} \) \( \text{axioms} \)-def
\( \text{max-normal} \) \( \text{subgroup} \)-def by metis

have \( \mathcal{S} ! (n - 1) \neq (\mathcal{S} ! (m - 1)) \triangleleft \mathcal{S} ! (n - 1) \) using \( \mathcal{S} P_{\text{min}} \mathcal{S} P_{\text{norm}} G \)
\( \mathcal{S} P_{\text{norm}} G \) by auto

hence \( \text{set-mult} G : (\mathcal{S} ! (m - 1)) \triangleleft \mathcal{S} ! (n - 1) = \text{carrier} G \)
using \( \mathcal{S} P_{\text{max}} \) is-normal \( \mathcal{S} P_{\text{norm}} G \) \( \mathcal{S} P_{\text{max}} \) is-normal \( \mathcal{S} P_{\text{norm}} G \) comp \( \mathcal{S} \)
\( \text{normal-subgroup-set-mult-closed} \)
\( \mathcal{S} P_{\text{norm}} G \) by auto
then obtain \( \varphi \) where \( \varphi \in \text{iso} (\mathcal{S} P_{\text{mod}} \mathcal{S} ! (m - 1) \cap \mathcal{S} ! (n - 1)) \)
\( (G[\text{carrier} := \text{carrier} G]) \) \( \text{Mod} \) \( \mathcal{S} ! (m - 1) \) using \( \text{second-isomorphism-grp} \)
\( \text{normal-intersection-quotient-isom} \) \( \mathcal{S} P_{\text{norm}} G \)
\( \mathcal{S} P_{\text{max}} \) is-normal \( \mathcal{S} P_{\text{norm}} G \) normal-imp-subgroup
unfolding \( \text{second-isomorphism-grp} \) \( \text{def} \) \( \text{second-isomorphism-grp} \) \( \text{axioms} \)-def
\( \text{max-normal} \) \( \text{subgroup} \)-def \( \mathcal{S} P_{\text{norm}} G \) def by metis
hence \( \varphi \in \text{iso} (\mathcal{S} P_{\text{mod}} \mathcal{S} ! (m - 1) \cap \mathcal{S} ! (n - 1)) \) \( (G \mathcal{S} \text{Mod} \mathcal{S} ! (m - 1)) \) by auto
then obtain \( \varphi 2 \) where \( \varphi 2 : \varphi 2 \in \text{iso} (G \mathcal{S} \text{Mod} \mathcal{S} ! (m - 1)) (\mathcal{S} P_{\text{mod}} \mathcal{S} ! (m - 1) \cap \mathcal{S} ! (n - 1)) \)
using group.iso-set-sym grp@PmMod矿山@Pn.is-group by auto

moreover have simple-group \( G[(\text{carrier} := S \cap (m - 1 + 1)] \) Mod \( S \cap (m - 1) \) using comp@.simplefact \( n'(7) \) by simp

hence simple-group \( G \text{ Mod } S \cap (m - 1) \) using comp@.last last-conv-nth comp@.notempty \( n'(7) \) by fastforce

ultimately have simple@PnModInt.simple-group \( (S \text{ Mod } S \cap (m - 1) \cap S \cap (n - 1)) \)

using simple-group.iso-simple grp@PmMod矿山@Pn.is-group by auto
interpret grp@GMod矿山Pn: group \( G \text{ Mod } S \cap (n - 1) \) by (metis矿山normalG normal.faactorgroup-is-group)

— Show analogues of the previous statements for \( S \cap (m - 1) \) instead of \( S \cap (n - 1) \).

have \( S \text{ PmSubSetmult' } S \cap (m - 1) \subseteq S \cap (n - 1) \) using second-isomorphism-grp.S-contained-in-set-mult矿山@Pmmax.is-normal矿山@PmnormG normal.imp-subgroup

unfolding second-isomorphism-grp-def second-isomorphism-grp-axioms-def max-normal-subgroup-def by metis

have \( S \text{ PmSubSetmult' } S \cap (n - 1) \subseteq S \cap (n - 1) \) using second-isomorphism-grp.H-contained-in-set-mult矿山@Pmmax.is-normal矿山@PmnormG comp矿山@normal.subgroup-set-mult-closed

from set-multG: obtain \( \psi \) where

\( \psi \in \text{iso } (S \text{ Pm Mod } (S \cap (n - 1) \cap S \cap (m - 1))) \)

using second-isomorphism-grp.normal-intersection-quotient-isom矿山@Pmmax.is-normal.imp-subgroup

unfolding second-isomorphism-grp-def second-isomorphism-grp-axioms-def max-normal-subgroup-def by metis

hence \( \psi: \psi \in \text{iso } (S \text{ Pm Mod } (S \cap (m - 1) \cap (S \cap (n - 1)))) \) using Int-commute by metis

then obtain \( \psi \) where

\( \psi: \psi \in \text{iso } (G \text{ Mod } (S \cap (n - 1))) \) using group.iso-set-sym grp@PmMod矿山@Pn.is-group by auto

moreover have simple-group \( G[(\text{carrier} := S \cap (n - 1 + 1)] \) Mod \( S \cap (n - 1) \) using comp@.simplefact \( n'(7) \) by simp

hence simple-group \( G \text{ Mod } S \cap (n - 1) \) using comp@.last last-conv-nth comp@.notempty \( n'(7) \) by fastforce

ultimately have simple@PmModInt.simple-group \( (S \text{ Pm Mod } S \cap (n - 1) \cap S \cap (n - 1)) \)

using simple-group.iso-simple grp@PmMod矿山@Pn.is-group by auto
interpret grp@GMod矿山Pn: group \( G \text{ Mod } S \cap (n - 1) \) by (metis矿山normalG normal.faactorgroup-is-group)
normal.factorgroup-is-group

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

\[ \text{define } \mathcal{R} \text{ where } \mathcal{R} = \text{remdups-adj (map } (\cap) (\mathcal{S} \setminus (m - 1))) \mathcal{G} \]
\[ \text{define } \mathcal{L} \text{ where } \mathcal{L} = \text{remdups-adj (map } (\cap) (\mathcal{S} \setminus (n - 1))) \mathcal{H} \]
\[ \text{interpret } \mathcal{R} \text{: composition-series } \mathcal{H} \mathcal{P} \mathcal{m} \text{ using } \text{compG.intersect-normal } 1(3) \]
\[ \text{\textit{\theta}} \mathcal{P} \mathcal{n} \text{\textit{\theta}} \text{normG unfolding } \mathcal{R} \text{-def } \mathcal{H} \mathcal{P} \mathcal{m} \text{-def by auto} \]
\[ \text{interpret } \mathcal{L} \text{: composition-series } \mathcal{G} \mathcal{P} \mathcal{n} \mathcal{L} \text{ using } \text{compH.intersect-normal } 1(3) \]
\[ \mathcal{G} \mathcal{P} \text{normG unfolding } \mathcal{L} \text{-def } \mathcal{G} \mathcal{P} \text{n-def by auto} \]

— Apply the induction hypothesis on \( \mathcal{G} \text{butlast} \) and \( \mathcal{L} \)

\[ \text{from } n'(\mathcal{L}) \text{ have } \text{Suc (length (take n } \mathcal{G})) \leq \text{length } \mathcal{G} \text{ by auto} \]
\[ \text{hence } \text{multisets}\mathcal{G} \text{butlast}\mathcal{L} \text{:mset (map group.iso-class } \mathcal{G} \text{butlast.quotients) = mset (map group.iso-class } \mathcal{L} \text{.quotients)} \]
\[ \text{using } \text{1.hyps grp}\mathcal{G} \text{.is-group finGbl } \mathcal{G} \text{butlast.is-composition-series } \mathcal{L} \text{.is-composition-series by metis} \]
\[ \text{hence } \text{length}\mathcal{L} \cdot n = \text{length } \mathcal{L} \text{ using } \mathcal{G} \text{butlast.quotients-length } \mathcal{L} \text{.quotients-length} \]
\[ \text{length-map size-mset taken by metis} \]
\[ \text{hence } \text{length}\mathcal{L} \cdot i \cdot \text{length } \mathcal{L} > 1 \text{ length } \mathcal{L} - 1 > 0 \text{ length } \mathcal{L} - 1 \leq \text{length } \mathcal{L} \text{ using } \]
\[ n'(\mathcal{L}) \text{ length by auto} \]
\[ \text{have } \text{Integ}\mathcal{L}\text{ndlast} \mathcal{H}! (m - 1) \cap \mathcal{G}! (n - 1) = \mathcal{L}! (\text{length } \mathcal{L} - 1 - 1) \]
\[ \text{proof} - \]
\[ \text{have } \text{length } \mathcal{L} - 1 - 1 + 1 < \text{length } \mathcal{L} \text{ using } \text{length}\mathcal{L} \cdot i \text{ by auto} \]
\[ \text{moreover have } \text{Knotempty}(\text{map } (((\cap) (\mathcal{S} \setminus (n - 1))) \mathcal{H})) \neq [] \text{ using } \text{compH}.\text{notempty by (metis Nil-is-map-cone)} \]
\[ \text{ultimately obtain } i \text{ where } i \cdot i + 1 < \text{length (map } ((\cap) (\mathcal{S} \setminus (n - 1))) \mathcal{H}) \]
\[ \mathcal{L}! (\text{length } \mathcal{L} - 1 - 1) = (\text{map } (((\cap) (\mathcal{S} \setminus (n - 1))) \mathcal{H}) ! i \mathcal{L}! (\text{length } \mathcal{L} - 1 - 1 + 1) = \mathcal{H}! (i + 1) \cap \mathcal{G}! (n - 1) \text{ by auto} \]
\[ \text{by (metis Suc-diff-1 Suc-neq-plas1)} \]
\[ \text{hence } \mathcal{G} \text{Pnsub} \mathcal{H} \mathcal{P} \mathcal{m}: \mathcal{G}! (n - 1) \subseteq \mathcal{H}! (i + 1) \text{ using } \text{last } \mathcal{L} \text{.notempty} \]
\[ \text{last-conv-nth unfolding } \mathcal{G} \mathcal{P} \text{n-def by auto} \]
\[ \text{from } i(\mathcal{L}) \text{ have } i + 1 < m + 1 \text{ unfolding } m \text{-def by auto} \]
\[ \text{moreover have } \neg (i + 1 \leq m - 1) \text{ using } \text{compH}.\text{entries-mono } m'(\mathcal{L}) \]
\[ \text{not}\mathcal{G} \mathcal{P} \text{nsub} \mathcal{H} \mathcal{P} \mathcal{m} \text{ by fastforce} \]
\[ \text{ultimately have } m - 1 = i \text{ by auto} \]
\[ \text{with } i \text{ show } \? \text{thesis by auto} \]
\[ \text{qed} \]
\[ \text{hence } \text{sndlast} \mathcal{H} \mathcal{P} \mathcal{m} \text{Int}\mathcal{G} \mathcal{P} \mathcal{n} = (\mathcal{G} \mathcal{P} \text{n(carrier := } \mathcal{L}! (\text{length } \mathcal{L} - 1 - 1))) \]
\[ \text{unfolding } \text{\textit{\theta}} \mathcal{P} \mathcal{m} \text{Int}\mathcal{G} \mathcal{P} \text{n-def by auto} \]
\[ \text{then interpret } \text{\textit{\theta}} \mathcal{P} \mathcal{m} \text{Int}: \text{composition-series } \mathcal{H} \mathcal{P} \mathcal{m} \mathcal{G} \text{take (length } \mathcal{L} - 1) \mathcal{L} \text{ using } \text{length}\mathcal{L} \cdot \mathcal{L} \text{.composition-series-prefix-closed by metis} \]
\[ \text{from } \text{length } \mathcal{L} > i \text{ have } \text{quot}\text{\textit{\theta}} \text{notempty}\mathcal{L}.\text{quotients} \neq [] \text{ unfolding } \mathcal{L}.\text{quotients-def by auto} \]
— Apply the induction hypothesis on $\mathcal{L}$butlast and $\mathcal{R}$butlast

**have** length $\mathcal{R} > 1$

**proof (rule contr)**

**assume** $\neg$ length $\mathcal{R} > 1$

**with** $\mathcal{R}$notempty **have** length $\mathcal{R} = 1$ by (metis One-nat-def Suc-lessI length-greater-0-cons)

**hence** carrier $\mathcal{H}Pm = \{1\}_{\mathcal{P}m}$ using $\mathcal{R}$composition-series-length-one

**hence** carrier $\mathcal{H}Pm = \{1\}_{\mathcal{G}}$ unfolding $\mathcal{H}Pm$-def by auto

**hence** carrier $\mathcal{H}Pm \subseteq \mathcal{G}$ ! (n − 1) using $\mathcal{G}Pn$max.is-subgroup subgroup one-closed by auto

**with** $\not\exists\mathcal{H}PmSub\mathcal{G}Pn$ show False unfolding $\mathcal{H}Pm$-def by auto

**qed**

**ultimately obtain i where i : i + 1 < length (map ((\(i\)) (\(\mathcal{H}\)) ! (m − 1))) \(\mathcal{G}\))

\[ \mathcal{R} ! (\text{length } \mathcal{R} − 1 − 1) = (\text{map } ((\(i\)) (\(\mathcal{H}\)) ! (m − 1))) \(\mathcal{G}\) ! i \(\mathcal{R} ! (\text{length } \mathcal{R} − 1 − 1 + 1) = (\text{map } ((\(i\)) (\(\mathcal{H}\)) ! (m − 1))) \(\mathcal{G}\) ! (i + 1)) using \(\mathcal{G}\)def,notempty by (metis Nil-is-map-cone)

**hence** $\mathcal{R} ! (\text{length } \mathcal{R} − 1 − 1) = \mathcal{G} ! i \cap \mathcal{H} ! (m − 1) \mathcal{R} ! (\text{length } \mathcal{R} − 1 − 1 + 1) = (\text{map } ((\(i\)) (\(\mathcal{H}\)) ! (m − 1))) \(\mathcal{G}\) ! (i + 1) using length$\(\mathcal{R}(!\)

**by** (metis Suc-diff-1 Suc-eq-plusI)

**hence** $\mathcal{H}Pn$sub$\mathcal{G}Pn$ ! (m − 1) $\subseteq$ \(\mathcal{G}\) ! (i + 1) using $\mathcal{R}$last $\mathcal{R}$notempty last-conv-nth unfolding $\mathcal{H}Pm$-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 $\mathcal{G}$.entries-mono n'(2)**

**not$\exists\mathcal{H}Pn$sub$\mathcal{G}Pn$ $\mathcal{H}Pn$sub$\mathcal{G}Pn$ by fastforce**

**ultimately have n = i by auto**

**with** i show ?thesis by auto

**qed**

**have** composition-series (G[[carrier := \(\mathcal{R} ! (\text{length } \mathcal{R} − 1 − 1))]] (take (length \(\mathcal{R} − 1)) \(\mathcal{R}\))

**using** length$\mathcal{R}$ ! $\mathcal{R}$composition-series-prefix-closed unfolding $\mathcal{H}Pm$Int$\mathcal{G}Pn$-def $\mathcal{H}Pm$-def by fastforce

**then interpret $\mathcal{R}$butlast: composition-series $\mathcal{H}Pm$Int$\mathcal{G}Pn$ (take (length \(\mathcal{R} − 1)) \(\mathcal{R}\)) using Inteq$\mathcal{R}$ndlast unfolding $\mathcal{H}Pm$Int$\mathcal{G}Pn$-def by auto**

**from finGb ! have finInt:finite (carrier $\mathcal{H}Pm$Int$\mathcal{G}Pn$) unfolding $\mathcal{H}Pm$Int$\mathcal{G}Pn$-def $\mathcal{G}Pn$-def by simp**

**moreover have Suc (length (take (length \(\mathcal{L} − 1)) \(\mathcal{L}\)) $\leq$ length $\mathcal{G}$ using length$\mathcal{L}$ unfolding n-def using n'(2) by auto**

**ultimately have multisets$\mathcal{L}$butlast:mset (map group.iso-class $\mathcal{L}$butlast.quotients) = mset (map group.iso-class $\mathcal{R}$butlast.quotients)**

**using 1.hyps $\mathcal{L}$butlast.is-group $\mathcal{R}$butlast.is-composition-series $\mathcal{L}$butlast.is-composition-series**
by auto

hence \( \text{length} \ (\text{take} \ (\text{length} \ R \ - \ 1) \ R) = \text{length} \ (\text{take} \ (\text{length} \ S \ - \ 1) \ S) \)

using \( \text{butlast}.\text{quotients-length} \ \text{butlast}.\text{quotients-length} \ \text{length-map} \ \text{size-mset} \)
by metis

hence \( \text{length} \ (\text{take} \ (\text{length} \ R \ - \ 1) \ R) = n - 1 \) using \( \text{length}S \ n'(1) \) by auto

hence \( \text{length}R.\text{length} R = n \) by (metis Suc-diff-1 R.notempty butlast-cone-Take length-butlast length-greater-0-conv n (1))

— Apply the induction hypothesis on \( R \) and \( \text{butlast} \)
from \( \text{Inteq}\text{sndlast} \) have \( \text{sndlast}Pn Inteq \) \( Pn = \{ Pn \{ \text{carrier} := R \ \text{length} \ R \ - \ 1 \ - \ 1 \} \} \) unfolding \( \text{Inteq} \text{Pm-def} \) \( \text{Pm-def} \) \( R \) \( \text{def} \) by auto

from \( \text{length}R \) have \( \text{Suc} \ (\text{length} \ R) \leq \text{length} \ S \) using \( n'2 \) by auto

hence \( \text{multisets}\text{butlast}R.\text{mset} \) (map group.iso-class \( \text{butlast}.\text{quotients} \)) = mset (map group.iso-class \( R.\text{quotients} \))
using 1.hyps grp\( Pm.\text{is-group} \) finHbl \( \text{butlast}.\text{is-composition-series} \ R.\text{is-composition-series} \)
by metis

hence \( \text{quot}R.\text{notempty} \ R.\text{quotients} \neq [] \) unfolding \( \text{R.quotients-def} \) by auto

interpret \( \text{butlast}R.\text{add}Pn \) : \( \text{composition-series} \ S \ Pn \) (take (length \ R \ - \ 1) \ R)
\( @ [S \ (n - 1)] \)

using grp\( Pn.\text{composition-series-extend} \) \( \text{butlast}.\text{is-composition-series} \) \( \text{simple}S \ Pn \text{ModInt} \text{Intnorm}Pn \)
unfolding \( \text{SPn-def} \) \( \text{Pm-def} \) \( \text{SPn-def} \) by auto

interpret \( \text{butlast}R.\text{add}Pn \) : \( \text{composition-series} \ S \ Pn \) (take (length \ S \ - \ 1) \ S)
\( @ [S \ (m - 1)] \)

using grp\( Pm.\text{composition-series-extend} \) \( \text{butlast}.\text{is-composition-series} \) \( \text{simple}S \ Pm \text{ModInt} \text{Intnorm}Pm \)
unfolding \( \text{SPm-def} \) \( \text{Pm-def} \) \( \text{SPm-def} \) by auto

— Prove equality of those composition series.

have mset (map group.iso-class \( \text{compS}.\text{quotients} \)) = mset (map group.iso-class (butlast \( S.\text{quotients} @ \{ \text{last} \ \text{compS}.\text{quotients} \} \)) unfolding \( \text{quotS-notempty} \) by simp
also have \( \ldots = \) mset (map group.iso-class (butlast \( S.\text{quotients} @ \{ G \text{ Mod} S \ ! (n - 1) \} \))
using \( \text{compS}.\text{quotients-butlast} \) \( \text{compS}.\text{last-quotient length} \) unfolding \( n \) \( \text{def} \)
\( \text{SPn-def} \) by auto
also have \( \ldots = \) mset (map group.iso-class (butlast \( S.\text{quotients} @ \{ \text{last} \ \text{L.quotients} \} \) + \{ \text{#} \text{ group.iso-class} \ (G \text{ Mod} S ! (n - 1)) \} \)
using \( \text{L.quotients-butlast} \) \( \text{L.last-quotient length} \) \( L > 1 \) \( \text{sndlast} \) \( \text{Inteq}\text{sndlast} \)
unfolding \( n \) \( \text{def} \) by auto
also have \( \ldots = \) mset (map group.iso-class \( R.\text{butlast}.\text{quotients} \) + \{ \text{#} \text{ group.iso-class} \)
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 ≅ length

proof –
  have length ≅ length using multisets R butlast by simp
  also have … = mset (map group.iso-class (butlast R.quotients) @ [last R.quotients]) + { # group.iso-class (G Mod (m - I)) #} by simp
  using R.quotients-butlast R.last-quotient length R > 1 Rsndlast InteqRsndlast
  unfolding m-def by auto
  also have … = mset (map group.iso-class (butlast R.quotients) @ [last R.quotients]) by simp
  using multisets R butlast R quotes R notempty by simp
  also have … = mset (map group.iso-class ((butlast R.quotients) @ [last R.quotients])) by simp
  using comp R.quotients-butlast comp R.last-quotient length R big unfolding m-def R P m-def by auto
  also have … = mset (map group.iso-class comp R.quotients) using quotes R not empty 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 ≅ length
References
