

Kneser's Theorem and the Cauchy–Davenport Theorem

Mantas Bakšys and Angeliki Koutsoukou-Argyraiki
University of Cambridge
{mb2412, ak2110}@cam.ac.uk

March 24, 2023

Abstract

We formalise Kneser's Theorem in combinatorics [2, 3] following the proof from the 2014 paper "A short proof of Kneser's addition theorem for abelian groups" by Matt DeVos [1]. We also show a strict version of Kneser's Theorem as well as the Cauchy–Davenport Theorem as a corollary of Kneser's Theorem.

Contents

1 Preliminaries	3
1.1 Elementary lemmas on sumsets	3
1.2 Stabilizer and basic properties	7
1.3 Convergent	19
1.4 Technical lemmas from DeVos’s proof of Kneser’s Theorem .	19
1.5 A function that picks coset representatives randomly	26
1.6 Useful group-theoretic results	33
2 Kneser’s Theorem and the CauchyDavenport Theorem: main proofs	35
2.1 Proof of Kneser’s Theorem	35
2.2 Strict version of Kneser’s Theorem	56
2.3 The CauchyDavenport Theorem	57

Acknowledgements

Angeliki Koutsoukou-Argyraki is funded by the ERC Advanced Grant ALEXANDRIA (Project GA 742178) funded by the European Research Council and led by Lawrence C. Paulson (University of Cambridge, Department of Computer Science and Technology). Mantas Bakšys received funding for his internship supervised by Koutsoukou-Argyraki by the Cambridge Mathematics Placements (CMP) Programme and by the ALEXANDRIA Project. We wish to thank Manuel Eberl for his useful suggestion for treating induction when there is a type discrepancy between the induction hypothesis and the induction step.

1 Preliminaries

theory *Kneser-Cauchy-Davenport-preliminaries*

imports

Complex-Main

Pluenecke-Ruzsa-Inequality.Pluenecke-Ruzsa-Inequality

HOL-Number-Theory.Prime-Powers

begin

context *subgroup-of-group*

begin

interpretation *left: left-translations-of-group ..*

interpretation *right: right-translations-of-group ..*

interpretation *transformation-group left.translation ' H G ..*

lemma *Right-Coset-eq-iff:*

assumes $x \in G$ **and** $y \in G$

shows $H \cdot x = (H \cdot y) \iff H \cdot x \cap (H \cdot y) \neq \{\}$

using *assms Right-Coset-is-orbit*

by (*metis Int-absorb orbit.disjoint orbit.natural.map-closed orbit.non-vacuous*)

end

context *additive-abelian-group*

begin

1.1 Elementary lemmas on sumsets

lemma *sumset-translate-eq-right:*

assumes $A \subseteq G$ **and** $B \subseteq G$ **and** $x \in G$

shows $(\text{sumset } A \{x\} = \text{sumset } B \{x\}) \iff A = B$ **using** *assms*

by (*smt (verit, best) Diff-Int-distrib2 Diff-eq-empty-iff*

Int-Un-eq(1) Int-absorb2 Un-Diff-cancel2 Un-commute insert-disjoint(2)

subset-refl sumset-is-empty-iff sumsetdiff-sing)

lemma *sumset-translate-eq-left:*

assumes $A \subseteq G$ **and** $B \subseteq G$ **and** $x \in G$

shows $(\text{sumset } \{x\} A = \text{sumset } \{x\} B) \iff A = B$ **using** *assms*

by (*simp add: sumset-commute sumset-translate-eq-right*)

lemma *differenceset-translate-eq-right:*

assumes $A \subseteq G$ **and** $B \subseteq G$ **and** $x \in G$

shows $(\text{differenceset } A \{x\} = \text{differenceset } B \{x\}) \iff A = B$ **using** *assms*

by (*metis Int-absorb2 differenceset-commute minus-minusset minusset-subset-carrier*
sumset-translate-eq-left)

lemma *differenceset-translate-eq-left*:
assumes $A \subseteq G$ **and** $B \subseteq G$ **and** $x \in G$
shows $(\text{differenceset } \{x\} A = \text{differenceset } \{x\} B) \longleftrightarrow A = B$ **using** *assms*
by (*metis differenceset-commute differenceset-translate-eq-right*)

lemma *sumset-inter-union-subset*:
 $\text{sumset } (A \cap B) (A \cup B) \subseteq \text{sumset } A B$
by (*metis Int-Diff-Un Int-Un-eq(2) Un-subset-iff sumset-commute sumset-subset-Un(2)*
sumset-subset-Un2)

lemma *differenceset-group-eq*:
 $G = \text{differenceset } G G$
using *equalityE minusset-eq minusset-triv subset-antisym sumset-D(1) sumset-subset-carrier*
sumset-mono image-mono Int-absorb **by** *metis*

lemma *card-sumset-singleton-subset-eq*:
assumes $a \in G$ **and** $A \subseteq G$
shows $\text{card } (\text{sumset } \{a\} A) = \text{card } A$
using *assms card-sumset-singleton-eq card.infinite card-sumset-0-iff' le-iff-inf*
sumset-commute
by *metis*

lemma *card-differenceset-singleton-mem-eq*:
assumes $a \in G$ **and** $A \subseteq G$
shows $\text{card } A = \text{card } (\text{differenceset } A \{a\})$
using *assms* **by** (*metis card-minusset' card-sumset-singleton-subset-eq difference-*
set-commute
minusset-subset-carrier)

lemma *card-singleton-differenceset-eq*:
assumes $a \in G$ **and** $A \subseteq G$
shows $\text{card } A = \text{card } (\text{differenceset } \{a\} A)$
using *assms* **by** (*metis card-minusset' card-sumset-singleton-subset-eq minus-*
set-subset-carrier)

lemma *sumset-eq-Union-left*:
assumes $A \subseteq G$
shows $\text{sumset } A B = (\bigcup a \in A. \text{sumset } \{a\} B)$
proof
show $\text{sumset } A B \subseteq (\bigcup a \in A. \text{sumset } \{a\} B)$
using *assms sumset.cases Int-absorb2 Int-iff UN-iff singletonI sumset.sumsetI*
by (*smt (verit, del-insts) subsetI*)
next

show $(\bigcup a \in A. \text{sumset } \{a\} B) \subseteq \text{sumset } A B$
using *sumset* **by** *auto*
qed

lemma *sumset-eq-Union-right*:
assumes $B \subseteq G$
shows $\text{sumset } A B = (\bigcup b \in B. \text{sumset } A \{b\})$
using *assms sumset-commute sumset-eq-Union-left* **by** (*metis (no-types, lifting) Sup.SUP-cong*)

lemma *sumset-singletons-eq*:
assumes $a \in G$ **and** $b \in G$
shows $\text{sumset } \{a\} \{b\} = \{a \oplus b\}$
using *assms sumset.simps subset-antisym* **by** *auto*

lemma *sumset-eq-subset-diffenceset*:
assumes $K \subseteq G$ **and** $K \neq \{\}$ **and** $A \subseteq G$ **and** $\text{sumset } A K = \text{sumset } B K$
shows $A \subseteq \text{diffenceset } (\text{sumset } B K) K$
proof
fix a **assume** $ha: a \in A$
obtain k **where** $hk: k \in K$ **using** *assms(2)* **by** *blast*
then have $a \oplus k \in \text{sumset } B K$ **using** *assms sumset.sumsetI ha* **by** *blast*
then have $a \oplus (k \ominus k) \in \text{diffenceset } (\text{sumset } B K) K$ **using** hk *assms ha minusset.minussetI*
subset-iff sumset.sumsetI **by** (*smt (verit) associative composition-closed inverse-closed*)
then show $a \in \text{diffenceset } (\text{sumset } B K) K$ **using** hk ha *subsetD assms right-unit*
by (*metis invertible invertible-right-inverse*)
qed

end

locale *subgroup-of-additive-abelian-group* =
subgroup-of-abelian-group $H G (\oplus) \mathbf{0} + \text{additive-abelian-group } G (\oplus) \mathbf{0}$
for $H G$ **and** *addition (infixl \oplus 65)* **and** *zero (0)*

begin

notation *Left-Coset* (*infixl* $\cdot |$ 70)

lemma *Left-Coset-eq-sumset*:
assumes $x \in G$
shows $\text{sumset } \{x\} H = x \cdot | H$
using *assms Left-Coset-memI sumset.simps* **by** *fastforce*

lemma *sumset-subgroup-eq-iff*:
assumes $a \in G$ **and** $b \in G$

shows $\text{sumset } \{a\} H = \text{sumset } \{b\} H \longleftrightarrow$
 $(\text{sumset } \{a\} H) \cap (\text{sumset } \{b\} H) \neq \{\}$
using *Right-Coset-eq-iff* *assms* *Left-Coset-eq-sumset* *Left-equals-Right-coset* **by**
presburger

lemma *card-divide-sumset*:
assumes $A \subseteq G$
shows $\text{card } H \text{ dvd } \text{card } (\text{sumset } A H)$
proof(*cases* $\text{finite } H \wedge \text{finite } A$)
case *hfin*: *True*
then have *hfinsum*: $\bigwedge X. X \in ((\lambda a. \text{sumset } \{a\} H) ' A) \implies \text{finite } X$
using *finite-sumset* **by** *force*
moreover have *pairwise disjoint* $((\lambda a. \text{sumset } \{a\} H) ' A)$
using *pairwise-imageI disjoint-def* *sumset-subgroup-eq-iff* *subset-eq* *assms* **by** (*smt*
(verit, best))
moreover have $\text{card } H \text{ dvd } \text{sum } \text{card } ((\lambda a. \text{sumset } \{a\} H) ' A)$
proof(*intro* *dvd-sum*)
fix X **assume** $X \in (\lambda a. \text{sumset } \{a\} H) ' A$
then show $\text{card } H \text{ dvd } \text{card } X$ **using** *dvd-refl*
using *Left-equals-Right-coset* *Right-Coset-cardinality* *assms* *Left-Coset-eq-sumset*
by *auto*
qed
ultimately show *?thesis* **using** *assms* *sumset-eq-Union-left* *card-Union-disjoint*
by *metis*
next
case *False*
then show *?thesis* **using** *assms* *card-sumset-0-iff* **by** (*metis* *card-eq-0-iff* *dvd-0-right*
sub *subsetI*)
qed

lemma *sumset-subgroup-eq-Class-Union*:
assumes $A \subseteq G$
shows $\text{sumset } A H = (\bigcup (\text{Class } ' A))$
proof
show $\text{sumset } A H \subseteq \bigcup (\text{Class } ' A)$
proof
fix x **assume** $x \in \text{sumset } A H$
then obtain $a b$ **where** $ha: a \in A$ **and** $b \in H$ **and** $x = a \oplus b$
using *sumset.cases* **by** *blast*
then have $x \in \text{Class } a$ **using** *Left-Coset-Class-unit* *Left-Coset-eq-sumset* *assms*
by *blast*
thus $x \in \bigcup (\text{Class } ' A)$ **using** *ha* **by** *blast*
qed
next
show $\bigcup (\text{Class } ' A) \subseteq \text{sumset } A H$
proof(*intro* *Union-least*)
fix X **assume** $X \in \text{Class } ' A$
then obtain a **where** $a \in A$ **and** $X = \text{Class } a$ **by** *blast*
moreover hence $\{a\} \subseteq A$ **by** *auto*

ultimately show $X \subseteq \text{sumset } A \ H$ **using** *Left-Coset-Class-unit*
Left-Coset-eq-sumset assms sumset-mono subset-refl in-mono **by** *metis*
qed
qed

lemma *Class-image-sumset-subgroup-eq*:
assumes $A \subseteq G$
shows $\text{Class } ' (\text{sumset } A \ H) = \text{Class } ' A$
proof
show $\text{Class } ' \text{sumset } A \ H \subseteq \text{Class } ' A$
proof
fix x **assume** $x \in \text{Class } ' \text{sumset } A \ H$
then obtain c **where** $hc: c \in \text{sumset } A \ H$ **and** $x = \text{Class } c$ **by** *blast*
moreover obtain $a \ b$ **where** $ha: a \in A$ **and** $b \in H$ **and** $c = a \oplus b$ **using** hc
sumset.cases
by *blast*
ultimately show $x \in \text{Class } ' A$ **using** ha *Class-eq CongruenceI assms composition-closed*
sumset.cases Partition-def commutative image-eqI left-unit sub unit-closed
by (*smt (verit, ccfv-threshold) Block-self Class-cong Normal-def*)
qed
next
show $\text{Class } ' A \subseteq \text{Class } ' \text{sumset } A \ H$ **using** *assms right-unit subsetD subsetI sumset.sumsetI*
unit-closed **by** (*smt (verit, del-insts) image-subset-iff sub-unit-closed*)
qed

lemma *Class-cover-imp-subset-or-disj*:
assumes $A = \bigcup (\text{Class } ' C)$ **and** $x \in G$ **and** $C \subseteq G$
shows $\text{Class } x \subseteq A \vee \text{Class } x \cap A = \{ \}$
proof(*intro disjCI*)
assume $\text{Class } x \cap A \neq \{ \}$
then obtain c **where** $c \in C$ **and** $\text{Class } x \cap \text{Class } c \neq \{ \}$ **using** *assms* **by** *blast*
then show $\text{Class } x \subseteq A$ **using** *assms not-disjoint-implies-equal Sup-upper imageI subset-iff*
by *blast*
qed

end

context *additive-abelian-group*

begin

1.2 Stabilizer and basic properties

We define the stabilizer or group of periods of a nonempty subset of an abelian group.

definition *stabilizer::'a set \Rightarrow 'a set* **where**

```

stabilizer S ≡ {x ∈ G. sumset {x} (S ∩ G) = S ∩ G}

lemma stabilizer-is-subgroup: fixes S :: 'a set
  shows subgroup (stabilizer S) G (⊕) (0)
proof (intro subgroupI)
  show stabilizer S ⊆ G using stabilizer-def by auto
next
  show 0 ∈ stabilizer S using stabilizer-def by (simp add: Int-absorb1 Int-commute)
next
  fix a b assume haS: a ∈ stabilizer S and hbS: b ∈ stabilizer S
  then have haG: a ∈ G and hbG: b ∈ G using stabilizer-def by auto
  have sumset {a ⊕ b} (S ∩ G) = sumset {a} (sumset {b} (S ∩ G))
  proof
    show sumset {a ⊕ b} (S ∩ G) ⊆ sumset {a} (sumset {b} (S ∩ G)) using haG
  hbG
    empty-subsetI insert-subset subsetI sumset.simps sumset-assoc sumset-mono
    by metis
    show sumset {a} (sumset {b} (S ∩ G)) ⊆ sumset {a ⊕ b} (S ∩ G)
      using empty-iff insert-iff sumset.simps sumset-assoc by (smt (verit, best)
subsetI)
  qed
  then show a ⊕ b ∈ stabilizer S using haS hbS stabilizer-def by auto
next
  fix g assume g ∈ stabilizer S
  thus invertible g using stabilizer-def by auto
next
  fix g assume hgS: g ∈ stabilizer S
  then have hinvsom : inverse g ⊕ g = 0 using stabilizer-def by simp
  have sumset {inverse g} (sumset {g} (S ∩ G)) = (S ∩ G)
  proof
    show sumset {inverse g} (sumset {g} (S ∩ G)) ⊆ (S ∩ G) using
      empty-iff insert-iff sumset.simps sumset-assoc subsetI left-unit hinvsom
      by (smt (verit, ccfv-threshold))
    show (S ∩ G) ⊆ sumset {inverse g} (sumset {g} (S ∩ G))
    proof
      fix s assume hs: s ∈ (S ∩ G)
      then have inverse g ⊕ g ⊕ s ∈ sumset {inverse g} (sumset {g} (S ∩ G))
using
  hgS stabilizer-def additive-abelian-group.sumset.sumsetI
  additive-abelian-group-axioms associative in-mono inverse-closed mem-Collect-eq
singletonI
  by (smt (z3) IntD2)
  thus s ∈ sumset {inverse g} (sumset {g} (S ∩ G)) using hinvsom hs
  by simp
  qed
  qed
  thus inverse g ∈ stabilizer S using hgS stabilizer-def by auto
qed

```


interpretation *subgroup-of-additive-abelian-group stabilizer A G* (\oplus) $\mathbf{0}$
using *stabilizer-is-subgroup subgroup-of-abelian-group-def*
by (*metis abelian-group-axioms additive-abelian-group-axioms group-axioms*
subgroup-of-additive-abelian-group-def subgroup-of-group-def)

lemma *zero-mem-stabilizer*: $\mathbf{0} \in \text{stabilizer } A$..

lemma *stabilizer-is-nonempty*:
shows *stabilizer* $S \neq \{\}$
using *sub-unit-closed* **by** *blast*

lemma *Left-Coset-eq-sumset-stabilizer*:
assumes $x \in G$
shows *sumset* $\{x\} (\text{stabilizer } B) = x \cdot | (\text{stabilizer } B)$
by (*simp add: Left-Coset-eq-sumset assms*)

lemma *stabilizer-subset-difference-singleton*:
assumes $S \subseteq G$ **and** $s \in S$
shows *stabilizer* $S \subseteq \text{differenceset } S \{s\}$
proof
fix x **assume** hx : $x \in \text{stabilizer } S$
then obtain t **where** ht : $t \in S$ **and** $x \oplus s = t$ **using** *assms stabilizer-def* **by**
blast
then have $x = t \ominus s$ **using** hx *stabilizer-def assms associative*
by (*metis (no-types, lifting) in-mono inverse-closed invertible invertible-right-inverse*
sub
sub.right-unit)
thus $x \in \text{differenceset } S \{s\}$ **using** ht *assms*
by (*simp add: minusset.minussetI subsetD sumset.sumsetI*)
qed

lemma *stabilizer-subset-singleton-difference*:
assumes $S \subseteq G$ **and** $s \in S$
shows *stabilizer* $S \subseteq \text{differenceset } \{s\} S$
proof–
have *stabilizer* $S \subseteq \text{minusset } (\text{stabilizer } S)$ **using** *assms Int-absorb2 minusset-eq*

subgroup.image-of-inverse submonoid.sub subset-eq
by (*smt (verit) stabilizer-is-subgroup stabilizer-subset-difference-singleton*
sumset-subset-carrier)
moreover have *minusset* $(\text{stabilizer } S) \subseteq \text{minusset } (\text{differenceset } S \{s\})$
proof
fix x **assume** $hx1$: $x \in \text{minusset } (\text{stabilizer } S)$
then have hx : *inverse* $x \in \text{stabilizer } S$
by (*metis invertible invertible-inverse-inverse minusset.cases*)
then obtain t **where** ht : $t \in S$ **and** *inverse* $x \oplus s = t$ **using** *assms stabilizer-def*
by *blast*
then have $hx2$: *inverse* $x = t \ominus s$ **using** hx *stabilizer-def assms*
by (*smt (verit, ccfv-threshold) commutative in-mono inverse-closed invertible*)

invertible-left-inverse2 sub
thus $x \in \text{minusset } (\text{differenceset } S \{s\})$ **using** *ht assms*
by (*smt (verit, best) hx1 additive-abelian-group.sumset.sumsetI additive-abelian-group-axioms*)

inverse-closed invertible invertible-inverse-inverse minusset.cases minusset.minussetI
singletonI subset-iff
qed
ultimately show *?thesis* **using** *differenceset-commute assms* **by** *blast*
qed

lemma *stabilizer-subset-nempty:*
assumes $S \neq \{\}$ **and** $S \subseteq G$
shows *stabilizer* $S \subseteq \text{differenceset } S S$
proof
fix x **assume** *hx: x ∈ stabilizer S*
then obtain s **where** *hs: s ∈ S ∩ G* **using** *assms* **by** *blast*
then have $x \oplus s \in S \cap G$ **using** *stabilizer-def assms hx mem-Collect-eq singletonI*

sumset.sumsetI sumset-Int-carrier **by** *blast*
then obtain t **where** *ht: t ∈ S* **and** $x \oplus s = t$ **by** *blast*
then have $x = t \ominus s$ **using** *hx stabilizer-def assms(2) hs ht associative*
by (*metis IntD2 inverse-closed invertible invertible-right-inverse sub sub.right-unit*)
thus $x \in \text{differenceset } S S$ **using** *ht hs* **using** *assms(2)* **by** *blast*
qed

lemma *stabilizer-coset-subset:*
assumes $A \subseteq G$ **and** $x \in A$
shows *sumset* $\{x\} (\text{stabilizer } A) \subseteq A$
proof
fix y **assume** *y ∈ sumset {x} (stabilizer A)*
moreover hence *stabilizer A ⊆ differenceset A {x}* **using** *assms*
stabilizer-subset-difference-singleton **by** *auto*
moreover have *sumset {x} (differenceset A {x}) ⊆ A*
proof
fix z **assume** *z ∈ sumset {x} (differenceset A {x})*
then obtain a **where** $a \in A$ **and** $z = x \oplus (a \ominus x)$
using *additive-abelian-group.sumset.cases additive-abelian-group-axioms singletonD*
minusset.cases assms subsetD **by** (*smt (verit, ccfv-SIG)*)
thus $z \in A$ **using** *assms*
by (*metis additive-abelian-group.inverse-closed additive-abelian-group-axioms commutative in-mono invertible invertible-right-inverse2*)
qed
ultimately show $y \in A$ **by** (*meson in-mono subset-singleton-iff sumset-mono*)
qed

lemma *stabilizer-subset-stabilizer-dvd:*

```

assumes stabilizer  $A \subseteq$  stabilizer  $B$ 
shows card (stabilizer  $A$ ) dvd card (stabilizer  $B$ )
proof(cases finite (stabilizer  $B$ ))
  case  $hB$ : True
    interpret  $H$ : subgroup-of-group stabilizer  $A$  stabilizer  $B$  ( $\oplus$ )  $\mathbf{0}$ 
    proof(unfold-locales)
      show stabilizer  $A \subseteq$  stabilizer  $B$  using assms by blast
    next
      show  $\bigwedge a b. a \in$  stabilizer  $A \implies b \in$  stabilizer  $A \implies a \oplus b \in$  stabilizer  $A$ 
        using stabilizer-is-subgroup group-axioms by simp
    next
      show  $\mathbf{0} \in$  stabilizer  $A$  using sub-unit-closed by blast
    qed
    show ?thesis using  $H.lagrange$   $hB$  by auto
  next
    case False
      then show ?thesis by simp
qed

```

```

lemma stabilizer-coset-Un:
  assumes  $A \subseteq G$ 
  shows  $(\bigcup x \in A. \text{sumset } \{x\} (\text{stabilizer } A)) = A$ 
proof
  show  $(\bigcup x \in A. \text{sumset } \{x\} (\text{stabilizer } A)) \subseteq A$ 
    using stabilizer-coset-subset assms by blast
  next
    show  $A \subseteq (\bigcup x \in A. \text{sumset } \{x\} (\text{stabilizer } A))$ 
    proof
      fix  $x$  assume  $hx: x \in A$ 
      then have  $x \in \text{sumset } \{x\} (\text{stabilizer } A)$  using sub-unit-closed assms
        by (metis in-mono right-unit singletonI sumset.sumsetI unit-closed)
      thus  $x \in (\bigcup x \in A. \text{sumset } \{x\} (\text{stabilizer } A))$  using  $hx$  by blast
    qed
qed

```

```

lemma stabilizer-empty: stabilizer  $\{\}$  =  $G$ 
  using sumset-empty Int-empty-left stabilizer-def subset-antisym
  by (smt (verit, best) mem-Collect-eq subsetI sumset-Int-carrier-eq(1) sumset-commute)

```

```

lemma stabilizer-finite:
  assumes  $S \subseteq G$  and  $S \neq \{\}$  and finite  $S$ 
  shows finite (stabilizer  $S$ )
  using stabilizer-subset-nempty assms
  by (meson finite-minusset finite-sumset rev-finite-subset)

```

```

lemma stabilizer-subset-group:
  shows stabilizer  $S \subseteq G$  using stabilizer-def by blast

```

```

lemma sumset-stabilizer-eq-iff:

```

assumes $a \in G$ **and** $b \in G$
shows $\text{sumset } \{a\} (\text{stabilizer } A) = \text{sumset } \{b\} (\text{stabilizer } A) \iff$
 $(\text{sumset } \{a\} (\text{stabilizer } A)) \cap (\text{sumset } \{b\} (\text{stabilizer } A)) \neq \{\}$
by (*simp add: assms sumset-subgroup-eq-iff*)

lemma *sumset-stabilizer-eq-Class-Union*:
assumes $A \subseteq G$
shows $\text{sumset } A (\text{stabilizer } B) = (\bigcup (\text{Class } B \text{ ' } A))$
by (*simp add: assms sumset-subgroup-eq-Class-Union*)

lemma *card-stabilizer-divide-sumset*:
assumes $A \subseteq G$
shows $\text{card } (\text{stabilizer } B) \text{ dvd } \text{card } (\text{sumset } A (\text{stabilizer } B))$
by (*simp add: assms card-divide-sumset*)

lemma *Class-image-sumset-stabilizer-eq*:
assumes $A \subseteq G$
shows $\text{Class } B \text{ ' } (\text{sumset } A (\text{stabilizer } B)) = \text{Class } B \text{ ' } A$
by (*simp add: Class-image-sumset-subgroup-eq assms*)

lemma *Class-cover-imp-subset-or-disj*:
assumes $A = (\bigcup (\text{Class } B \text{ ' } C))$ **and** $x \in G$ **and** $C \subseteq G$
shows $\text{Class } B \ x \subseteq A \vee \text{Class } B \ x \cap A = \{\}$
by (*simp add: Class-cover-imp-subset-or-disj assms*)

lemma *stabilizer-sumset-disjoint*:
fixes $S1 \ S2 :: 'a \text{ set}$
assumes $\text{stabilizer } S1 \cap \text{stabilizer } S2 = \{0\}$ **and** $S1 \subseteq G$ **and** $S2 \subseteq G$
and *finite* $S1$ **and** *finite* $S2$ **and** $S1 \neq \{\}$ **and** $S2 \neq \{\}$
shows $\text{card } (\text{sumset } (\text{stabilizer } S1) (\text{stabilizer } S2)) =$
 $\text{card } (\text{stabilizer } S1) * \text{card } (\text{stabilizer } S2)$

proof–

have *inj-on* : *inj-on* $(\lambda (a, b). a \oplus b) (\text{stabilizer } S1 \times \text{stabilizer } S2)$

proof(*intro inj-onI*)

fix $x \ y$ **assume** $x \in \text{stabilizer } S1 \times \text{stabilizer } S2$ **and** $y \in \text{stabilizer } S1 \times \text{stabilizer } S2$ **and**

$(\text{case } x \text{ of } (a, b) \Rightarrow a \oplus b) = (\text{case } y \text{ of } (a, b) \Rightarrow a \oplus b)$

then obtain $a \ b \ c \ d$ **where** $hx: x = (a, b)$ **and** $hy: y = (c, d)$ **and** $ha: a \in \text{stabilizer } S1$ **and**

$hb: b \in \text{stabilizer } S2$ **and** $hc: c \in \text{stabilizer } S1$ **and** $hd: d \in \text{stabilizer } S2$ **and**

$habcd: a \oplus b = c \oplus d$ **by** *auto*

then have $haG: a \in G$ **using** *stabilizer-def* **by** *blast*

have $hbG: b \in G$ **using** hb *stabilizer-def* **by** *blast*

have $hcG: c \in G$ **using** hc *stabilizer-def* **by** *blast*

have $hdG: d \in G$ **using** hd *stabilizer-def* **by** *blast*

then have $a \oplus c = d \oplus b$ **using** $habcd \ haG \ hbG \ hcG \ hdG$

by (*metis (full-types) associative commutative composition-closed inverse-equality invertible*

invertible-def invertible-left-inverse2)

moreover have $a \oplus c \in \text{stabilizer } S1$ **using** *ha hc stabilizer-is-subgroup subgroup.axioms(1) submonoid.sub-composition-closed*
by *(metis group.invertible group-axioms hcG subgroup.subgroup-inverse-iff)*
moreover have $d \oplus b \in \text{stabilizer } S2$ **using** *hd hb stabilizer-is-subgroup subgroup.axioms(1) submonoid.sub-composition-closed*
by *(metis group.invertible group-axioms hbG subgroup.subgroup-inverse-iff)*
ultimately have $a \oplus c = \mathbf{0}$ **and** $d \oplus b = \mathbf{0}$ **using** *assms(1)* **by** *auto*
thus $x = y$ **using** *hx hy haG hbG hcG hdG*
by *(metis inverse-closed invertible invertible-right-cancel invertible-right-inverse)*

qed

moreover have $\text{himage} : (\lambda (a, b). a \oplus b) ' (\text{stabilizer } S1 \times \text{stabilizer } S2) = \text{sumset } (\text{stabilizer } S1) (\text{stabilizer } S2)$

proof

show $(\lambda (a, b). a \oplus b) ' (\text{stabilizer } S1 \times \text{stabilizer } S2) \subseteq \text{sumset } (\text{stabilizer } S1) (\text{stabilizer } S2)$

using *stabilizer-subset-group* **by** *force*

next

show $\text{sumset } (\text{stabilizer } S1) (\text{stabilizer } S2) \subseteq (\lambda (a, b). a \oplus b) ' (\text{stabilizer } S1 \times \text{stabilizer } S2)$

proof

fix x **assume** $x \in \text{sumset } (\text{stabilizer } S1) (\text{stabilizer } S2)$

then obtain $s1$ $s2$ **where** $hs1: s1 \in \text{stabilizer } S1$ **and** $hs2: s2 \in \text{stabilizer } S2$ **and**

$x = s1 \oplus s2$ **by** *(meson sumset.cases)*

thus $x \in (\lambda (a, b). a \oplus b) ' (\text{stabilizer } S1 \times \text{stabilizer } S2)$ **using** $hs1$ $hs2$ **by** *auto*

qed

qed

ultimately show *?thesis* **using** *card-image card-cartesian-product* **by** *fastforce*

qed

lemma *stabilizer-sub-sumset-left:*

$\text{stabilizer } A \subseteq \text{stabilizer } (\text{sumset } A B)$

proof

fix x **assume** $hx: x \in \text{stabilizer } A$

then have $\text{sumset } \{x\} (\text{sumset } A B) = \text{sumset } A B$ **using** *stabilizer-def sumset-assoc*

mem-Collect-eq **by** *(smt (verit, del-insts) sumset-Int-carrier-eq(1) sumset-commute)*

thus $x \in \text{stabilizer } (\text{sumset } A B)$ **using** hx *stabilizer-def*

by *(metis (mono-tags, lifting) mem-Collect-eq sumset-Int-carrier)*

qed

lemma *stabilizer-sub-sumset-right:*

$\text{stabilizer } B \subseteq \text{stabilizer } (\text{sumset } A B)$

using *stabilizer-sub-sumset-left sumset-commute* **by** *fastforce*

lemma *not-mem-stabilizer-obtain:*

assumes $A \neq \{\}$ **and** $x \notin \text{stabilizer } A$ **and** $x \in G$ **and** $A \subseteq G$ **and** *finite* A
obtains a **where** $a \in A$ **and** $x \oplus a \notin A$
proof –
have $\text{sumset } \{x\} A \neq A$ **using** *assms stabilizer-def*
by (*metis (mono-tags, lifting) inf.orderE mem-Collect-eq*)
moreover **have** $\text{card } (\text{sumset } \{x\} A) = \text{card } A$ **using** *assms*
by (*metis card-sumset-singleton-eq inf.orderE sumset-commute*)
ultimately obtain y **where** $y \in \text{sumset } \{x\} A$ **and** $y \notin A$ **using** *assms*
by (*meson card-subset-eq subsetI*)
then obtain a **where** $a \in A$ **and** $x \oplus a \notin A$ **using** *assms*
by (*metis singletonD sumset.cases*)
thus *?thesis* **using** *that* **by** *blast*
qed

lemma *sumset-eq-sub-stabilizer*:
assumes $A \subseteq G$ **and** $B \subseteq G$ **and** *finite* B
shows $\text{sumset } A B = B \implies A \subseteq \text{stabilizer } B$
proof
fix x **assume** $hsum: \text{sumset } A B = B$ **and** $hx: x \in A$
have $\text{sumset } \{x\} B = B$
proof –
have $\text{sumset } \{x\} B \subseteq B$ **using** $hsum$ hx
by (*metis empty-subsetI equalityE insert-subset sumset-mono*)
moreover **have** $\text{card } (\text{sumset } \{x\} B) = \text{card } B$ **using** *assms*
by (*metis IntD1 Int-absorb1 card-sumset-singleton-eq hx inf-commute sum-set-commute*)
ultimately show *?thesis* **using** *card-subset-eq assms(3)* **by** *auto*
qed
thus $x \in \text{stabilizer } B$ **using** hx *assms(1)* *stabilizer-def*
by (*metis (mono-tags, lifting) assms(2) inf.orderE mem-Collect-eq subsetD*)
qed

lemma *sumset-stabilizer-eq*:
shows $\text{sumset } (\text{stabilizer } A) (\text{stabilizer } A) = \text{stabilizer } A$
proof
show $\text{sumset } (\text{stabilizer } A) (\text{stabilizer } A) \subseteq \text{stabilizer } A$
using *stabilizer-is-subgroup subgroup.axioms(1) subsetI*
by (*metis (mono-tags, lifting) additive-abelian-group.sumset.simps additive-abelian-group-axioms*
submonoid.sub-composition-closed)
next
show $\text{stabilizer } A \subseteq \text{sumset } (\text{stabilizer } A) (\text{stabilizer } A)$
using *Left-Coset-eq-sumset stabilizer-is-nonempty*
stabilizer-subset-group sub-unit-closed additive-abelian-group-axioms right-unit
subset-iff sumsetI **by** (*smt (verit, best)*)
qed

```

lemma differenceset-stabilizer-eq:
  shows differenceset (stabilizer A) (stabilizer A) = stabilizer A
proof
  show differenceset (stabilizer A) (stabilizer A) ⊆ stabilizer A
  proof
    fix x assume x ∈ differenceset (stabilizer A) (stabilizer A)
    then obtain a b where a ∈ stabilizer A and b ∈ stabilizer A and x = a ⊖ b
    by (metis minusset.cases sumset.cases)
    thus x ∈ stabilizer A using stabilizer-is-subgroup subgroup.axioms(1)
    by (smt (verit, ccfv-threshold) in-mono invertible stabilizer-subset-group
      subgroup-inverse-iff sub-composition-closed)
  qed
next
  show stabilizer A ⊆ differenceset (stabilizer A) (stabilizer A)
  proof
    fix x assume hx: x ∈ stabilizer A
    then have x ⊖ 0 ∈ differenceset (stabilizer A) (stabilizer A) by blast
    then show x ∈ differenceset (stabilizer A) (stabilizer A) using hx by simp
  qed
qed

lemma stabilizer2-sub-stabilizer:
  shows stabilizer(stabilizer A) ⊆ stabilizer A
proof(cases A ≠ {})
  case True
    then have stabilizer(stabilizer A) ⊆ differenceset (stabilizer A) (stabilizer A)
    by (simp add: stabilizer-is-nonempty stabilizer-subset-group stabilizer-subset-nempty)
    thus ?thesis using differenceset-stabilizer-eq by blast
  next
  case False
    then show ?thesis by (simp add: stabilizer-empty stabilizer-subset-group)
qed

lemma stabilizer-left-sumset-invariant:
  assumes a ∈ G and A ⊆ G
  shows stabilizer (sumset {a} A) = stabilizer A

proof
  show stabilizer (sumset {a} A) ⊆ stabilizer A
  proof
    fix x assume hx: x ∈ stabilizer (sumset {a} A)
    then have hxG: x ∈ G using stabilizer-def by blast
    have sumset {x} (sumset {a} A) = sumset {a} A using stabilizer-def hx
    by (metis (mono-tags, lifting) mem-Collect-eq sumset-Int-carrier)
    then have sumset {x} A = A using assms
    by (metis (full-types) sumset-assoc sumset-commute sumset-subset-carrier
      sumset-translate-eq-right)
    thus x ∈ stabilizer A using hxG stabilizer-def
  qed

```

by (*metis (mono-tags, lifting) mem-Collect-eq sumset-Int-carrier*)
 qed
 next
 show $\text{stabilizer } A \subseteq \text{stabilizer } (\text{sumset } \{a\} A)$ using *stabilizer-def*
 using *stabilizer-sub-sumset-right* by *meson*
 qed

lemma *stabilizer-right-sumset-invariant*:
 assumes $a \in G$ and $A \subseteq G$
 shows $\text{stabilizer } (\text{sumset } A \{a\}) = \text{stabilizer } A$
 using *sumset-commute stabilizer-left-sumset-invariant* *assms* by *simp*

lemma *stabilizer-right-differenceset-invariant*:
 assumes $b \in G$ and $A \subseteq G$
 shows $\text{stabilizer } (\text{differenceset } A \{b\}) = \text{stabilizer } A$
 using *assms minuset-eq stabilizer-right-sumset-invariant* by *auto*

lemma *stabilizer-unchanged*:
 assumes $a \in G$ and $b \in G$
 shows $\text{stabilizer } (\text{sumset } A B) = \text{stabilizer } (\text{sumset } A (\text{sumset } (\text{differenceset } B \{b\}) \{a\}))$

proof–
 have $\text{sumset } A (\text{sumset } (\text{differenceset } B \{b\}) \{a\}) = \text{sumset } (\text{differenceset } (\text{sumset } A B) \{b\}) \{a\}$
 by (*simp add: sumset-assoc*)
 thus *?thesis* using *stabilizer-right-sumset-invariant*
stabilizer-right-differenceset-invariant *assms* *sumset-subset-carrier* by *simp*
 qed

lemma *subset-stabilizer-of-subset-sumset*:
 assumes $A \subseteq \text{sumset } \{x\} (\text{stabilizer } B)$ and $x \in G$ and $A \neq \{\}$ and $A \subseteq G$
 shows $\text{stabilizer } A \subseteq \text{stabilizer } B$

proof–
 obtain a where $ha: a \in A$ using *assms* by *blast*
 moreover then obtain b where $hb: b \in \text{stabilizer } B$ and $haxb: a = x \oplus b$
 using *sumset.cases* *assms* by *blast*
 ultimately have $\text{stabilizer } A \subseteq \text{differenceset } A \{a\}$ using *assms* *sumset-subset-carrier*

stabilizer-subset-difference-singleton by (*meson subset-trans*)
 also have $\dots = \text{sumset } \{\text{inverse } a\} A$ using *sumset-commute* ha *assms*(4) *inverse-closed*

subsetD *minuset-eq* by *auto*
 also have $\dots \subseteq \text{sumset } \{\text{inverse } x \oplus \text{inverse } b\} (\text{sumset } \{x\} (\text{stabilizer } B))$
 using *assms* *sumset-mono* $haxb$ *inverse-closed* hb *stabilizer-subset-group* *subsetD*

commutative inverse-composition-commute by (*metis invertible subset-singleton-iff*)
 also have $\dots = \text{sumset } \{\text{inverse } b\} (\text{stabilizer } B)$ using *sumset-singletons-eq*

commutative
assms sumset-assoc hb stabilizer-subset-group inverse-closed invertible
by (*metis composition-closed invertible-right-inverse2 sub*)
also have $\dots = \text{stabilizer } B$ **using** *hb Left-Coset-eq-sumset sub-unit-closed sub subset-iff*
additive-abelian-group-axioms calculation disjoint-iff-not-equal factor-unit inverse-closed
sumset-subgroup-eq-iff **by** (*smt (verit, del-insts)*)
finally show *?thesis* .
qed

lemma *sumset-stabilizer-eq-self*:
assumes $A \subseteq G$
shows *sumset (stabilizer A) A = A*
using *assms sumset-eq-Union-left[OF stabilizer-subset-group]*
Int-absorb2 stabilizer-coset-Un sumset-commute sumset-eq-Union-left **by** *presburger*

lemma *stabilizer-neq-subset-sumset*:
assumes $A \subseteq \text{sumset } \{x\}$ (*stabilizer B*) **and** $x \in A$ **and** $\neg \text{sumset } \{x\}$ (*stabilizer B*) $\subseteq C$ **and**
 $A \subseteq C$ **and** $C \subseteq G$
shows *stabilizer A \neq stabilizer B*
proof
assume *heq: stabilizer A = stabilizer B*
obtain *a* **where** $a \in \text{sumset } \{x\}$ (*stabilizer B*) **and**
 $a \notin C$ **using** *assms* **by** *blast*
moreover then obtain *b* **where** $b \in \text{stabilizer B}$ **and** $a = x \oplus b$ **using** *sumset.cases* **by** *blast*
ultimately have $b \oplus x \notin A$ **using** *commutative stabilizer-subset-group assms in-mono* **by** *metis*
thus *False* **using** *assms heq stabilizer-coset-subset subset-trans* **by** *metis*
qed

lemma *subset-stabilizer-Un*:
shows *stabilizer A \cap stabilizer B \subseteq stabilizer (A \cup B)*
proof
fix *x* **assume** *hx: x \in stabilizer A \cap stabilizer B*
then have *sumset {x} (A \cap G) = A \cap G* **using** *stabilizer-def* **by** *blast*
moreover have *sumset {x} (B \cap G) = (B \cap G)* **using** *stabilizer-def hx* **by** *blast*
ultimately have *sumset {x} ((A \cup B) \cap G) = (A \cup B) \cap G* **using** *sumset-subset-Un2*
boolean-algebra.conj-disj-distrib2 **by** *auto*
then show $x \in \text{stabilizer (A \cup B)}$ **using** *hx stabilizer-subset-group stabilizer-def*
by *blast*
qed

lemma *mem-stabilizer-Un-and-left-imp-right*:
assumes *finite B* **and** $x \in \text{stabilizer (A \cup B)}$ **and** $x \in \text{stabilizer A}$ **and** *disjnt A*

B
shows $x \in \text{stabilizer } B$
proof –
have $(A \cap G) \cup \text{sumset } \{x\} (B \cap G) = (A \cap G) \cup (B \cap G)$
using *assms(2) sumset-subset-Un2[of {x} A ∩ G B ∩ G] stabilizer-def[of A ∪ B]*
Int-Un-distrib2[of A B G] assms(3) stabilizer-def
by (*metis (mono-tags, lifting) mem-Collect-eq*)
then have $B \cap G \subseteq \text{sumset } \{x\} (B \cap G)$ **using** *assms(4) disjnt-def Int-Un-distrib2 Int-commute*
sumset-subset-Un1 **by** (*smt (verit, del-insts) Int-assoc Un-Int-eq(2) inf.orderI insert-is-Un*
sumset-empty(2))
then show $x \in \text{stabilizer } B$ **using** *stabilizer-def[of B] assms(1) assms(3) card-subset-eq*

card-sumset-singleton-subset-eq finite.emptyI finite.insertI finite-Int finite-sumset

inf.cobounded2 stabilizer-subset-group subsetD **by** (*smt (verit) mem-Collect-eq*)
qed

lemma *mem-stabilizer-Un-and-right-imp-left*:
assumes *finite A* **and** $x \in \text{stabilizer } (A \cup B)$ **and** $x \in \text{stabilizer } B$ **and** *disjnt A B*
shows $x \in \text{stabilizer } A$
using *mem-stabilizer-Un-and-left-imp-right Un-commute assms disjnt-sym* **by** *metis*

lemma *Union-stabilizer-Class-eq*:
assumes $A \subseteq G$
shows $A = (\bigcup (\text{Class } A ' A))$ **using** *assms sumset-commute sumset-subgroup-eq-Class-Union*

sumset-stabilizer-eq-self **by** *presburger*

lemma *card-stabilizer-sumset-divide-sumset*:
card (stabilizer (sumset A B)) dvd card (sumset A B) **using** *card-divide-sumset sumset-commute sumset-stabilizer-eq-self sumset-subset-carrier* **by** *metis*

lemma *card-stabilizer-le*:
assumes $A \subseteq G$ **and** *finite A* **and** $A \neq \{\}$
shows $\text{card } (\text{stabilizer } A) \leq \text{card } A$ **using** *assms*
by (*metis card-le-sumset finite.cases insertCI insert-subset stabilizer-finite stabilizer-subset-group sumset-commute sumset-stabilizer-eq-self*)

lemma *sumset-Inter-subset-sumset*:
assumes $a \in G$ **and** $b \in G$
shows $\text{sumset } (A \cap \text{sumset } \{a\} (\text{stabilizer } C)) (B \cap \text{sumset } \{b\} (\text{stabilizer } C))$
 \subseteq
 $\text{sumset } \{a \oplus b\} (\text{stabilizer } C)$ (**is** $\text{sumset } ?A ?B \subseteq -$)
proof

fix x **assume** $x \in \text{sumset } ?A \ ?B$
then obtain $d1 \ d2$ **where** $d1 \in \text{sumset } \{a\}$ (*stabilizer* C) **and**
 $d2 \in \text{sumset } \{b\}$ (*stabilizer* C) **and** $x = d1 \oplus d2$ **by** (*meson IntD2 sumset.cases*)
then obtain $c1 \ c2$ **where** $hc1: c1 \in \text{stabilizer } C$ **and** $hc2: c2 \in \text{stabilizer } C$
and
 $x = (a \oplus c1) \oplus (b \oplus c2)$ **using** *sumset.simps* **by** *auto*
then have $x = (a \oplus b) \oplus (c1 \oplus c2)$ **using** $hc1 \ hc2$ *assms associative commutative*

stabilizer-subset-group **by** *simp*
thus $x \in \text{sumset } \{a \oplus b\}$ (*stabilizer* C) **using** *stabilizer-is-subgroup hc1 hc2*
stabilizer-subset-group sumset.simps sumset-stabilizer-eq assms **by** *blast*
qed

1.3 Convergent

definition *convergent* $:: 'a \text{ set} \Rightarrow 'a \text{ set} \Rightarrow 'a \text{ set} \Rightarrow \text{bool}$ **where**
convergent $C \ A \ B \equiv C \subseteq \text{sumset } A \ B \wedge C \neq \{\}$ \wedge
 $\text{card } C + \text{card } (\text{stabilizer } C) \geq \text{card } (A \cap B) + \text{card } (\text{sumset } (A \cup B) \ (\text{stabilizer } C))$

definition *convergent-set* $:: 'a \text{ set} \Rightarrow 'a \text{ set} \Rightarrow 'a \text{ set set}$ **where**
convergent-set $A \ B = \text{Collect } (\lambda \ C. \text{convergent } C \ A \ B)$

lemma *convergent-set-sub-powerset*:
convergent-set $A \ B \subseteq \text{Pow } (\text{sumset } A \ B)$ **using** *convergent-set-def convergent-def*
by *blast*

lemma *finite-convergent-set*:
assumes *finite* A **and** *finite* B
shows *finite* (*convergent-set* $A \ B$)
using *convergent-set-sub-powerset finite-Pow-iff finite-sumset assms finite-subset*
by *metis*

1.4 Technical lemmas from DeVos's proof of Kneser's Theorem

The following lemmas correspond to intermediate arguments in the proof of Kneser's Theorem by DeVos that we will be following [1].

lemma *stabilizer-sumset-psubset-stabilizer*:
assumes $a \in G$ **and** $b \in G$ **and** $A \cap \text{sumset } \{a\}$ (*stabilizer* C) $\neq \{\}$ **and**
 $B \cap \text{sumset } \{b\}$ (*stabilizer* C) $\neq \{\}$ **and** $\text{hnotsub: } \neg \text{sumset } \{a \oplus b\}$ (*stabilizer* C) $\subseteq \text{sumset } A \ B$
shows *stabilizer* (*sumset* ($A \cap \text{sumset } \{a\}$ (*stabilizer* C)) ($B \cap \text{sumset } \{b\}$ (*stabilizer* C))) \subset
stabilizer C (**is** $?H \subset -$)
proof
have *sumset* ($A \cap \text{sumset } \{a\}$ (*stabilizer* C)) ($B \cap \text{sumset } \{b\}$ (*stabilizer* C)) $\neq \{\}$
using *assms* **by** (*simp add: inf-assoc*)

then show $?H \subseteq \text{stabilizer } C$
by (*meson* *assms*(1) *assms*(2) *composition-closed subset-stabilizer-of-subset-sumset* *sumset-Inter-subset-sumset* *sumset-subset-carrier*)
next
obtain $c1\ c2$ **where** $a \oplus c1 \in A$ **and** $b \oplus c2 \in B$ **and** $hc1: c1 \in \text{stabilizer } C$
and $hc2: c2 \in \text{stabilizer } C$
using *assms*(1, 2, 3, 4) *Left-Coset-eq-sumset-stabilizer* **by** *fastforce*
then have $hc1mem: (a \oplus c1) \in A \cap \text{sumset } \{a\}$ (*stabilizer } C*) **and** $hc1G: a \oplus c1 \in G$ **and**
 $hc2mem: (b \oplus c2) \in B \cap \text{sumset } \{b\}$ (*stabilizer } C*) **and** $hc2G: b \oplus c2 \in G$
by (*auto simp add: assms*(1, 2) *sumset.sumsetI*)
have $(a \oplus c1) \oplus (b \oplus c2) \in \text{sumset } \{a \oplus b\}$ (*stabilizer } C*) **using** *assms* $hc1\ hc2$
by (*smt* (*verit*) *associative commutative composition-closed insertI1* *sub sub-composition-closed* *sumset.sumsetI*)
then have $\text{sumset } \{a \oplus b\}$ (*stabilizer } C*) \cap $\text{sumset } \{(a \oplus c1) \oplus (b \oplus c2)\}$
(*stabilizer } C*) $\neq \{\}$
using *zero-mem-stabilizer* **by** (*smt* (*verit*, *ccfv-threshold*) *composition-closed* *disjoint-iff-not-equal* $hc1G\ hc2G$ *insertCI* *right-unit* *sumset.sumsetI* *unit-closed*)
then have $hsumeq: \text{sumset } \{a \oplus b\}$ (*stabilizer } C*) $= \text{sumset } \{(a \oplus c1) \oplus (b \oplus c2)\}$
(*stabilizer } C*)
using *sumset-stabilizer-eq-iff* *assms* $hc1G\ hc2G$ *composition-closed* **by** *presburger*
have $(\text{sumset } (A \cap \text{sumset } \{a\}$ (*stabilizer } C*)) $(B \cap \text{sumset } \{b\}$ (*stabilizer } C*)))
 $\subseteq \text{sumset } \{a \oplus b\}$ (*stabilizer } C*)
by (*simp add: assms*(1, 2) *sumset-Inter-subset-sumset*)
have $hsummem: (a \oplus c1) \oplus (b \oplus c2) \in \text{sumset } (A \cap \text{sumset } \{a\}$ (*stabilizer } C*))
 $(B \cap \text{sumset } \{b\}$ (*stabilizer } C*))
using $hc1mem\ hc2mem\ hc1G\ hc2G\ \text{sumset.sumsetI}$ **by** *blast*
show $?H \neq \text{stabilizer } C$
using *stabilizer-neq-subset-sumset*[*OF - hsummem*] *hnotsub* $hsumeq$ *sumset-Inter-subset-sumset* *assms*
sumset-subset-carrier *composition-closed* *sumset-mono* *sumset.sumsetI* *zero-mem-stabilizer*

inf.cobounded1 *right-unit* *unit-closed* **by** *metis*
qed

lemma *stabilizer-eq-stabilizer-union*:

assumes $a \in G$ **and** $b \in G$ **and** $A \cap \text{sumset } \{a\}$ (*stabilizer } C*) $\neq \{\}$ **and**
 $B \cap \text{sumset } \{b\}$ (*stabilizer } C*) $\neq \{\}$ **and** *hnotsub*: $\neg \text{sumset } \{a \oplus b\}$ (*stabilizer } C*)
 $\subseteq \text{sumset } A\ B$ **and**
 $C \subseteq \text{sumset } A\ B$ **and** *finite } C* **and**
 $C \cap \text{sumset } (A \cap \text{sumset } \{a\}$ (*stabilizer } C*)) $(B \cap \text{sumset } \{b\}$ (*stabilizer } C*)) $=$
 $\{\}$ **and** $C \neq \{\}$ **and**
finite } A **and** *finite } B*
shows *stabilizer* ($\text{sumset } (A \cap \text{sumset } \{a\}$ (*stabilizer } C*)) $(B \cap \text{sumset } \{b\}$
(*stabilizer } C*))) $=$
stabilizer ($C \cup \text{sumset } (A \cap \text{sumset } \{a\}$ (*stabilizer } C*)) $(B \cap \text{sumset } \{b\}$
(*stabilizer } C*))) **(is** *stabilizer } ?H = stabilizer } ?K*)
proof

```

show stabilizer ?H  $\subseteq$  stabilizer ?K using subset-stabilizer-Un Int-absorb1
  stabilizer-sumset-psubset-stabilizer psubset-imp-subset assms by metis
next
  have hCG : C  $\subseteq$  G using assms(6) sumset-subset-carrier by force
  show stabilizer ?K  $\subseteq$  stabilizer ?H
  proof
    fix x assume hxC1: x  $\in$  stabilizer ?K
    moreover have x  $\in$  stabilizer C
    proof–
      have hC-Un: C = ( $\bigcup$  (Class C ‘ C)) using Union-stabilizer-Class-eq hCG
    by simp
      have hCsumx: sumset {x} C = ( $\bigcup$  y  $\in$  Class C ‘ C. sumset {x} y)
      proof
        show sumset {x} C  $\subseteq$   $\bigcup$  (sumset {x} ‘ Class C ‘ C)
        proof
          fix y assume hy: y  $\in$  sumset {x} C
          then obtain c where hc: c  $\in$  C and hyc: y = x  $\oplus$  c using sumset.cases
        by blast
          then obtain K where hK: K  $\in$  Class C ‘ C and c  $\in$  K using hC-Un
        by blast
          then have y  $\in$  sumset {x} K using hyc hc by (metis sumset.cases
            sumset.sumsetI hCG hy singletonD subset-iff)
          then show y  $\in$   $\bigcup$  (sumset {x} ‘ Class C ‘ C) using hK by auto
        qed
      next
        show  $\bigcup$  (sumset {x} ‘ Class C ‘ C)  $\subseteq$  sumset {x} C
        proof(intro Union-least)
          fix X assume X  $\in$  sumset {x} ‘ Class C ‘ C
          then obtain K where K  $\in$  Class C ‘ C and X = sumset {x} K by blast
          then show X  $\subseteq$  sumset {x} C using sumset-mono[of {x} {x} K C]
            hC-Un subset-refl by blast
        qed
      qed
    have x  $\notin$  stabilizer C  $\implies$  False
    proof–
      assume hxC: x  $\notin$  stabilizer C
      then have hxG: x  $\in$  G using hxC1 stabilizer-subset-group by blast
      then have hxCne: sumset {x} C  $\neq$  C using stabilizer-def[of C] hCG
    Int-absorb2
      hxC by (metis (mono-tags, lifting) mem-Collect-eq)
      moreover have hxsplitt: sumset {x} C  $\cup$  sumset {x} ?H = C  $\cup$  ?H
      using hxC1 stabilizer-def[of ?K] sumset-subset-carrier assms(6) sum-
    set-subset-Un2 by force
      have sumset {x} C  $\cap$  ?H  $\neq$  {}
      proof
        assume sumset {x} C  $\cap$  ?H = {}
        then have sumset {x} C  $\subset$  C using hxsplitt hxCne by blast
        thus False using hCG assms(6) assms(7) hxC1 stabilizer-subset-group
    psubset-card-mono

```

by (metis card-sumset-singleton-eq sumset-Int-carrier sumset-commute
 sumset-stabilizer-eq-self hxG less-irrefl-nat)
 qed
 then obtain c where hc: c ∈ C and
 hxcne: sumset {x} (Class C c) ∩ ?H ≠ {} using hCsumx by blast
 then have hxc: sumset {x} (Class C c) = Class C (x ⊕ c)
 using hxG assms(6) Left-Coset-Class-unit Left-Coset-eq-sumset-stabilizer
 sumset-assoc
 sumset-singletons-eq composition-closed sumset.cases sumset-stabilizer-eq-self
 hCG by (smt (verit))
 have hClassEmpty: Class C (x ⊕ c) ∩ C = {}
 proof-
 have ¬ Class C (x ⊕ c) ⊆ C using hxc hxcne assms(8) by blast
 then show ?thesis using Class-cover-imp-subset-or-disj[OF hC-Un - hCG]
 by (meson composition-closed hCG hc hxG subsetD)
 qed
 have Class C (x ⊕ c) ⊆ sumset {x} C using hCsumx hc hxc by blast
 then have Class C (x ⊕ c) ⊆ ?H using hClassEmpty hxsplitt by auto
 moreover have card (Class C (x ⊕ c)) = card (stabilizer C) using hxG hc
 hCG
 composition-closed Right-Coset-Class-unit Right-Coset-cardinality sum-
 set-Int-carrier
 Class-cover-imp-subset-or-disj assms by auto
 ultimately have card (stabilizer C) ≤ card ?H using card-mono finite-sumset
 assms(10, 11) finite-Int by metis
 moreover have card ?H < card (sumset {a ⊕ b} (stabilizer C))
 proof (intro psubset-card-mono psubsetI sumset-Inter-subset-sumset assms(1)
 assms(2))
 show finite (sumset {a ⊕ b} (stabilizer C))
 using stabilizer-finite assms finite-sumset by (simp add: hCG)
 next
 show ?H ≠ sumset {a ⊕ b} (stabilizer C)
 using hnotsub sumset-mono by (metis Int-lower1)
 qed
 ultimately show False
 using assms(1, 2) stabilizer-subset-group by (simp add: card-sumset-singleton-subset-eq)

qed
 then show ?thesis by auto
 qed
 moreover have finite ?H using finite-sumset assms(10, 11) finite-Int by simp
 ultimately show x ∈ stabilizer ?H using mem-stabilizer-Un-and-right-imp-left[of
 ?H x C]
 disjnt-def assms Un-commute by (metis disjoint-iff-not-equal)
 qed
 qed

lemma sumset-inter-ineq:
 assumes B ∩ sumset {a} (stabilizer C) = {} and stabilizer (sumset (A ∩ sumset

$\{a\} (\text{stabilizer } C) (B \cap \text{sumset } \{b\} (\text{stabilizer } C)) \subseteq \text{stabilizer } C$ **and**
 $a \in A$ **and** $a \in G$ **and** *finite* A **and** *finite* B **and** $A \neq \{\}$ **and** $B \neq \{\}$ **and** *finite*
(stabilizer $C)$
shows $\text{int} (\text{card} (\text{sumset} (A \cup B) (\text{stabilizer } C))) - \text{card} (\text{sumset} (A \cup B)$
 $(\text{stabilizer} (\text{sumset} (A \cap \text{sumset } \{a\} (\text{stabilizer } C)) (B \cap \text{sumset } \{b\} (\text{stabilizer}$
 $C)))) \geq$
 $\text{int} (\text{card} (\text{stabilizer } C)) - \text{card} (\text{sumset} (A \cap \text{sumset } \{a\} (\text{stabilizer } C))$
 $(\text{stabilizer} (\text{sumset} (A \cap \text{sumset } \{a\} (\text{stabilizer } C)) (B \cap \text{sumset } \{b\} (\text{stabilizer}$
 $C))))$
(is $\text{int} (\text{card} (\text{sumset} (A \cup B) (\text{stabilizer } C))) - \text{card} (\text{sumset} (A \cup B) ?H1) \geq$
 $\text{int} (\text{card} (\text{stabilizer } C)) - \text{card} (\text{sumset } ?A1 ?H1)$

proof –

have $\text{hfnsumH1}: \text{finite} (\text{sumset} (A \cup B) ?H1)$
using *finite-sumset* *assms* **by** (*meson* *finite-Un* *psubsetE* *rev-finite-subset*)
have $\text{hsubsumH1}: \text{sumset} (A \cup B) ?H1 \subseteq \text{sumset} (A \cup B) (\text{stabilizer } C)$
using *sumset.cases* *assms* **by** (*meson* *psubsetE* *subset-refl* *sumset-mono*)
have $\text{hsumH1card-le}: \text{card} (\text{sumset} (A \cup B) ?H1) \leq \text{card} (\text{sumset} (A \cup B)$
 $(\text{stabilizer } C))$
using *card-mono* *finite-sumset* *stabilizer-finite* *assms*
by (*metis* *equalityE* *finite-UnI* *psubset-imp-subset* *sumset-mono*)
have $\text{hsub}: \text{sumset } ?A1 ?H1 \subseteq \text{sumset } \{a\} (\text{stabilizer } C)$

proof

fix x **assume** $x \in \text{sumset } ?A1 ?H1$
then obtain $h1$ f **where** $h1 \in ?A1$ **and** $hf: f \in ?H1$ **and**
 $hx: x = h1 \oplus f$ **by** (*meson* *sumset.cases*)
then obtain c **where** $hc: c \in \text{stabilizer } C$ **and** $hac: h1 = a \oplus c$
by (*metis* *Int-iff* *empty-iff* *insert-iff* *sumset.cases*)
then have $hcf: c \oplus f \in \text{stabilizer } C$ **using** hf *assms*(2) *stabilizer-is-subgroup*
subgroup-def *monoid-axioms* *Group-Theory.group.axioms*(1)
Group-Theory.monoid-def *subset-iff* *psubset-imp-subset*
by (*smt* (*verit*) *stabilizer-subset-group* *sumset.sumsetI* *sumset-stabilizer-eq*)
have $hcG: c \in G$ **using** hc *stabilizer-subset-group* **by** *auto*
have $hfG: f \in G$ **using** hf *stabilizer-subset-group* **by** *auto*
show $x \in \text{sumset } \{a\} (\text{stabilizer } C)$ **using** hx hac *assms* *stabilizer-subset-group*
 hcf
using *Left-Coset-eq-sumset-stabilizer* *Left-Coset-memI* *associative* hcG hfG

by *presburger*

qed

moreover have *finite* $(\text{sumset } ?A1 ?H1)$ **using** *finite-sumset* *assms* *stabilizer-finite*
finite-subset
by (*metis* *finite.simps* $hsub$)
ultimately have $\text{card} (\text{sumset } \{a\} (\text{stabilizer } C)) - \text{card} (\text{sumset } ?A1 ?H1) =$
 $\text{card} (\text{sumset } \{a\} (\text{stabilizer } C)) - \text{sumset } ?A1 ?H1$
using *card-Diff-subset* **by** *metis*
moreover have $\text{card} (\text{sumset } ?A1 ?H1) \leq \text{card} (\text{sumset } \{a\} (\text{stabilizer } C))$
using *card-mono* $hsub$ *finite-sumset* *assms* **by** (*metis* *finite.simps*)
ultimately have $\text{int} (\text{card} (\text{sumset } \{a\} (\text{stabilizer } C))) - \text{card} (\text{sumset } ?A1$
 $?H1) =$
 $\text{card} (\text{sumset } \{a\} (\text{stabilizer } C)) - \text{sumset } ?A1 ?H1$ **by** *linarith*

also have $\dots \leq \text{card} ((\text{sumset } (A \cup B) (\text{stabilizer } C)) - (\text{sumset } (A \cup B) ?H1))$
proof–
have $\text{sumset } \{a\} (\text{stabilizer } C) - \text{sumset } ?A1 ?H1 \subseteq \text{sumset } (A \cup B) (\text{stabilizer } C) - \text{sumset } (A \cup B) ?H1$
proof
fix x **assume** $hx: x \in \text{sumset } \{a\} (\text{stabilizer } C) - \text{sumset } ?A1 ?H1$
then obtain c **where** $hxac: x = a \oplus c$ **and** $hc: c \in \text{stabilizer } C$ **and** $hcG: c \in G$
using sumset.cases **by** blast
then have $x \in \text{sumset } (A \cup B) (\text{stabilizer } C)$ **using** $\text{assms sumset.cases}$ **by** blast
moreover have $x \notin \text{sumset } (A \cup B) ?H1$
proof
assume $x \in \text{sumset } (A \cup B) ?H1$
then obtain y $h1$ **where** $hy: y \in A \cup B$ **and** $hyG: y \in G$ **and** $hh1G: h1 \in G$
and $hh1: h1 \in ?H1$ **and** $hxy: x = y \oplus h1$ **by** $(\text{meson sumset.cases})$
then have $y = a \oplus (c \oplus \text{inverse } h1)$ **using** $hxac$ hxy $\text{assms associative commutative composition-closed}$
inverse-closed invertible invertible-left-inverse2 **by** (metis hcG)
moreover have $h1 \in \text{stabilizer } C$ **using** $hh1$ assms by auto
moreover hence $c \oplus \text{inverse } h1 \in \text{stabilizer } C$ **using** hc $\text{stabilizer-is-subgroup subgroup-def}$
 $\text{group-axioms invertible subgroup.subgroup-inverse-iff submonoid.sub-composition-closed}$
 $hh1G$ **by** metis
ultimately have $y \in \text{sumset } \{a\} (\text{stabilizer } C)$
using assms hcG hh1G **by** blast
moreover hence $y \in A$ **using** $\text{assms}(1)$ hy **by** auto
ultimately have $x \in \text{sumset } ?A1 ?H1$ **using** hxy $hh1$
by $(\text{simp add: hyG hh1G sumset.sumsetI})$
thus False **using** hx **by** auto
qed
ultimately show $x \in \text{sumset } (A \cup B) (\text{stabilizer } C) - \text{sumset } (A \cup B) ?H1$
by simp
qed
thus $?thesis$ **using** $\text{card-mono finite-Diff finite-sumset assms}$
by $(\text{metis finite-UnI nat-int-comparison}(3))$
qed
also have $\dots = \text{int} (\text{card} (\text{sumset } (A \cup B) (\text{stabilizer } C))) - \text{card} (\text{sumset } (A \cup B) ?H1)$
using $\text{card-Diff-subset}[OF hfinsumH1 hsubsumH1] hsumH1\text{card-le}$ **by** linarith
finally show $\text{int} (\text{card} (\text{sumset } (A \cup B) (\text{stabilizer } C))) - \text{card} (\text{sumset } (A \cup B) ?H1) \geq$
 $\text{int} (\text{card} (\text{stabilizer } C)) - \text{card} (\text{sumset } ?A1 ?H1)$
using $\text{assms by (metis card-sumset-singleton-subset-eq stabilizer-subset-group)}$
qed

lemma $\text{exists-convergent-min-stabilizer}$:

assumes $\text{hind}: \forall m < n. \forall C D. C \subseteq G \longrightarrow D \subseteq G \longrightarrow \text{finite } C \longrightarrow \text{finite } D \longrightarrow$

$C \neq \{\}$ \longrightarrow
 $D \neq \{\}$ \longrightarrow $\text{card}(\text{sumset } C D) + \text{card } C = m \longrightarrow$
 $\text{card}(\text{sumset } C (\text{stabilizer}(\text{sumset } C D))) + \text{card}(\text{sumset } D (\text{stabilizer}(\text{sumset } C D))) -$
 $\text{card}((\text{stabilizer}(\text{sumset } C D)))$
 $\leq \text{card}(\text{sumset } C D)$ **and** $hAG: A \subseteq G$ **and** $hBG: B \subseteq G$ **and** $hA: \text{finite } A$
and
 $hB: \text{finite } B$ **and** $hAne: A \neq \{\}$ **and** $A \cap B \neq \{\}$ **and**
 $hcardsum: \text{card}(\text{sumset } A B) + \text{card } A = n$ **and** $hintercardA: \text{card}(A \cap B) <$
 $\text{card } A$
obtains X **where** $\text{convergent } X A B$ **and** $\bigwedge Y. Y \in \text{convergent-set } A B \implies$
 $\text{card}(\text{stabilizer } Y) \geq \text{card}(\text{stabilizer } X)$
proof –
let $?C0 = \text{sumset}(A \cap B)(A \cup B)$
have $hC0ne: ?C0 \neq \{\}$ **using** assms **by** fast
moreover **have** $\text{finite } ?C0$ **using** $\text{sumset-inter-union-subset finite-sumset assms}$
by auto
ultimately **have** $\text{finite}(\text{stabilizer } ?C0)$ **using** stabilizer-finite
using $\text{sumset-subset-carrier}$ **by** presburger
then **have** $hcard\text{-sumset-le}: \text{card}(A \cap B) \leq \text{card}(\text{sumset}(A \cap B)(\text{stabilizer } ?C0))$
using $\text{card-le-sumset sumset-commute sub-unit-closed assms}$
by $(\text{metis Int-Un-eq(3) Un-subset-iff finite-Int unit-closed})$
have $\text{card } ?C0 \leq \text{card}(\text{sumset } A B)$
using $\text{card-mono sumset-inter-union-subset finite-sumset assms}$
by $(\text{simp add: card-mono finite-sumset hA hB sumset-inter-union-subset})$
then **have** $\text{card } ?C0 + \text{card}(A \cap B) < \text{card}(\text{sumset } A B) + \text{card } A$
using $hintercardA$ **by** auto
then **obtain** m **where** $m < n$ **and** $\text{card } ?C0 + \text{card}(A \cap B) = m$ **using**
 $hcardsum$ **by** auto
then **have** $\text{card}(\text{sumset}(A \cap B)(\text{stabilizer } ?C0)) +$
 $\text{card}(\text{sumset}(A \cup B)(\text{stabilizer } ?C0)) - \text{card}(\text{stabilizer } ?C0) \leq \text{card } ?C0$
using $\text{assms finite-Un finite-Int}$
by $(\text{metis Int-Un-eq(4) Un-empty Un-subset-iff})$
then **have** $\text{card } ?C0 + \text{card}(\text{stabilizer } ?C0) \geq$
 $\text{card}(A \cap B) + \text{card}(\text{sumset}(A \cup B)(\text{stabilizer } ?C0))$ **using** $hcard\text{-sumset-le}$
by auto
then **have** $?C0 \in \text{convergent-set } A B$ **using** $\text{convergent-set-def convergent-def}$
 $\text{sumset-inter-union-subset hC0ne}$ **by** auto
then **have** $hconvergent\text{-ne}: \text{convergent-set } A B \neq \{\}$ **by** auto
define KS **where** $KS \equiv (\lambda X. \text{card}(\text{stabilizer } X))$ ‘ $\text{convergent-set } A B$
define K **where** $K \equiv \text{Min } KS$
define C **where** $C \equiv @C. C \in \text{convergent-set } A B \wedge K = \text{card}(\text{stabilizer } C)$
obtain $KS: \text{finite } KS$ $KS \neq \{\}$
using $hconvergent\text{-ne finite-convergent-set assms KS-def}$ **by** auto
then **have** $K \in KS$ **using** $K\text{-def Min-in}$ **by** blast
then **have** $\exists X. X \in \text{convergent-set } A B \wedge K = \text{card}(\text{stabilizer } X)$
using $KS\text{-def}$ **by** auto
then **obtain** $C \in \text{convergent-set } A B$ **and** $\text{Keq}: K = \text{card}(\text{stabilizer } C)$

by (*metis* (*mono-tags*, *lifting*) *C-def someI-ex*)
then have $hC: C \subseteq \text{sumset } A B$ **and** $hCne: C \neq \{\}$ **and**
 $hCcard: \text{card } C + \text{card } (\text{stabilizer } C) \geq$
 $\text{card } (A \cap B) + \text{card } (\text{sumset } (A \cup B) (\text{stabilizer } C))$
using *convergent-set-def convergent-def* **by** *auto*
have $hCmin: \bigwedge Y. Y \in \text{convergent-set } A B \implies$
 $\text{card } (\text{stabilizer } Y) \geq \text{card } (\text{stabilizer } C)$
using *K-def KS-def Keq Min-le KS(1)* **by** *auto*
show *?thesis* **using** $hCmin$ hC $hCcard$ $hCne$ *local.convergent-def* **that** **by** *pres-*
burger
qed

end

context *normal-subgroup*
begin

1.5 A function that picks coset representatives randomly

definition $\varphi :: 'a \text{ set} \Rightarrow 'a$ **where**

$\varphi = (\lambda x. \text{if } x \in G // K \text{ then } (\text{SOME } a. a \in G \wedge x = a \cdot | K) \text{ else undefined})$

definition *quot-comp-alt* $:: 'a \Rightarrow 'a \Rightarrow 'a$ **where** *quot-comp-alt* $a b = \varphi ((a \cdot b) \cdot | K)$

lemma *phi-eq-coset*:

assumes $\varphi x = a$ **and** $a \in G$ **and** $x \in G // K$

shows $x = a \cdot | K$

proof –

have $(\text{SOME } a. a \in G \wedge x = a \cdot | K) = a$ **using** *φ-def assms* **by** *simp*

then show *?thesis* **using** *some-eq-ex representant-exists Left-Coset-Class-unit assms*

by (*metis* (*mono-tags*, *lifting*))

qed

lemma *phi-coset-mem*:

assumes $a \in G$

shows $\varphi (a \cdot | K) \in a \cdot | K$

proof –

obtain x **where** $hx: x = \varphi (a \cdot | K)$ **by** *auto*

then have $x = (\text{SOME } x. x \in G \wedge a \cdot | K = x \cdot | K)$ **using** *φ-def assms*

Class-in-Partition Left-Coset-Class-unit **by** *presburger*

then show *?thesis* **using** *φ-def Class-self Left-Coset-Class-unit hx assms*

by (*smt* (*verit*, *ccfv-SIG*) *tfl-some*)

qed

lemma *phi-coset-eq*:

assumes $a \in G$ **and** $\varphi x = a$ **and** $x \in G // K$

shows $\varphi (a \cdot | K) = a$ **using** *phi-eq-coset assms* **by** *metis*

lemma *phi-inverse-right*:
assumes $g \in G$
shows $\text{quot-comp-alt } g (\varphi (\text{inverse } g \cdot | K)) = \varphi K$
proof –
have $g \cdot (\varphi (\text{inverse } g \cdot | K)) \in (g \cdot (\text{inverse } g) \cdot | K)$
using *phi-coset-mem assms by (smt (z3) Left-Coset-memE factor-unit invertible invertible-right-inverse invertible-inverse-closed invertible-inverse-inverse sub invertible-left-inverse2)*
then have $g \cdot (\varphi (\text{inverse } g \cdot | K)) \cdot | K = K$
using *Block-self Left-Coset-Class-unit Normal-def quotient.unit-closed sub*
by (*metis assms composition-closed invertible invertible-inverse-closed invertible-right-inverse*)
then show *?thesis using quot-comp-alt-def by auto*
qed

lemma *phi-inverse-left*:
assumes $g \in G$
shows $\text{quot-comp-alt } (\varphi (\text{inverse } g \cdot | K)) g = \varphi K$
proof –
have $(\varphi (\text{inverse } g \cdot | K)) \cdot g \in ((\text{inverse } g) \cdot g) \cdot | K$ **using** *phi-coset-mem assms*
by (*metis Left-Coset-memE factor-unit invertible invertible-inverse-closed invertible-left-inverse normal*)
then have $(\varphi (\text{inverse } g \cdot | K)) \cdot g \cdot | K = K$ **using** *Block-self Left-Coset-Class-unit Normal-def*
quotient.unit-closed sub **by** (*smt (verit, best) assms composition-closed invertible invertible-inverse-closed invertible-left-inverse*)
then show *?thesis using quot-comp-alt-def by auto*
qed

lemma *phi-mem-coset-eq*:
assumes $a \in G // K$ **and** $b \in G$
shows $\varphi a \in b \cdot | K \implies a = (b \cdot | K)$
proof –
assume $\varphi a \in b \cdot | K$
then have $a \cap (b \cdot | K) \neq \{\}$
by (*metis Class-closed Class-is-Left-Coset Int-iff assms empty-iff phi-coset-mem phi-eq-coset*)
then show $a = b \cdot | K$ **by** (*metis Class-in-Partition Class-is-Left-Coset assms disjoint*)
qed

lemma *forall-unique-repr*:
 $\forall x \in G // K. \exists! k \in \varphi '(G // K). x = k \cdot | K$
proof
fix x **assume** $hx: x \in G // K$
then have $\varphi x \cdot | K = x$

by (*metis Class-is-Left-Coset block-closed phi-coset-mem phi-eq-coset representant-exists*)
then have $hex: \exists k \in \varphi '(G // K). x = k \cdot | K$ **using** hx **by** *blast*
moreover have $\bigwedge a b. a \in \varphi '(G // K) \implies x = a \cdot | K \implies b \in \varphi '(G // K)$
 $\implies x = b \cdot | K \implies$
 $a = b$
proof–
fix $a b$ **assume** $a \in \varphi '(G // K)$ **and** $hxa: x = a \cdot | K$ **and** $b \in \varphi '(G // K)$
and
 $hxb: x = b \cdot | K$
then obtain $z w$ **where** $a = \varphi (z \cdot | K)$ **and** $b = \varphi (w \cdot | K)$ **and** $z \in G$ **and**
 $w \in G$
using *representant-exists Left-Coset-Class-unit* **by** *force*
then show $a = b$ **using** $hxa hxb$
by (*metis Class-in-Partition Class-is-Left-Coset block-closed phi-coset-mem phi-eq-coset*)
qed
ultimately show $\exists! k \in \varphi '(G // K). x = k \cdot | K$ **by** *blast*
qed

lemma *phi-inj-on*:
shows *inj-on* $\varphi (G // K)$
proof(*intro inj-onI*)
fix $x y$ **assume** $x \in G // K$ **and** $hy: y \in G // K$ **and** $hxy: \varphi x = \varphi y$
then obtain $a b$ **where** $x = a \cdot | K$ **and** $y = b \cdot | K$ **and** $a \in G$ **and** $b \in G$
using *representant-exists Left-Coset-Class-unit* **by** *metis*
then show $x = y$ **using** $hxy hy$ **by** (*metis phi-coset-mem phi-mem-coset-eq*)
qed

lemma *phi-coset-eq-self*:
assumes $a \in G // K$
shows $\varphi a \cdot | K = a$
by (*metis Class-closed Class-is-Left-Coset assms phi-coset-mem phi-eq-coset representant-exists*)

lemma *phi-coset-comp-eq*:
assumes $a \in G // K$ **and** $b \in G // K$
shows $\varphi a \cdot \varphi b \cdot | K = a \cdot | b$ **using** *assms phi-coset-eq-self*
by (*metis Class-is-Left-Coset block-closed factor-composition phi-coset-mem representant-exists*)

lemma *phi-comp-eq*:
assumes $a \in G // K$ **and** $b \in G // K$
shows $\varphi (a \cdot | b) = \text{quot-comp-alt } (\varphi a) (\varphi b)$
using *phi-coset-comp-eq quot-comp-alt-def assms* **by** *auto*

lemma *phi-image-subset*:
 $\varphi '(G // K) \subseteq G$
proof(*intro image-subsetI, simp add: phi-def*)

```

fix  $x$  assume  $x \in G // K$ 
then show (SOME  $a. a \in G \wedge x = a \cdot | K \in G$ )
using Left-Coset-Class-unit representant-exists someI-ex by (metis (mono-tags, lifting))
qed

lemma phi-image-group:
  Group-Theory.group ( $\varphi \text{ ' } (G // K)$ ) quot-comp-alt ( $\varphi K$ )
proof –
  have hmonoid: Group-Theory.monoid ( $\varphi \text{ ' } (G // K)$ ) quot-comp-alt ( $\varphi K$ )
  proof
    show  $\bigwedge a b. a \in \varphi \text{ ' } (G // K) \implies b \in \varphi \text{ ' } (G // K) \implies$ 
      quot-comp-alt  $a b \in \varphi \text{ ' } (G // K)$  using quot-comp-alt-def imageI phi-image-subset
      by (metis Class-in-Partition Left-Coset-Class-unit composition-closed subset-iff)
    next
      show ( $\varphi K \in \varphi \text{ ' } (G // K)$ ) using  $\varphi$ -def Left-Coset-Class-unit imageI Normal-def by blast
    next
      show  $\bigwedge a b c. a \in \varphi \text{ ' } \textit{Partition} \implies b \in \varphi \text{ ' } \textit{Partition} \implies c \in \varphi \text{ ' } \textit{Partition}$ 
       $\implies$ 
        quot-comp-alt (quot-comp-alt  $a b$ )  $c = \textit{quot-comp-alt}$   $a$  (quot-comp-alt  $b c$ )
      proof –
        fix  $a b c$  assume  $ha: a \in \varphi \text{ ' } (G // K)$  and  $hb: b \in \varphi \text{ ' } (G // K)$  and  $hc: c \in \varphi \text{ ' } (G // K)$ 
        have  $habc: a \cdot b \cdot c \in G$  using  $ha hb hc$  composition-closed phi-image-subset by (meson subsetD)
        have  $hab: \textit{quot-comp-alt}$   $a b \in (a \cdot b) \cdot | K$  using phi-image-subset quot-comp-alt-def ha hb
        by (metis composition-closed phi-coset-mem subsetD)
        then have quot-comp-alt (quot-comp-alt  $a b$ )  $c \in (a \cdot b \cdot c) \cdot | K$  using quot-comp-alt-def phi-image-subset ha hb hc
        by (smt (z3) Block-self Class-closed Class-in-Partition Left-Coset-Class-unit composition-closed
          natural.commutates-with-composition phi-coset-mem subset-iff)
        moreover have  $hbc: \textit{quot-comp-alt}$   $b c \in (b \cdot c) \cdot | K$  using  $hb hc$  phi-image-subset quot-comp-alt-def
        by (metis composition-closed phi-coset-mem subset-iff)
        moreover hence quot-comp-alt  $a$  (quot-comp-alt  $b c$ )  $\in (a \cdot b \cdot c) \cdot | K$ 
using quot-comp-alt-def phi-image-subset ha hb hc
        by (smt (verit, del-Insts) Block-self Class-closed Class-in-Partition Left-Coset-Class-unit
          associative composition-closed natural.commutates-with-composition phi-coset-mem subset-iff)
        moreover have  $a \cdot (\textit{quot-comp-alt}$   $b c) \cdot | K \in G // K$  using  $ha hb hc$  phi-image-subset
        by (metis Class-closed Class-in-Partition Class-is-Left-Coset hbc composition-closed in-mono subset-eq)
        moreover have (quot-comp-alt  $a b$ )  $\cdot c \cdot | K \in G // K$  using  $ha hb hc$ 

```

phi-image-subset
by (*metis Class-closed Class-in-Partition Left-Coset-Class-unit hab com-position-closed in-mono*)
ultimately show *quot-comp-alt (quot-comp-alt a b) c = quot-comp-alt a (quot-comp-alt b c)*
using *phi-mem-coset-eq[OF - habc] quot-comp-alt-def by metis*
qed
next
show $\bigwedge a. a \in \varphi \text{ ' Partition} \implies \text{quot-comp-alt } (\varphi K) a = a$ **using** *quot-comp-alt-def*
phi-def
phi-image-subset image-iff phi-coset-eq subsetD by (smt (z3) Normal-def
Partition-def
natural.image.sub-unit-closed phi-comp-eq quotient.left-unit)
next
show $\bigwedge a. a \in \varphi \text{ ' Partition} \implies \text{quot-comp-alt } a (\varphi K) = a$ **using** *quot-comp-alt-def*
phi-def
phi-image-subset image-iff phi-coset-eq subsetD by (smt (verit) Normal-def
factor-composition factor-unit normal-subgroup.phi-coset-eq-self normal-subgroup-axioms

quotient.unit-closed right-unit unit-closed)
qed
moreover show *Group-Theory.group* $(\varphi \text{ ' } (G // K)) \text{quot-comp-alt } (\varphi K)$
proof (*simp add: group-def group-axioms-def hmonoid*)
show $\forall u. u \in \varphi \text{ ' Partition} \longrightarrow \text{monoid.invertible } (\varphi \text{ ' Partition}) \text{quot-comp-alt}$
 $(\varphi K) u$
proof (*intro allI impI*)
fix *g* **assume** *hg: g ∈ φ ' (G // K)*
then have *quot-comp-alt g (φ ((inverse g) · | K)) = (φ K)*
and *quot-comp-alt (φ ((inverse g) · | K)) g = (φ K)*
using *phi-image-subset phi-inverse-right phi-inverse-left by auto*
moreover have $\varphi ((\text{inverse } g) \cdot | K) \in \varphi \text{ ' } (G // K)$ **using** *imageI hg*
phi-image-subset
by (*metis (no-types, opaque-lifting) Class-in-Partition Left-Coset-Class-unit*
in-mono
invertible invertible-inverse-closed)
ultimately show *monoid.invertible* $(\varphi \text{ ' Partition}) \text{quot-comp-alt } (\varphi K) g$
using *monoid.invertibleI[OF hmonoid] hg by presburger*
qed
qed
qed

lemma *phi-map: Set-Theory.map* $\varphi \text{ Partition } (\varphi \text{ ' Partition})$
by (*auto simp add: Set-Theory.map-def phi-def*)

lemma *phi-image-isomorphic:*
group-isomorphism $\varphi (G // K) ([\cdot]) (\text{Class } \mathbf{1}) (\varphi \text{ ' } (G // K)) \text{quot-comp-alt } (\varphi K)$
proof –
have *bijective-map* $\varphi \text{ Partition } (\varphi \text{ ' Partition})$

```

    using bijective-map-def bijective-def bij-betw-def phi-inj-on phi-map by blast
    moreover have Group-Theory.monoid ( $\varphi$  ' Partition) quot-comp-alt ( $\varphi$  K)
    using phi-image-group group-def by metis
    moreover have  $\varphi$  (Class 1) =  $\varphi$  K using Left-Coset-Class-unit Normal-def by
auto
    ultimately show ?thesis
    by (auto simp add: group-isomorphism-def group-homomorphism-def monoid-homomorphism-def

    phi-image-group quotient.monoid-axioms quotient.group-axioms monoid-homomorphism-axioms-def

    phi-comp-eq phi-map)
qed

end

context subgroup-of-additive-abelian-group

begin

lemma Union-Coset-card-eq:
  assumes hSG:  $S \subseteq G$  and hSU:  $(\bigcup (Class ' S)) = S$ 
  shows  $card\ S = card\ H * card\ (Class\ ' S)$ 
proof (cases finite H)
  case hH: True
  have hfin:  $\bigwedge A. A \in Class\ ' S \implies finite\ A$  using hSG Right-Coset-Class-unit
    Right-Coset-cardinality hH card-eq-0-iff empty-iff sub-unit-closed subsetD
  by (smt (verit, del-Insts) imageE)
  have  $card\ S = card\ H * card\ (Class\ ' S)$  when hS: finite S
  proof -
    have hdisj: pairwise ( $\lambda s\ t. disjnt\ s\ t$ ) (Class ' S)
    proof (intro pairwiseI)
      fix  $x\ y$  assume  $x \in Class\ ' S$  and  $y \in Class\ ' S$  and hxy:  $x \neq y$ 
      then obtain  $a\ b$  where  $x \in Class\ a$  and  $y \in Class\ b$  and
         $a \in S$  and  $b \in S$  by blast
      then show  $disjnt\ x\ y$  using disjnt-def hxy
      by (smt (verit, ccfv-threshold) not-disjoint-implies-equal hSG subsetD)
    qed
    then have  $card\ (\bigcup (Class\ ' S)) = sum\ card\ (Class\ ' S)$  using card-Union-disjoint
    hfin by blast
    moreover have finite (Class ' S) using hS by blast
    ultimately have  $card\ (\bigcup (Class\ ' S)) = (\sum a \in Class\ ' S. card\ a)$ 
      using sum-card-image hdisj by blast
    moreover have  $\bigwedge a. a \in Class\ ' S \implies card\ a = card\ H$ 
      using hSG Right-Coset-Class-unit Right-Coset-cardinality by auto
    ultimately show  $card\ S = card\ H * card\ (Class\ ' S)$ 
      using hSU by simp
  qed
  moreover have  $card\ S = card\ H * card\ (Class\ ' S)$  when hS:  $\neg finite\ S$ 
  using finite-Union hfin hS hSU by (metis card-eq-0-iff mult-0-right)

```

```

ultimately show ?thesis by blast
next
case hH: False
have card S = card H * card (Class ' S) when S = {}
  by (simp add: that)
then have hinf:  $\bigwedge A. A \in \text{Class ' } S \implies \text{infinite } A$  using hSG Right-Coset-Class-unit

  Right-Coset-cardinality hH card-eq-0-iff empty-iff sub-unit-closed subsetD
  by (smt (verit) Class-self imageE)
moreover have card S = card H * card (Class ' S) when S  $\neq$  {} using hSU
by (metis Class-closed2
  Normal-def card.infinite card-sumset-0-iff hH hSG mult-is-0 sumset-subgroup-eq-Class-Union
  unit-closed)
ultimately show ?thesis by fastforce
qed

end

context subgroup-of-abelian-group
begin

interpretation GH: additive-abelian-group G // H ([·]) Class 1
proof
  fix x y assume x  $\in$  G // H and y  $\in$  G // H
  then show x [·] y = y [·] x using Class-commutes-with-composition commutative
  representant-exists
  by metis
qed

interpretation GH-repr: additive-abelian-group  $\varphi$  ' (G // H) quot-comp-alt  $\varphi$  H
proof (simp add: additive-abelian-group-def abelian-group-def phi-image-group
  commutative-monoid-def commutative-monoid-axioms-def, intro conjI allI impI)
  show Group-Theory.monoid ( $\varphi$  ' Partition) quot-comp-alt ( $\varphi$  H)
  using phi-image-group group-def by metis
next
  show  $\bigwedge x y. x \in \varphi$  ' Partition  $\implies y \in \varphi$  ' Partition  $\implies$  quot-comp-alt x y =
  quot-comp-alt y x
  by (auto) (metis GH.commutative phi-comp-eq)
qed

lemma phi-image-sumset-eq:
  assumes A  $\subseteq$  G // H and B  $\subseteq$  G // H
  shows  $\varphi$  ' (GH.sumset A B) = GH-repr.sumset ( $\varphi$  ' A) ( $\varphi$  ' B)
proof (intro subset-antisym image-subsetI subsetI)
  fix x assume x  $\in$  GH.sumset A B
  then obtain c d where x = quotient-composition c d and hc: c  $\in$  A and hd:
  d  $\in$  B
  using GH.sumset.cases by blast
  then have  $\varphi$  x = quot-comp-alt ( $\varphi$  c) ( $\varphi$  d)

```



```

    using phi-comp-eq assms subsetD by blast
  then show  $\varphi x \in GH\text{-repr.sumset } (\varphi ' A) (\varphi ' B)$ 
    using hc hd assms subsetD GH-repr.sumsetI imageI by auto
next
  fix x assume  $x \in GH\text{-repr.sumset } (\varphi ' A) (\varphi ' B)$ 
  then obtain a b where  $x = \text{quot-comp-alt } a b$  and ha:  $a \in \varphi ' A$  and hb:  $b \in \varphi ' B$ 
    using GH-repr.sumset.cases by metis
  moreover obtain c d where  $a = \varphi c$  and  $b = \varphi d$  and  $c \in A$  and  $d \in B$ 
    using ha hb by blast
  ultimately show  $x \in \varphi ' GH.\text{sumset } A B$  using phi-comp-eq assms imageI
  GH.sumsetI
    by (smt (verit, del-insts) subsetD)
qed

```

lemma phi-image-stabilizer-eq:

```

  assumes  $A \subseteq G // H$ 
  shows  $\varphi ' (GH.\text{stabilizer } A) = GH\text{-repr.stabilizer } (\varphi ' A)$ 
proof(intro subset-antisym image-subsetI subsetI)
  fix x assume  $x \in GH.\text{stabilizer } A$ 
  then have  $GH.\text{sumset } \{x\} A = A$  and hx:  $x \in G // H$  using GH.stabilizer-def
  assms by auto
  then have  $GH\text{-repr.sumset } (\varphi ' \{x\}) (\varphi ' A) = \varphi ' A$  using assms phi-image-sumset-eq
    by (metis empty-subsetI insert-subset)
  then show  $\varphi x \in GH\text{-repr.stabilizer } (\varphi ' A)$  using GH-repr.stabilizer-def assms
    by (smt (z3) GH-repr.sumset-Int-carrier hx image-empty image-eqI image-insert
  mem-Collect-eq)
next
  fix x assume  $x \in GH\text{-repr.stabilizer } (\varphi ' A)$ 
  then have hstab:  $GH\text{-repr.sumset } \{x\} (\varphi ' A) = (\varphi ' A)$  and hx:  $x \in \varphi ' (G // H)$ 
    using GH-repr.stabilizer-def assms phi-image-subset by auto
  then obtain B where hB:  $B \in G // H$  and hBx:  $\varphi B = x$  by blast
  then have  $GH\text{-repr.sumset } (\varphi ' \{B\}) (\varphi ' A) = \varphi ' A$  using hstab by auto
  then have  $GH.\text{sumset } \{B\} A = A$  using phi-image-sumset-eq phi-inj-on assms
  hB
  GH.sumset-subset-carrier by (smt (z3) GH.sumset-singletons-eq inj-on-image-eq-iff

  quotient.right-unit quotient.unit-closed)
  then show  $x \in \varphi ' (GH.\text{stabilizer } A)$  using assms hBx GH.stabilizer-def
    by (smt (z3) GH.sumset-Int-carrier hB image-iff mem-Collect-eq)
qed

```

end

1.6 Useful group-theoretic results

lemma residue-group: abelian-group $\{0..(m :: nat)-1\} (\lambda x y. ((x + y) \text{ mod } m))$
 $(0 :: int)$

```

proof(cases m > 1)
  case hm: True
    then have hmonoid: Group-Theory.monoid {0..m-1} (λ x y. ((x + y) mod m))
      (0 :: int)
      by (unfold-locales, auto simp add: of-nat-diff, presburger)
    moreover have monoid.invertible {0..int (m - 1)} (λ x y. (x + y) mod int m)
      0 u if u ∈ {0..int (m - 1)} for u
    proof(cases u = 0)
      case True
        then show ?thesis using monoid.invertible-def[OF hmonoid that] monoid.unit-invertible[OF
hmonoid] by simp
      next
        case hx: False
          then have ((m - u) + u) mod m = 0 and (u + (m - u)) mod m = 0 and
m - u ∈ {0..int(m-1)}
          using atLeastAtMost-iff hx that by auto
          then show ?thesis using monoid.invertible-def[OF hmonoid that] by metis
        qed
      moreover have commutative-monoid {0..m-1} (λ x y. ((x + y) mod m)) (0 ::
int)
      using hmonoid commutative-monoid-def commutative-monoid-axioms-def by
(smt (verit))
      ultimately show ?thesis by (simp add: abelian-group-def group-def group-axioms-def

      hmonoid)
    next
      case hm: False
        moreover have hmonoid: Group-Theory.monoid {0} (λ x y. ((x + y) mod m))
          (0 :: int)
          by (unfold-locales, auto)
        moreover have monoid.invertible {0} (λ x y. (x + y) mod int m) 0 0 using
monoid.invertible-def[OF hmonoid]
          monoid.unit-invertible[OF hmonoid] hm by simp
        ultimately show ?thesis by (unfold-locales, auto)
      qed

lemma (in subgroup-of-group) prime-order-simple:
  assumes prime (card G)
  shows H = {1} ∨ H = G
proof -
  have card H dvd card G using lagrange assms card.infinite dvdI not-prime-0 by
fastforce
  then have card H = 1 ∨ card H = card G using assms prime-nat-iff by blast
  then show ?thesis using card-1-singletonE sub-unit-closed card.infinite card-subset-eq
sub
  assms not-prime-0 subsetI insertE empty-iff by metis
qed

lemma residue-group-simple:

```

```

assumes prime p and subgroup H {0..(p :: nat)-1} (λ x y. ((x + y) mod p))
(0 :: int)
shows H = {0} ∨ H = {0..int(p-1)}
proof –
  have hprime: prime (card {0..int(p-1)}) using card-atLeastAtMost-int assms
int-ops by auto
  moreover have hsub:subgroup-of-group H {0..(p :: nat)-1} (λ x y. ((x + y)
mod p)) (0 :: int)
  using subgroup-of-group-def assms abelian-group-def residue-group by fast
  ultimately show ?thesis using assms subgroup-of-group.prime-order-simple[OF
hsub hprime] by blast
qed

end

```

2 Kneser’s Theorem and the CauchyDavenport Theorem: main proofs

```

theory Kneser-Cauchy-Davenport-main-proofs
imports
  Kneser-Cauchy-Davenport-preliminaries

```

```

begin

```

```

context additive-abelian-group

```

```

begin

```

2.1 Proof of Kneser’s Theorem

The proof we formalise follows the paper [1]. This version of Kneser’s Theorem corresponds to Theorem 3.2 in [3], or to Theorem 4.3 in [2].

theorem *Kneser*:

```

assumes A ⊆ G and B ⊆ G and finite A and finite B and hAne: A ≠ {} and
hBne: B ≠ {}

```

```

shows card (sumset A B) ≥ card (sumset A (stabilizer (sumset A B))) +
card (sumset B (stabilizer (sumset A B))) – card (stabilizer (sumset A B))

```

```

proof –

```

```

have ∧ n A B. additive-abelian-group G (⊕) 0 ⇒ A ⊆ G ⇒ B ⊆ G ⇒
finite A ⇒ finite B ⇒ A ≠ {} ⇒ B ≠ {} ⇒ card (sumset A B) + card
A = n ⇒

```

```

card (sumset A B) ≥ card (sumset A (stabilizer (sumset A B))) +
card (sumset B (stabilizer (sumset A B))) – card ((stabilizer (sumset A B)))

```

```

proof –

```

```

fix n

```

```

show ∧ A B. additive-abelian-group G (⊕) 0 ⇒ A ⊆ G ⇒ B ⊆ G ⇒
finite A ⇒ finite B ⇒ A ≠ {} ⇒ B ≠ {} ⇒ card (sumset A B) + card
A = n ⇒

```

```

card (sumset A B) ≥ card (sumset A (stabilizer (sumset A B))) +
card (sumset B (stabilizer (sumset A B))) - card ((stabilizer (sumset A B)))
proof(induction n arbitrary : G (⊕) 0 rule: nat-less-induct)
  fix n
  fix A B G :: 'a set
  fix add (infixl [⊕] 65)
  fix zero ([0])
  assume hind: ∀ m < n. ∀ x xa xb :: 'a set. ∀ xc xd.
    additive-abelian-group xb xc xd → x ⊆ xb →
    xa ⊆ xb → finite x → finite xa → x ≠ {} → xa ≠ {} →
    card (additive-abelian-group.sumset xb xc x xa) + card x = m →
    card (additive-abelian-group.sumset xb xc x (additive-abelian-group.stabilizer
xb xc
  (additive-abelian-group.sumset xb xc x xa))) + card (additive-abelian-group.sumset
xb xc xa
  (additive-abelian-group.stabilizer xb xc (additive-abelian-group.sumset xb xc
x xa))) -
    card (additive-abelian-group.stabilizer xb xc (additive-abelian-group.sumset
xb xc x xa))
    ≤ card (additive-abelian-group.sumset xb xc x xa) and
    hGroupG: additive-abelian-group G ([⊕]) [0] and hAG: A ⊆ G and hBG: B
⊆ G and
    hA: finite A and hB: finite B and hAne: A ≠ {} and hBne: B ≠ {} and
    hcardsum: card (additive-abelian-group.sumset G ([⊕]) A B) + card A = n
  interpret G: additive-abelian-group G ([⊕]) [0] using hGroupG by simp
  have hindG: ∀ m < n. ∀ C D. C ⊆ G →
    D ⊆ G → finite C → finite D → C ≠ {} → D ≠ {} →
    card (G.sumset C D) + card C = m →
    card (G.sumset C (G.stabilizer
(G.sumset C D))) + card (G.sumset D
(G.stabilizer (G.sumset C D))) -
    card (G.stabilizer (G.sumset C D))
    ≤ card (G.sumset C D) using hind hGroupG by blast
  show card (G.sumset A (G.stabilizer (G.sumset A B))) +
    card (G.sumset B (G.stabilizer (G.sumset A B))) -
    card (G.stabilizer (G.sumset A B)) ≤ card (G.sumset A B)
  proof(cases G.stabilizer (G.sumset A B) = {[0]})
    case hstab0: True
      show ?thesis
      proof (cases card A = 1)
        case True
          then obtain a where A = {a} and a ∈ G
            by (metis hAG card-1-singletonE insert-subset)
          then show ?thesis using G.card-sumset-singleton-subset-eq
            G.stabilizer-left-sumset-invariant G.stabilizer-subset-group G.sumset-commute

            G.sumset-stabilizer-eq-self hBG by (metis diff-add-inverse eq-imp-le)
        next
          case False

```

then have $\text{card } A \geq 2$ **using** *Suc-1 order-antisym-conv*
by (*metis Suc-eq-plus1 bot.extremum card-seteq hA hAne le-add2*
not-less-eq-eq)
then obtain $a \ a'$ **where** $\text{haA}: \{a', a\} \subseteq A$ **and** $\text{hanot}: a' \neq a$ **and** $\text{ha1G}: a \in G$ **and**
 $\text{ha2G}: a' \in G$
by (*smt (verit, ccfv-threshold) card-2-iff hAG insert-subset*
obtain-subset-with-card-n subset-trans)
then have $(a' [\oplus] (G.\text{inverse } a)) \notin G.\text{stabilizer } B$ **using** *G.stabilizer-sub-sumset-right*

 $\text{hstab0 subset-singletonD}$ **by** (*metis G.commutative G.inverse-closed*
G.invertible G.invertible-right-inverse2 G.right-unit empty-iff insert-iff)
then obtain b **where** $\text{hb}: b \in B$ **and** $(a' [\oplus] (G.\text{inverse } a)) [\oplus] b \notin B$
using *G.stabilizer-def G.not-mem-stabilizer-obtain hBG hB hBne ha1G*
 ha2G
G.composition-closed G.inverse-closed **by** (*metis (no-types, lifting)*)
then have $\text{habG}: a' [\oplus] b [\oplus] G.\text{inverse } a \notin B$
using $\text{hb } hBG \text{ ha1G } \text{ha2G}$ **by** (*metis G.associative G.commutative*
G.inverse-closed subset-iff)
have $\text{hbG}: b \in G$ **using** $\text{hb } hBG$ **by** *auto*
let $?B' = G.\text{sumset } (G.\text{differenceset } B \ \{b\}) \ \{a\}$
have $\text{hB'}: \text{finite } ?B'$ **using** hB
by (*simp add: G.finite-minusset G.finite-sumset*)
have $\text{hB'G}: ?B' \subseteq G$ **using** *G.sumset-subset-carrier* **by** *blast*
have $\text{hB'ne}: ?B' \neq \{\}$ **using** $\text{hBne } \text{hbG } \text{ha1G } \text{sumset-is-empty-iff } hBG$ **by**
auto
have $\text{hstabeq}: G.\text{stabilizer } (G.\text{sumset } A \ B) = G.\text{stabilizer } (G.\text{sumset } A$
 $\ ?B')$
using $\text{hbG } \text{ha1G } hAG \ hBG$ *G.stabilizer-unchanged* **by** *blast*
have $\text{hstab0'}: G.\text{stabilizer } (G.\text{sumset } A \ ?B') = \{[0]\}$ **using** $\text{hstab0 } \text{hstabeq}$
by *blast*
have $\text{ha1B'}: a \in ?B'$
proof–
have $(b [\oplus] G.\text{inverse } b) [\oplus] a \in ?B'$ **using** $\text{hBG } \text{hbG } \text{ha1G } \text{hb}$
G.minusset.minussetI **by** *blast*
thus $a \in ?B'$ **by** (*simp add: hbG ha1G*)
qed
then have $\text{hinter-nempty}: A \cap ?B' \neq \{\}$ **using** $\text{ha1B'} \text{ haA}$ **by** *blast*
have $\text{ha2B'}: a' \notin ?B'$
proof–
have $\text{h1}: (a' [\oplus] b [\oplus] G.\text{inverse } a) [\oplus] G.\text{inverse } b \notin G.\text{differenceset } B$
 $\{b\}$
proof
assume $(a' [\oplus] b [\oplus] G.\text{inverse } a) [\oplus] G.\text{inverse } b \in G.\text{differenceset } B$
 $\{b\}$
then obtain b' **where** $(a' [\oplus] b [\oplus] G.\text{inverse } a) [\oplus] G.\text{inverse } b = b'$
 $[\oplus] G.\text{inverse } b$
and $b' \in B$ **using** *hbG G.minusset.simps G.sumset.cases* **by** *force*
then have $(a' [\oplus] b [\oplus] G.\text{inverse } a) \in B$ **using** hbG

```

    by (smt (verit) G.composition-closed hBG ha1G ha2G G.inverse-closed
G.invertible
      G.invertible-right-cancel subsetD)
    thus False using habG by auto
  qed
  have ((a' [⊕] b [⊕] G.inverse a) [⊕] G.inverse b) [⊕] a ∉ ?B'
  proof
    assume ((a' [⊕] b [⊕] G.inverse a) [⊕] G.inverse b) [⊕] a ∈ ?B'
    then obtain b' where ((a' [⊕] b [⊕] G.inverse a) [⊕] G.inverse b) [⊕]
a = b' [⊕] a
      and b' ∈ G.differenceset B {b} using ha1G G.sumset.simps by auto
    then have ((a' [⊕] b [⊕] G.inverse a) [⊕] G.inverse b) ∈ G.differenceset
B {b}
      using ha1G by (smt (z3) G.sumset.simps G.additive-abelian-group-axioms
      G.composition-closed ha2G hbG G.inverse-closed G.invertible
G.invertible-right-cancel)
    thus False using h1 by auto
  qed
  thus a' ∉ ?B' using ha1G hbG
  by (smt (verit, del-Insts) G.associative G.commutative G.composition-closed
ha2G
      G.inverse-closed G.invertible G.invertible-left-inverse G.right-unit)
  qed
  have hinterA: A ∩ ?B' ≠ A using haA ha2B' by auto
  have hintercard0: 0 < card (A ∩ ?B')
    using hA hB hinter-nempty card-gt-0-iff by blast
  have hintercardA: card (A ∩ ?B') < card A using hA hB hinterA card-mono
    by (simp add: psubsetI psubset-card-mono)
  have inj: inj-on (λ x. x [⊕] G.inverse b [⊕] a) G using inj-onI ha1G hb
G.invertible
      G.inverse-closed G.composition-closed by (smt (verit) G.invertible-right-cancel
hbG)
  have 1: card (G.sumset A B) = card (G.sumset A ?B')
  using G.card-differenceset-singleton-mem-eq G.card-sumset-singleton-subset-eq
hAG hB'G
      ha1G hbG G.sumset-assoc G.sumset-commute G.sumset-subset-carrier
    by (smt (verit, del-Insts))
  obtain C where hCconv: G.convergent C A ?B' and hCmin: ⋀ Y. Y ∈
G.convergent-set A ?B'
    ⇒ card (G.stabilizer Y) ≥ card (G.stabilizer C)
  using G.exists-convergent-min-stabilizer[of n A ?B']
      hindG hA hB' hAG hB'G hinter-nempty hAne hcardsum hintercardA 1
hGroupG by auto
  have hC: C ⊆ G.sumset A ?B' and hCne: C ≠ {} and
    hCcard: card C + card (G.stabilizer C) ≥
      card (A ∩ ?B') + card (G.sumset (A ∪ ?B') (G.stabilizer C))
    using G.convergent-def hCconv by auto
  then have hCfinite: finite C using hC G.finite-sumset hA hB'

```

```

    by (meson finite-subset)
    have htranslate: card (G.sumset A {[0]}) + card (G.sumset ?B' {[0]}) -
card {[0]} ≤
    card (G.sumset A ?B')
  proof(cases G.stabilizer C = {[0]})
    case hCstab0: True
      have card (G.sumset A {[0]}) + card (G.sumset ?B' {[0]}) - card {[0]}
= card A + card ?B' -
      card {[0]} using hAG hB'G G.card-minusset' by fastforce
      also have ... = card (A ∩ ?B') + card (A ∪ ?B') - card {[0]}
      using hA hB' card-Un-Int by fastforce
      also have ... = card (A ∩ ?B') + card (G.sumset (A ∪ ?B') {[0]}) -
card {[0]}
      by (simp add: Int-absorb1 Int-commute hAG G.sumset-subset-carrier)
      also have ... ≤ card C using hCcard hCstab0 by auto
      also have ... ≤ card (G.sumset A ?B')
      using hC card-mono G.finite-sumset hA hB' by metis
      finally show ?thesis by simp
    next
      case hCstab-ne0: False
      have hCG: C ⊆ G using hC by (meson subset-trans G.sumset-subset-carrier)
      then have hstabC: finite (G.stabilizer C) using G.stabilizer-finite hCne
hC
      G.finite-sumset hA hB' by (metis Nat.add-0-right add-leE card.infinite
hCcard
      hintercard0 le-0-eq not-gr0)
      then have hcardstabC-gt-1: card (G.stabilizer C) > 1 using G.zero-mem-stabilizer
hCstab-ne0 hCG by (metis card-1-singletonE card-gt-0-iff diffs0-imp-equal
empty-iff
      gr-zeroI insertE less-one zero-less-diff)
      have G.sumset (G.stabilizer C) (G.sumset A ?B') ≠ G.sumset A ?B'
      using G.finite-sumset G.stabilizer-is-nonempty G.stabilizer-subset-group
      G.sumset-eq-sub-stabilizer G.sumset-subset-carrier hA hB' hCstab-ne0
hstab0'
      subset-singletonD by metis
      moreover have card (G.sumset (G.stabilizer C) (G.sumset A ?B')) ≥
card (G.sumset A ?B')
      using G.card-le-sumset G.finite-sumset hA hB' hstabC
      by (meson hCG G.sumset-subset-carrier G.unit-closed G.zero-mem-stabilizer)
      ultimately have ¬ G.sumset (G.stabilizer C) (G.sumset A ?B') ⊆
G.sumset A ?B'
      using G.finite-sumset hA hB' card-seteq by metis
      then obtain x where hx1: x ∈ G.sumset (G.stabilizer C) (G.sumset A
?B') and
      hx2: x ∉ G.sumset A ?B' by auto
      then obtain a1 b1 c where x = c [⊕] (a1 [⊕] b1) and c ∈ G.stabilizer
C and
      ha1A: a1 ∈ A and hb1B': b1 ∈ ?B' and ha1memG: a1 ∈ G and
hb1memG: b1 ∈ G

```

```

    by (metis (no-types, lifting) G.sumset.cases)
  then have hx:  $x \in G.\text{sumset } (G.\text{stabilizer } C) \{a1 \oplus b1\}$ 
    using hx1 by (meson G.composition-closed hAG hB'G insertI1
G.stabilizer-subset-group
  subsetD G.sumset.sumsetI)
  then have hnotsubAB:  $\neg G.\text{sumset } \{a1 \oplus b1\} (G.\text{stabilizer } C) \subseteq$ 
G.sumset A ?B'
    using hx2 G.sumset-commute by auto
  let ?A1 =  $A \cap (G.\text{sumset } \{a1\} (G.\text{stabilizer } C))$ 
  let ?A2 =  $A \cap (G.\text{sumset } \{b1\} (G.\text{stabilizer } C))$ 
  let ?B1 =  $?B' \cap (G.\text{sumset } \{b1\} (G.\text{stabilizer } C))$ 
  let ?B2 =  $?B' \cap (G.\text{sumset } \{a1\} (G.\text{stabilizer } C))$ 
  let ?C1 =  $C \cup (G.\text{sumset } ?A1 ?B1)$ 
  let ?C2 =  $C \cup (G.\text{sumset } ?A2 ?B2)$ 
  let ?H1 =  $G.\text{stabilizer } (G.\text{sumset } ?A1 ?B1)$ 
  let ?H2 =  $G.\text{stabilizer } (G.\text{sumset } ?A2 ?B2)$ 
  have hA1ne:  $?A1 \neq \{\}$  using ha1A G.zero-mem-stabilizer hCG
    by (metis (full-types) disjoint-iff-not-equal hAG insertCI G.right-unit
subset-eq
  G.sumset.sumsetI G.unit-closed)
  have hB1ne:  $?B1 \neq \{\}$  using hb1B' G.zero-mem-stabilizer hCG
    by (metis G.composition-closed disjoint-iff-not-equal insertCI G.left-unit

  G.sumset.cases G.sumset.sumsetI G.sumset-commute G.unit-closed)
  have hnotsubC:  $\neg G.\text{sumset } \{a1 \oplus b1\} (G.\text{stabilizer } C) \subseteq C$  using
hnotsubAB hC by blast
  have habstabempty:  $G.\text{sumset } \{a1 \oplus b1\} (G.\text{stabilizer } C) \cap C = \{\}$ 
  proof(rule ccontr)
    assume  $G.\text{sumset } \{a1 \oplus b1\} (G.\text{stabilizer } C) \cap C \neq \{\}$ 
    then obtain z where
      hz:  $G.\text{sumset } \{a1 \oplus b1\} (G.\text{stabilizer } C) \cap G.\text{sumset } \{z\} (G.\text{stabilizer }
C) \neq \{\}$  and
      hzC:  $z \in C$  using G.stabilizer-coset-Un hCG by blast
    then have  $G.\text{sumset } \{a1 \oplus b1\} (G.\text{stabilizer } C) = G.\text{sumset } \{z\}$ 
(G.stabilizer C) using hz
      G.sumset-stabilizer-eq-iff hCG G.sumset.simps hx by auto
    then have  $G.\text{sumset } \{a1 \oplus b1\} (G.\text{stabilizer } C) \subseteq C$  using hzC
      by (simp add: hCG G.stabilizer-coset-subset)
    thus False using hnotsubC by simp
  qed
  have hA1B1sub:  $G.\text{sumset } ?A1 ?B1 \subseteq G.\text{sumset } \{a1 \oplus b1\} (G.\text{stabilizer }
C)$  and
    hB2A2sub:  $G.\text{sumset } ?B2 ?A2 \subseteq G.\text{sumset } \{a1 \oplus b1\} (G.\text{stabilizer }
C)$ 
    using G.sumset-Inter-subset-sumset ha1memG hb1memG by auto
  then have hA1B1Cempty:  $G.\text{sumset } ?A1 ?B1 \cap C = \{\}$  using
habstabempty by blast
  then have heardC1:  $\text{card } ?C1 = \text{card } C + \text{card } (G.\text{sumset } ?A1 ?B1)$ 
using card-Un-disjoint

```


$G.\text{finite-sumset } hA \ hB' \ \text{finite-Int } hC \ \text{finite-subset Int-commute}$ **by** *metis*
have $hA1B1\text{cardless: card } (G.\text{sumset } ?A1 \ ?B1) < \text{card } (G.\text{sumset } A \ B)$
proof–
have $G.\text{sumset } ?A1 \ ?B1 \subseteq G.\text{sumset } A \ ?B'$ **using** $G.\text{sumset-mono}$ **by**
auto
moreover have $G.\text{sumset } ?A1 \ ?B1 \neq G.\text{sumset } A \ ?B'$
using $hA1B1\text{Cempty } hC \ hCne \ hA1B1\text{sub}$ **by** *auto*
ultimately show $\text{card } (G.\text{sumset } ?A1 \ ?B1) < \text{card } (G.\text{sumset } A \ B)$
using $G.\text{finite-sumset } hA \ hB' \ \text{psubset-card-mono psubset-eq 1}$ **by**
metis
qed
have $hB2A2\text{Cempty: } G.\text{sumset } ?B2 \ ?A2 \cap C = \{\}$ **using** $h\text{abstabempty}$
 $hB2A2\text{sub}$ **by** *blast*
then have $h\text{cardC2: card } ?C2 = \text{card } C + \text{card } (G.\text{sumset } ?B2 \ ?A2)$
using card-Un-disjoint
 $G.\text{finite-sumset } hA \ hB' \ \text{finite-Int } hC \ \text{finite-subset Int-commute}$
 $G.\text{sumset-commute}$ **by** *metis*
have $hA2B2\text{cardless: card } (G.\text{sumset } ?A2 \ ?B2) < \text{card } (G.\text{sumset } A \ B)$
proof–
have $G.\text{sumset } ?A2 \ ?B2 \subset G.\text{sumset } A \ ?B'$
using $G.\text{sumset-mono } hB2A2\text{Cempty } hC \ hCne \ hB2A2\text{sub } G.\text{sumset-commute}$
 psubset-eq
by (*metis Int-absorb1 Int-lower1*)
then show $?thesis$ **by** (*simp add: 1 G.finite-sumset hA hB' psubset-card-mono*)
qed
have $\text{card } ?A1 \leq \text{card } A$ **using** $\text{card-mono } hA$ **by** (*metis Int-lower1*)
then have $\text{card } (G.\text{sumset } ?A1 \ ?B1) + \text{card } ?A1 < \text{card } (G.\text{sumset } A \ B) + \text{card } A$
using $hA1B1\text{cardless}$ **by** *linarith*
then obtain l **where** $hln: l < n$ **and** $hind1: \text{card } (G.\text{sumset } ?A1 \ ?B1) + \text{card } ?A1 = l$
using $h\text{cardsum}$ **by** *auto*
have $\text{card } ?A2 \leq \text{card } A$ **using** $\text{card-mono } hA \ \text{Int-lower1}$ **by** *metis*
then have $\text{card } (G.\text{sumset } ?A2 \ ?B2) + \text{card } ?A2 < \text{card } (G.\text{sumset } A \ B) + \text{card } A$
using $hA2B2\text{cardless}$ **by** *linarith*
then obtain k **where** $hkn: k < n$ **and** $hind2: \text{card } (G.\text{sumset } ?A2 \ ?B2) + \text{card } ?A2 = k$
using $h\text{cardsum}$ **by** *auto*
have $hH1\text{stabC: } ?H1 \subset G.\text{stabilizer } C$ **using** $G.\text{stabilizer-sumset-psubset-stabilizer}$
 $hA1ne \ hB1ne \ ha1\text{memG} \ hb1\text{memG} \ h\text{notsubAB}$ **by** *presburger*
then have $\text{card } ?H1 < \text{card } (G.\text{stabilizer } C)$ **using** psubset-card-mono
 $h\text{stabC}$ **by** *auto*
moreover have $hC1H1: ?H1 = G.\text{stabilizer } ?C1$ **using** $G.\text{stabilizer-eq-stabilizer-union}$
by (*metis Int-commute hA hA1B1Cempty hA1ne hB' hB1ne hC hCfinite hCne ha1memG hb1memG hnotsubAB*)
ultimately have $hC1\text{notconv: } \neg G.\text{convergent } ?C1 \ A \ ?B'$ **using** $hC\text{min}$

G.convergent-set-def
le-antisym less-imp-le-nat less-not-refl2 **by** *fastforce*
have *hC1ne: ?C1 ≠ {}* **and** *hC2ne: ?C2 ≠ {}* **using** *hCne* **by** *auto*
have *hC1AB': ?C1 ⊆ G.sumset A ?B'* **and** *hC2AB': ?C2 ⊆ G.sumset*
A ?B' **using** *hC*
by (*auto simp add: G.sumset-mono*)
have *hA1G: ?A1 ⊆ G* **and** *hA1 : finite ?A1* **and** *hB1G: ?B1 ⊆ G* **and**
hB1: finite ?B1
using *hAG hB'G hA hB' finite-Int* **by** *auto*
then have *hindA1B1: card (G.sumset ?A1 ?H1) + card (G.sumset ?B1*
?H1) - card ?H1 ≤
card (G.sumset ?A1 ?B1) **using** *hindG hGroupG hA1ne hB1ne hind1*
hln hAG hB'G
hA hB' **by** *metis*
have *hC1notconv-ineq:*
(int (card ?C1) + card ?H1 - card (A ∩ ?B')) < int (card (G.sumset
(A ∪ ?B') ?H1))
using *hC1notconv hC1ne hC1AB' hC1H1 G.convergent-def* **by** *auto*
have *int (card (G.sumset (A ∪ ?B') (G.stabilizer C))) - card (G.sumset*
(A ∪ ?B') ?H1)
≤ (int (card C) + card (G.stabilizer C) - card (A ∩ ?B')) - card
(G.sumset (A ∪ ?B') ?H1)
using *hCcard* **by** *linarith*
then have *int (card (G.sumset (A ∪ ?B') (G.stabilizer C))) - card*
(G.sumset (A ∪ ?B') ?H1) <
(int (card C) + card (G.stabilizer C) - card (A ∩ ?B')) -
(int (card ?C1) + card ?H1 - card (A ∩ ?B')) **using** *hC1notconv-ineq*
by *linarith*
also have *... = int (card (G.stabilizer C)) - card ?H1 - card (G.sumset*
?A1 ?B1)
using *hcardC1* **by** *presburger*
also have *... ≤ int (card (G.stabilizer C)) - card (G.sumset ?A1 ?H1)*
- card (G.sumset ?B1 ?H1)
using *hindA1B1* **by** *linarith*

Finally, we deduce the inequality that is referred to as inequality (1) in [1] for $A \cap G.\text{sumset } \{a1\} (G.\text{stabilizer } C)$ and $G.\text{sumset } (G.\text{differenceset } B \{b\}) \{a\} \cap G.\text{sumset } \{b1\} (G.\text{stabilizer } C)$.

finally have *hA1B1ineq: int (card (G.sumset (A ∪ ?B') (G.stabilizer*
C))) -
card (G.sumset (A ∪ ?B') ?H1) < int (card (G.stabilizer C)) -
card (G.sumset ?A1 ?H1) - card (G.sumset ?B1 ?H1) **by** *simp*
have *hB2ne: ?B2 ≠ {}* **using** *G.sumset-inter-ineq hA1B1ineq ha1A ha1G*
hA hB' hAne hB'ne
hstabC hH1stabC ha1memG of-nat-0-le-iff **by** (*smt (verit, del-insts)*)
have *hA2ne: ?A2 ≠ {}* **using** *G.sumset-inter-ineq[of A b1 C ?B' a1]*
hA1B1ineq
hb1B' hb1memG hA hB' hAne hB'ne hstabC hH1stabC of-nat-0-le-iff
G.sumset-commute Un-commute **by** (*smt (verit, ccfv-SIG)*)

have $hH2stabC$: $?H2 \subseteq G.stabilizer\ C$ **using** $G.stabilizer-sumset-psubset-stabilizer$
 $G.commutative\ hA2ne\ hB2ne\ ha1memG\ hb1memG\ hnotsubAB$ **by**
presburger
then have $card\ ?H2 < card\ (G.stabilizer\ C)$ **using** $psubset-card-mono$
 $hstabC$ **by** *auto*
moreover have $hC2H2$: $?H2 = G.stabilizer\ ?C2$ **using** $G.stabilizer-eq-stabilizer-union$
by $(smt\ (verit,\ ccfv-threshold)\ G.sumset-commute\ Int-commute\ hA$
 $hA2ne\ hB'\ hB2A2Cempty$
 $hB2ne\ hC\ hCfinite\ hCne\ ha1memG\ hb1memG\ hnotsubAB)$
ultimately have $hC2notconv$: $\neg G.convergent\ ?C2\ A\ ?B'$
using $hCmin\ G.convergent-set-def\ le-antisym\ less-imp-le-nat\ less-not-refl2$

by *fastforce*
have $hA2G$: $?A2 \subseteq G$ **and** $hA2$: $finite\ ?A2$ **and** $hB2G$: $?B2 \subseteq G$ **and**
 $hB2$: $finite\ ?B2$
using $hAG\ hB'G\ hA\ hB'\ finite-Int$ **by** *auto*
then have $hindA2B2$: $card\ (G.sumset\ ?A2\ ?H2) + card\ (G.sumset\ ?B2$
 $?H2) - card\ ?H2 \leq$
 $card\ (G.sumset\ ?A2\ ?B2)$
using $hindG\ hGroupG\ hA2ne\ hB2ne\ hind2\ hkn\ hAG\ hB'G\ hA\ hB'$ **by**
metis
have $hC2notconv-ineq$:
 $(int\ (card\ ?C2) + card\ ?H2 - card\ (A \cap ?B')) < int\ (card\ (G.sumset$
 $(A \cup ?B')\ ?H2))$
using $hC2notconv\ hC2ne\ hC2AB'\ hC2H2\ G.convergent-def$ **by** *auto*
have $int\ (card\ (G.sumset\ (A \cup ?B')\ (G.stabilizer\ C))) - card\ (G.sumset$
 $(A \cup ?B')\ ?H2)$
 $\leq (int\ (card\ C) + card\ (G.stabilizer\ C) - card\ (A \cap ?B')) - card$
 $(G.sumset\ (A \cup ?B')\ ?H2)$
using $hCcard$ **by** *linarith*
then have $int\ (card\ (G.sumset\ (A \cup ?B')\ (G.stabilizer\ C))) - card$
 $(G.sumset\ (A \cup ?B')\ ?H2) <$
 $(int\ (card\ C) + card\ (G.stabilizer\ C) - card\ (A \cap ?B')) -$
 $(int\ (card\ ?C2) + card\ ?H2 - card\ (A \cap ?B'))$ **using** $hC2notconv-ineq$
by *linarith*
also have $\dots = int\ (card\ (G.stabilizer\ C)) - card\ ?H2 - card\ (G.sumset$
 $?A2\ ?B2)$
using $hcardC2\ G.sumset-commute$ **by** *simp*
also have $\dots \leq int\ (card\ (G.stabilizer\ C)) - card\ (G.sumset\ ?A2\ ?H2)$
 $- card\ (G.sumset\ ?B2\ ?H2)$
using $hindA2B2$ **by** *linarith*

Analogously, we deduce the inequality that is referred to as inequality (1) in [1] for $A \cap G.sumset\ \{b1\}\ (G.stabilizer\ C)$ and $G.sumset\ (G.differenceset\ B\ \{b\})\ \{a\} \cap G.sumset\ \{a1\}\ (G.stabilizer\ C)$.

finally have $hA2B2ineq$: $int\ (card\ (G.sumset\ (A \cup ?B')\ (G.stabilizer$
 $C))) -$
 $card\ (G.sumset\ (A \cup ?B')\ ?H2) < int\ (card\ (G.stabilizer\ C)) -$

$\text{card } (G.\text{sumset } ?A2 \ ?H2) - \text{card } (G.\text{sumset } ?B2 \ ?H2)$ **by** *simp*
have $\text{hfinsumH2}:\text{finite } (G.\text{sumset } (A \cup ?B') \ ?H2)$
using $G.\text{finite-sumset } hA \ hB' \ \text{finite-UnI } \text{hstabC } \text{hH2stabC } \text{finite-subset}$
psubset-imp-subset
by *metis*
have $\text{hsubsumH2}:\text{finite } (G.\text{sumset } (A \cup ?B') \ ?H2) \subseteq G.\text{sumset } (A \cup ?B')$
 $(G.\text{stabilizer } C)$
using $G.\text{sumset.cases } hAG \ hB'G \ G.\text{stabilizer-subset-group } \text{hH2stabC}$
psubset-imp-subset
by (*smt* (*verit*, *best*) *subset-Un-eq* $G.\text{sumset-commute } G.\text{sumset-subset-Un1}$)
then **have** $\text{hsumH2card-le}:\text{card } (G.\text{sumset } (A \cup ?B') \ ?H2) \leq$
 $\text{card } (G.\text{sumset } (A \cup ?B') \ (G.\text{stabilizer } C))$
using $\text{card-mono } G.\text{finite-sumset } G.\text{stabilizer-finite } hC \ hCne \ hCG \ hA$
 hB'
by (*metis* *finite-UnI* hstabC)
have $\text{hfinsumH1}:\text{finite } (G.\text{sumset } (A \cup ?B') \ ?H1)$
using $\text{finite-sumset } \text{finite-Un } \text{psubsetE}$ **by** (*metis* $G.\text{finite-sumset}$
 $G.\text{stabilizer-finite}$
 $G.\text{sumset-subset-carrier } \text{Int-Un-eq}(4) \ hA \ hA1 \ hB' \ hB1 \ hC1H1 \ \text{hH1stabC}$
 habstabempty)
have $\text{hsubsumH1}:\text{finite } (G.\text{sumset } (A \cup ?B') \ ?H1) \subseteq G.\text{sumset } (A \cup ?B')$
 $(G.\text{stabilizer } C)$
using $G.\text{sumset.cases } \text{psubsetE } \text{subset-refl } G.\text{sumset-mono } \text{hH1stabC}$
by *simp*
have $\text{hsumH1card-le}:\text{card } (G.\text{sumset } (A \cup ?B') \ ?H1) \leq \text{card } (G.\text{sumset}$
 $(A \cup ?B') \ (G.\text{stabilizer } C))$
using $\text{card-mono } G.\text{finite-sumset } G.\text{stabilizer-finite}$ **by** (*metis* *finite-UnI*
 $hA \ hB' \ \text{hstabC } \text{hsubsumH1}$)
have $\text{ha1b1stabCne}:\text{card } (G.\text{sumset } \{a1\} \ (G.\text{stabilizer } C)) \neq \text{card } (G.\text{sumset } \{b1\}$
 $(G.\text{stabilizer } C))$
proof
assume $\text{ha1b1}:\text{card } (G.\text{sumset } \{a1\} \ (G.\text{stabilizer } C)) = \text{card } (G.\text{sumset } \{b1\}$
 $(G.\text{stabilizer } C))$
have $\text{hfin}:\text{finite } (G.\text{sumset } ?A1 \ ?H1 \cup G.\text{sumset } ?B1 \ ?H1)$
using $\text{finite-UnI } G.\text{finite-sumset } hA1 \ hB1 \ \text{hH1stabC } \text{hstabC}$ *psub-*
set-imp-subset
by (*metis* *finite-subset*)
have $\text{int } (\text{card } (G.\text{sumset } \{a1\} \ (G.\text{stabilizer } C))) - \text{card } (G.\text{sumset}$
 $?A1 \ ?H1) -$
 $\text{card } (G.\text{sumset } ?B1 \ ?H1) \leq \text{int } (\text{card } (G.\text{sumset } \{a1\} \ (G.\text{stabilizer}$
 $C))) -$
 $\text{card } (G.\text{sumset } ?A1 \ ?H1 \cup G.\text{sumset } ?B1 \ ?H1)$
using card-Un-le [of $G.\text{sumset } ?A1 \ ?H1 \ G.\text{sumset } ?B1 \ ?H1$] **by** *linarith*
also **have** $\dots \leq \text{card } (G.\text{sumset } \{a1\} \ (G.\text{stabilizer } C)) - (G.\text{sumset}$
 $?A1 \ ?H1 \cup G.\text{sumset } ?B1 \ ?H1)$
using $\text{diff-card-le-card-Diff}$ [of $G.\text{sumset } ?A1 \ ?H1 \cup G.\text{sumset } ?B1$
 $?H1$
 $G.\text{sumset } \{a1\} \ (G.\text{stabilizer } C)$] *hfin* **by** *linarith*
also **have** $\dots \leq \text{card } (G.\text{sumset } \{a1\} \ (G.\text{stabilizer } C)) - \text{card } (G.\text{sumset } (?A1$

$\cup ?B1) ?H1)$
using $G.sumset-subset-Un1$ **by** $auto$
also have $\dots \leq \text{card } (G.sumset (A \cup ?B') (G.stabilizer C) - G.sumset (A \cup ?B') ?H1)$
proof–
have $hsub: G.sumset \{a1\} (G.stabilizer C) - G.sumset (?A1 \cup ?B1)$
 $?H1 \subseteq$
 $G.sumset (A \cup ?B') (G.stabilizer C) - G.sumset (A \cup ?B') ?H1$
proof
fix x **assume** $hx: x \in G.sumset \{a1\} (G.stabilizer C) - G.sumset (?A1 \cup ?B1) ?H1$
then obtain c **where** $hxac: x = a1 [\oplus] c$ **and** $hc: c \in G.stabilizer C$
using $G.sumset.cases$ **by** $blast$
then have $x \in G.sumset (A \cup ?B') (G.stabilizer C)$ **using** $ha1A$
 hAG
 $G.stabilizer-subset-group$ **by** $(simp \text{ add: subset-iff } G.sumset.sumsetI)$
moreover have $x \notin G.sumset (A \cup ?B') ?H1$
proof
assume $hx1: x \in G.sumset (A \cup ?B') ?H1$
then obtain $y \ h1$ **where** $hxy: x = y [\oplus] h1$ **and** $hy: y \in A \cup ?B'$
and
 $hh1: h1 \in ?H1$ **using** $G.sumset.cases$ **by** $blast$
then have $hyG: y \in G$ **and** $hcG: c \in G$ **and** $hh1G: h1 \in G$
using $hAG \ hB'G \ G.stabilizer-subset-group \ hc$ **by** $auto$
then have $y = a1 [\oplus] (c [\oplus] G.inverse \ h1)$ **using** $hxac \ hxy \ ha1A$
 hAG
by $(metis \ G.associative \ G.commutative \ G.composition-closed \ in-mono \ G.inverse-closed \ G.invertible \ G.invertible-left-inverse2)$
moreover have $h1 \in G.stabilizer C$ **using** $hh1 \ hH1stabC$ **by** $auto$
moreover hence $c [\oplus] G.inverse \ h1 \in G.stabilizer C$ **using** $hc \ G.stabilizer-is-subgroup \ subgroup-def \ G.group-axioms \ group.invertible \ subgroup.subgroup-inverse-iff \ submonoid.sub-composition-closed \ hh1G$ **by** $metis$
ultimately have $y \in G.sumset \{a1\} (G.stabilizer C)$ **using** $ha1G$
 $hcG \ hAG \ ha1A \ hh1G$
by $blast$
then have $y \in ?A1 \cup ?B1$ **using** hy **by** $(simp \text{ add: } ha1b1)$
thus $False$ **using** $hx \ hxy \ hh1 \ hh1G \ hyG$ **by** $auto$
qed
ultimately show $x \in G.sumset (A \cup ?B') (G.stabilizer C) - G.sumset (A \cup ?B') ?H1$ **by** $simp$
qed
moreover have $finite (G.sumset \{a1\} (G.stabilizer C) - G.sumset (?A1 \cup ?B1) ?H1)$
using $finite-subset \ G.finite-sumset \ hstabC$ **by** $simp$
moreover hence $finite (G.sumset (A \cup ?B') (G.stabilizer C) - G.sumset (A \cup ?B') ?H1)$

using *finite-subset* G .*finite-sumset* *hstabC* hA hB' *finite-UnI* **by**
simp
moreover have $\text{card } (G.\text{sumset } \{a1\} (G.\text{stabilizer } C)) - G.\text{sumset}$
 $(?A1 \cup ?B1) ?H1) \leq$
 $\text{card } (G.\text{sumset } (A \cup ?B') (G.\text{stabilizer } C)) - G.\text{sumset } (A \cup ?B')$
 $?H1)$
using *card-mono* *hsub* *calculation(3)* **by** *auto*
ultimately show *?thesis* **using** *card-Diff-subset* **by** *linarith*
qed
also have $\dots = \text{int } (\text{card } (G.\text{sumset } (A \cup ?B') (G.\text{stabilizer } C))) -$
 $\text{card } (G.\text{sumset } (A \cup ?B') ?H1)$
using *card-Diff-subset[OF hfinsumH1 hsubsumH1] hsumH1card-le* **by**
linarith
finally have $\text{int } (\text{card } (G.\text{stabilizer } C)) - \text{card } (G.\text{sumset } ?A1 ?H1)$
 $- \text{card } (G.\text{sumset } ?B1 ?H1)$
 $\leq \text{int } (\text{card } (G.\text{sumset } (A \cup ?B') (G.\text{stabilizer } C))) - \text{card } (G.\text{sumset}$
 $(A \cup ?B') ?H1)$
using *hCG subset-iff* G .*card-sumset-singleton-subset-eq* G .*stabilizer-subset-group*

 hAG $ha1A$ **by** *auto*
thus *False* **using** *hA1B1ineq* **by** *linarith*
qed
have $\text{int } (\text{card } (G.\text{sumset } (A \cup ?B') (G.\text{stabilizer } C))) -$
 $\text{card } (G.\text{sumset } (A \cup ?B') ?H1) \geq 0$ **by** (*simp add: hsumH1card-le*)
then have $\text{int } (\text{card } (G.\text{stabilizer } C)) -$
 $\text{card } (G.\text{sumset } ?A1 ?H1) - \text{card } (G.\text{sumset } ?B1 ?H1) > 0$ **using**
 $hA1B1ineq$ **by** *linarith*
moreover have $\text{int } (\text{card } ?H1) \text{ dvd } \text{int } (\text{card } (G.\text{stabilizer } C)) -$
 $\text{card } (G.\text{sumset } ?A1 ?H1) - \text{card } (G.\text{sumset } ?B1 ?H1)$ **using**
 G .*stabilizer-subset-stabilizer-dvd* $hH1stabC$ *psubset-imp-subset int-dvd-int-iff*
dvd-diff
 G .*card-stabilizer-divide-sumset[OF hA1G]* G .*card-stabilizer-divide-sumset[OF*
 $hB1G]$
by *fastforce*
ultimately have $\text{int } (\text{card } (G.\text{stabilizer } C)) -$
 $\text{card } (G.\text{sumset } ?A1 ?H1) - \text{card } (G.\text{sumset } ?B1 ?H1) \geq \text{int } (\text{card}$
 $?H1)$
using *zdvd-imp-le* **by** *blast*
moreover have $hA1-le-sum: \text{card } ?A1 \leq \text{card } (G.\text{sumset } ?A1 ?H1)$
using G .*sumset-commute* G .*card-le-sumset* G .*zero-mem-stabilizer* $hA1G$
 $hA1$ $hH1stabC$ *hstabC*
by (*metis finite-subset psubset-imp-subset G.unit-closed*)
moreover have $hB1-le-sum: \text{card } ?B1 \leq \text{card } (G.\text{sumset } ?B1 ?H1)$
using G .*sumset-commute* G .*card-le-sumset* G .*zero-mem-stabilizer* $hB1G$
 $hB1$ $hH1stabC$ *hstabC*
by (*metis finite-subset psubset-imp-subset G.unit-closed*)

The above facts combined allow us to deduce the inequality that is referred to as inequality (2) in [1] for $A \cap G.\text{sumset } \{a1\} (G.\text{stabilizer } C)$, $G.\text{sumset } (G.\text{differenceset } B \{b\}) \{a\} \cap G.\text{sumset } \{b1\} (G.\text{stabilizer } C)$ and

$G.stabilizer (G.sumset (A \cap G.sumset \{a1\} (G.stabilizer C)) (G.sumset (G.differenceset B \{b\}) \{a\} \cap G.sumset \{b1\} (G.stabilizer C)))$.

ultimately have $21: int (card (G.stabilizer C)) \geq int (card ?H1) + card ?A1 + card ?B1$
by *linarith*
have $int (card (G.sumset (A \cup ?B') (G.stabilizer C))) - card (G.sumset (A \cup ?B') ?H2) \geq 0$ **by** (*simp add: hsumH2card-le*)
then have $int (card (G.stabilizer C)) - card (G.sumset ?A2 ?H2) - card (G.sumset ?B2 ?H2) > 0$ **using** *hA2B2ineq* **by** *linarith*
moreover have $int (card ?H2) dvd int (card (G.stabilizer C)) - card (G.sumset ?A2 ?H2) - card (G.sumset ?B2 ?H2)$ **using** *psubset-imp-subset*
 $G.stabilizer-subset-stabilizer-dvd$ *hH2stabC int-dvd-int-iff dvd-diff*
 $G.card-stabilizer-divide-sumset[OF hA2G]$ $G.card-stabilizer-divide-sumset[OF hB2G]$
by *fastforce*
ultimately have $int (card (G.stabilizer C)) - card (G.sumset ?A2 ?H2) - card (G.sumset ?B2 ?H2) \geq int (card ?H2)$
using *zdvd-imp-le* **by** *blast*
moreover have $hA2-le-sum: card ?A2 \leq card (G.sumset ?A2 ?H2)$
using $G.sumset-commute$ $G.card-le-sumset$ $G.zero-mem-stabilizer$ $G.stabilizer-subset-group$
 $hA2G$ $hA2$ $hH2stabC$ $hstabC$ *psubset-imp-subset* **by** (*metis finite-subset G.unit-closed*)
moreover have $hB2-le-sum: card ?B2 \leq card (G.sumset ?B2 ?H2)$
using $G.sumset-commute$ $G.card-le-sumset$ $G.zero-mem-stabilizer$ $G.stabilizer-subset-group$
 $hB2G$ $hB2$ $hH2stabC$ $hstabC$ *psubset-imp-subset* **by** (*metis finite-subset G.unit-closed*)

Analogously, the above facts combined allow us to deduce the inequality that is referred to as inequality (2) in [1] for $A \cap G.sumset \{b1\} (G.stabilizer C)$, $G.sumset (G.differenceset B \{b\}) \{a\} \cap G.sumset \{a1\} (G.stabilizer C)$ and $G.stabilizer (G.sumset (A \cap G.sumset \{b1\} (G.stabilizer C)) (G.sumset (G.differenceset B \{b\}) \{a\} \cap G.sumset \{a1\} (G.stabilizer C)))$.

ultimately have $22: int (card (G.stabilizer C)) \geq int (card ?H2) + card ?A2 + card ?B2$
by *linarith*
let $?S = G.sumset \{a1\} (G.stabilizer C) - (?A1 \cup ?B2)$
let $?T = G.sumset \{b1\} (G.stabilizer C) - (?A2 \cup ?B1)$
have $hS : finite ?S$ **and** $hT : finite ?T$ **using** $G.finite-sumset$ $hstabC$ **by** *auto*
have $hST: ?S \cap ?T = \{\}$ **using** $ha1b1stabCne$ $Diff-Int2$ $Diff-Int-distrib2$ $Int-Diff$ $Int-Un-eq(4)$
 $Int-absorb$ $Int-commute$ $Int-empty-right$ $Int-insert-right$ $Un-empty$ $empty-Diff$ $hA1ne$

$hA2ne$ $G.sumset-commute$ $G.sumset-is-empty-iff$ $G.sumset-stabilizer-eq-iff$
by (smt ($verit$, $ccfv-threshold$) $G.sumset-assoc$)
have $hSTcard-le$: $card\ ?S + card\ ?T + card\ (G.sumset\ (A \cup\ ?B')\ \{[0]\})$
 \leq
 $card\ (G.sumset\ (A \cup\ ?B')\ (G.stabilizer\ C))$
proof-
have $G.sumset\ \{a1\}\ (G.stabilizer\ C) \subseteq G.sumset\ (A \cup\ ?B')\ (G.stabilizer$
 $C)$ **and**
 $G.sumset\ \{b1\}\ (G.stabilizer\ C) \subseteq G.sumset\ (A \cup\ ?B')\ (G.stabilizer$
 $C)$
using $G.sumset-mono$ $ha1A$ $hb1B'$ $subset-refl$ $empty-subsetI$ $insert-subset$
by $auto$
moreover **have** $(G.sumset\ (A \cup\ ?B')\ \{[0]\}) \subseteq G.sumset\ (A \cup\ ?B')$
 $(G.stabilizer\ C)$
using $G.sumset-mono$ $subset-refl$ $empty-subsetI$ $insert-subset$
 $G.zero-mem-stabilizer$
by $metis$
ultimately **have** $hsub$: $?S \cup\ ?T \cup\ (G.sumset\ (A \cup\ ?B')\ \{[0]\}) \subseteq$
 $G.sumset\ (A \cup\ ?B')\ (G.stabilizer\ C)$ **by** $blast$
have $?S \cap G.sumset\ (A \cup\ ?B')\ \{[0]\} = \{\}$ **by** $auto$
moreover **have** $(?S \cup\ ?T) \cap G.sumset\ (A \cup\ ?B')\ \{[0]\} = \{\}$ **by** $auto$
ultimately **have** $card\ ?S + card\ ?T + card\ (G.sumset\ (A \cup\ ?B')$
 $\{[0]\}) =$
 $card\ (?S \cup\ ?T \cup\ (G.sumset\ (A \cup\ ?B')\ \{[0]\}))$ **using** $card-Un-disjoint$
 hS hT
 $G.finite-sumset$ $finite-UnI$ hA hB' hST **by** ($metis$ $finite.emptyI$
 $finite.insertI$)
also **have** $\dots \leq card\ (G.sumset\ (A \cup\ ?B')\ (G.stabilizer\ C))$ **using**
 $card-mono$
 $G.finite-sumset$ $finite-UnI$ hA hB' $hstabC$ $hsub$ **by** $metis$
finally **show** $?thesis$ **by** $simp$
qed
have $hAB-not-conv$: $\neg G.convergent\ (G.sumset\ A\ ?B')\ A\ ?B'$ **using**
 $hCmin$ $hstab0$
 $G.convergent-set-def$ $hcardstabC-gt-1$ $hstab0'$ **by** $fastforce$
then **have** $card\ (G.sumset\ A\ ?B') + card\ \{[0]\} < card\ (A \cap\ ?B') +$
 $card\ (G.sumset\ (A \cup\ ?B')\ \{[0]\})$ **using** $G.convergent-def$ $hAne$ $hB'ne$
 hAG $hB'G$ $hstab0'$
 $subset-refl$ **by** $auto$
then **have** $hAB'sum$: $int\ (card\ (G.sumset\ (A \cup\ ?B')\ \{[0]\})) + card\ (A$
 $\cap\ ?B') -$
 $card\ (G.sumset\ A\ ?B') > 1$ **by** $simp$
moreover **have** $int\ (card\ ?A1) + card\ ?B1 \leq int\ (card\ (G.sumset\ ?A1$
 $?H1)) +$
 $card\ (G.sumset\ ?B1\ ?H1)$ **using** $hA1-le-sum$ $hB1-le-sum$ **by** $linarith$
moreover **hence** $int\ (card\ ?A1) + card\ ?B1 - card\ ?H1 \leq card$
 $(G.sumset\ ?A1\ ?B1)$
using $hindA1B1$ **by** $linarith$
ultimately **have** $int\ (card\ ?S) + card\ ?T + card\ ?A1 + card\ ?B1 -$

$\text{card } ?H1 <$
 $\text{int } (\text{card } ?S) + \text{card } ?T + \text{card } (G.\text{sumset } (A \cup ?B') \{[0]\}) + \text{card } (A \cap ?B') -$
 $\text{card } (G.\text{sumset } A ?B') + \text{card } (G.\text{sumset } ?A1 ?B1)$ **by** *linarith*
also have $\dots \leq \text{int } (\text{card } (G.\text{sumset } (A \cup ?B') (G.\text{stabilizer } C))) + \text{card } (A \cap ?B') - \text{card } C$
proof–
have $G.\text{sumset } ?A1 ?B1 \cup C \subseteq G.\text{sumset } A ?B'$ **using** hC *G.sumset-mono*
by *auto*
then have $\text{card } (G.\text{sumset } ?A1 ?B1) + \text{card } C \leq$
 $\text{card } (G.\text{sumset } A ?B')$ **using** *card-Un-disjoint hCfinite G.finite-sumset*
 $hA1 hB1 hA1B1Cempty$
card-mono hA hB' **by** *metis*
then show *?thesis* **using** *hSTcard-le* **by** *linarith*
qed

From this, inequality (3) [1] follows for $A \cap G.\text{sumset } \{a1\} (G.\text{stabilizer } C)$, $G.\text{sumset } (G.\text{differenceset } B \{b\}) \{a\} \cap G.\text{sumset } \{b1\} (G.\text{stabilizer } C)$ and $G.\text{stabilizer } (G.\text{sumset } (A \cap G.\text{sumset } \{a1\} (G.\text{stabilizer } C)) (G.\text{sumset } (G.\text{differenceset } B \{b\}) \{a\} \cap G.\text{sumset } \{b1\} (G.\text{stabilizer } C)))$.

finally have $31: \text{int } (\text{card } ?S) + \text{card } ?T + \text{card } ?A1 + \text{card } ?B1 -$
 $\text{card } ?H1 <$
 $\text{card } (G.\text{stabilizer } C)$ **using** $hCcard$ **by** *linarith*
have $\text{int } (\text{card } ?A2) + \text{card } ?B2 \leq \text{int } (\text{card } (G.\text{sumset } ?A2 ?H2)) +$
 $\text{card } (G.\text{sumset } ?B2 ?H2)$ **using** *hA2-le-sum hB2-le-sum* **by** *linarith*
moreover hence $\text{int } (\text{card } ?A2) + \text{card } ?B2 - \text{card } ?H2 \leq \text{card } (G.\text{sumset } ?A2 ?B2)$
using *hindA2B2* **by** *linarith*
ultimately have $\text{int } (\text{card } ?S) + \text{card } ?T + \text{card } ?A2 + \text{card } ?B2 -$
 $\text{card } ?H2 <$
 $\text{int } (\text{card } ?S) + \text{card } ?T + \text{card } (G.\text{sumset } (A \cup ?B') \{[0]\}) + \text{card } (A \cap ?B') -$
 $\text{card } (G.\text{sumset } A ?B') + \text{card } (G.\text{sumset } ?A2 ?B2)$ **using** *hAB'sum*
by *linarith*
also have $\dots \leq \text{int } (\text{card } (G.\text{sumset } (A \cup ?B') (G.\text{stabilizer } C))) + \text{card } (A \cap ?B') - \text{card } C$
proof–
have $G.\text{sumset } ?A2 ?B2 \cup C \subseteq G.\text{sumset } A ?B'$ **using** hC *G.sumset-mono*
by *auto*
then have $\text{card } (G.\text{sumset } ?A2 ?B2) + \text{card } C \leq$
 $\text{card } (G.\text{sumset } A ?B')$ **using** *card-Un-disjoint hCfinite G.finite-sumset*
 $hA2 hB2 hA1B1Cempty$
card-mono hA hB' hB2A2Cempty G.sumset-commute **by** *metis*
then show *?thesis* **using** *hSTcard-le* **by** *linarith*
qed

From this, inequality (3) [1] follows for $A \cap G.\text{sumset } \{b1\} (G.\text{stabilizer } C)$, $G.\text{sumset } (G.\text{differenceset } B \{b\}) \{a\} \cap G.\text{sumset } \{a1\} (G.\text{stabilizer } C)$ and $G.\text{stabilizer } (G.\text{sumset } (A \cap G.\text{sumset } \{b1\} (G.\text{stabilizer } C)) (G.\text{sumset } (G.\text{differenceset } B \{b\}) \{a\} \cap G.\text{sumset } \{a1\} (G.\text{stabilizer } C)))$.

$(G.differenceset\ B\ \{b\})\ \{a\} \cap G.sumset\ \{a1\}\ (G.stabilizer\ C))$.

finally have $32: int\ (card\ ?S) + card\ ?T + card\ ?A2 + card\ ?B2 - card\ ?H2 < card\ (G.stabilizer\ C)$ **using** $hCcard$ **by** $linarith$

Adding together the four inequalities obtained by versions of inequalities (2) and (3) and dividing by 2 gives the following inequality:

have $4: 2 * int\ (card\ (G.stabilizer\ C)) > int\ (card\ ?A1) + card\ ?B2 + card\ ?S + card\ ?T + card\ ?B1 + card\ ?A2$
using $21\ 22\ 31\ 32$ **by** $linarith$

have $G.sumset\ \{a1\}\ (G.stabilizer\ C) = (?S \cup ?A1) \cup ?B2$ **and**
 $hb1T: G.sumset\ \{b1\}\ (G.stabilizer\ C) = (?T \cup ?A2) \cup ?B1$ **by** $auto$
then have $card\ (G.stabilizer\ C) \leq card\ ?S + card\ ?A1 + card\ ?B2$
using $card-Un-le[of\ ?S\ ?A1]\ card-Un-le[of\ ?S \cup ?A1\ ?B2]$
 $G.card-sumset-singleton-subset-eq\ G.stabilizer-subset-group\ ha1A\ hAG$

by $auto$

moreover have $card\ (G.stabilizer\ C) \leq card\ ?T + card\ ?A2 + card\ ?B1$

using $hb1T\ card-Un-le[of\ ?T\ ?A2]\ card-Un-le[of\ ?T \cup ?A2\ ?B1]$
 $G.card-sumset-singleton-subset-eq\ G.stabilizer-subset-group\ hb1B'\ hb'G$

by $auto$

Combining the inequality labelled $4::'b$ above with the above facts, the claim follows:

ultimately show $?thesis$ **using** 4 **by** $linarith$
qed

It remains to transfer the statement of inequality labelled $1::'b$ into an analogous one, which replaces $G.sumset\ (G.differenceset\ B\ \{b\})\ \{a\}$ with B .

have $2: card\ (G.sumset\ A\ (G.stabilizer\ (G.sumset\ A\ B))) = card\ (G.sumset\ A\ (G.stabilizer\ (G.sumset\ A\ ?B')))$
using $hstabeq$ **by** $auto$

have $3: card\ (G.sumset\ B\ (G.stabilizer\ (G.sumset\ A\ B))) = card\ (G.sumset\ ?B'\ (G.stabilizer\ (G.sumset\ A\ ?B')))$ **using** $hstabeq$
 $hstab0'\ G.sumset-commute$

by $(metis\ G.card-differenceset-singleton-mem-eq\ G.card-sumset-singleton-subset-eq$

$G.sumset-D(1)\ G.sumset-commute\ G.sumset-subset-carrier\ Int-absorb1\ Int-commute\ hBG\ ha1G\ hbG)$

then show $?thesis$ **using** $1\ 2\ 3\ hstabeq\ hstab0'\ htranslate$ **by** $auto$
qed

next

case $hstabne0: False$

let $?K = G.stabilizer\ (G.sumset\ A\ B)$

have $hcardK-gt-1: card\ ?K > 1$ **using** $G.stabilizer-finite\ G.sumset-subset-carrier\ G.finite-sumset$

$hA\ hB\ hstabne0\ G.zero-mem-stabilizer$ **by** $(metis\ card-0-eq\ card-1-singletonE\ G.card-sumset-0-iff$

```

empty-iff hAG hAne hBG hBne insertE less-one linorder-neqE-nat)
interpret K: subgroup-of-additive-abelian-group ?K G ([⊕]) [0]
using G.stabilizer-is-subgroup subgroup-of-additive-abelian-group-def
by (metis G.abelian-group-axioms G.group-axioms hGroupG subgroup-of-abelian-group-def

subgroup-of-group.intro)
let ?φ = K.Class
have hφ:  $\bigwedge a. a \in G \implies ?\phi a = (\lambda x. G.sumset \{x\} ?K) a$ 
using K.Left-Coset-Class-unit G.Left-Coset-eq-sumset-stabilizer by simp
interpret GK: additive-abelian-group G.Factor-Group G ?K K.quotient-composition
K.Class [0]
proof
fix x y assume x ∈ K.Partition and y ∈ K.Partition
then obtain a b where x = K.Class a and y = K.Class b and a ∈ G
and b ∈ G
by (meson K.representant-exists)
then show K.quotient-composition x y = K.quotient-composition y x
using K.Class-commutes-with-composition G.commutative by presburger
qed
have hGroupGK: additive-abelian-group (G.Factor-Group G ?K) K.quotient-composition
(K.Class [0]) ..

```

Here, we specialize the induction hypothesis to the factor group.

```

let ?K-repr = K.φ ‘ K.Partition
have  $\bigwedge x y. x \in ?K-repr \implies y \in ?K-repr \implies K.quot-comp-alt x y =$ 
K.quot-comp-alt y x
using K.quot-comp-alt-def G.commutative K.phi-image-subset subsetD by
(metis (full-types))
then interpret K-repr: additive-abelian-group ?K-repr K.quot-comp-alt K.φ
?K
using Group-Theory.group.axioms(1)[OF K.phi-image-group]
by (auto simp add: additive-abelian-group-def abelian-group-def K.phi-image-group
commutative-monoid-axioms-def commutative-monoid-def)
have hindrepr:  $\bigwedge m C D. m < n \longrightarrow C \subseteq ?K-repr \longrightarrow D \subseteq ?K-repr \longrightarrow$ 
finite C  $\longrightarrow$ 
finite D  $\longrightarrow C \neq \{\}$   $\longrightarrow D \neq \{\}$   $\longrightarrow \text{card } (K-repr.sumset C D) + \text{card } C$ 
 $= m \longrightarrow$ 
card (K-repr.sumset C (K-repr.stabilizer (K-repr.sumset C D))) +
card (K-repr.sumset D (K-repr.stabilizer (K-repr.sumset C D))) - card
(K-repr.stabilizer (K-repr.sumset C D))  $\leq$ 
card (K-repr.sumset C D) using hind K-repr.additive-abelian-group-axioms
by blast
have hindfactor:  $\bigwedge m C D. m < n \longrightarrow C \subseteq K.Partition \longrightarrow D \subseteq$ 
K.Partition  $\longrightarrow$  finite C  $\longrightarrow$ 
finite D  $\longrightarrow C \neq \{\}$   $\longrightarrow D \neq \{\}$   $\longrightarrow \text{card } (GK.sumset C D) + \text{card } C =$ 
m  $\longrightarrow$ 
card (GK.sumset C (GK.stabilizer (GK.sumset C D))) +
card (GK.sumset D (GK.stabilizer (GK.sumset C D))) - card (GK.stabilizer
(GK.sumset C D))  $\leq$ 

```

$card (GK.sumset C D)$
proof(*intro impI*)
fix $m C D$ **assume** $hmn: m < n$ **and** $hCK: C \subseteq K.Partition$ **and** $hDK: D \subseteq K.Partition$ **and**
 $hC: finite C$ **and** $hD: finite D$ **and** $hCne: C \neq \{\}$ **and** $hDne: D \neq \{\}$ **and**
 $hCDcard: card (GK.sumset C D) + card C = m$
let $?C = K.\varphi ' C$ **and** $?D = K.\varphi ' D$
have $hCrepr: ?C \subseteq ?K-repr$ **and** $hDrepr: ?D \subseteq ?K-repr$ **using** $hCK hDK$
by *auto*
have $hCfin: finite ?C$ **and** $hDfin: finite ?D$ **and** $hCne-1: ?C \neq \{\}$ **and**
 $hDne-1: ?D \neq \{\}$ **using** $hC hD hCne hDne$ **by** *auto*
have $hcardC: card ?C = card C$ **using** $K.phi-inj-on hC card-image$
inj-on-subset hCK **by** *metis*
have $card (GK.sumset C D) = card (K-repr.sumset ?C ?D)$
using $card-image K.phi-inj-on inj-on-subset K.phi-image-sumset-eq$
 $GK.sumset-subset-carrier hCK hDK$ **by** (*smt (verit, best)*)
then **have** $card (K-repr.sumset ?C ?D) + card ?C = m$ **using** $hCDcard$
 $hcardC$ **by** *presburger*
then **have** $card (K-repr.sumset ?C (K-repr.stabilizer (K-repr.sumset ?C$
 $?D))) +$
 $card (K-repr.sumset ?D (K-repr.stabilizer (K-repr.sumset ?C ?D))) - card$
 $(K-repr.stabilizer (K-repr.sumset ?C ?D)) \leq$
 $card (K-repr.sumset ?C ?D)$
using $hindrepr hCfin hDfin hCne-1 hDne-1 hCrepr hDrepr hmn$ **by** *blast*
then **show** $card (GK.sumset C (GK.stabilizer (GK.sumset C D))) +$
 $card (GK.sumset D (GK.stabilizer (GK.sumset C D))) - card (GK.stabilizer$
 $(GK.sumset C D)) \leq$
 $card (GK.sumset C D)$ **using** $K.phi-image-sumset-eq K.phi-image-stabilizer-eq$
 $K.phi-inj-on inj-on-subset hCK hDK card-image$
by (*smt (z3) GK.stabilizer-subset-group GK.sumset-subset-carrier*)
qed
have $hstab0: GK.stabilizer (?\varphi ' (G.sumset A B)) = \{K.Class [0]\}$
proof
show $GK.stabilizer (?\varphi ' G.sumset A B) \subseteq \{K.Class [0]\}$
proof
fix x **assume** $hx: x \in GK.stabilizer (?\varphi ' G.sumset A B)$
moreover **have** $?\varphi ' G.sumset A B \subseteq K.Partition$
using $K.natural.map-closed G.sumset-subset-carrier$ **by** *blast*
ultimately **have** $hsum: GK.sumset \{x\} (?\varphi ' G.sumset A B) = ?\varphi ' G.sumset A B$
using $GK.stabilizer-def$ **by** *auto*
obtain x' **where** $hx\varphi: x = ?\varphi x'$ **and** $hx'G: x' \in G$
using $hx GK.stabilizer-subset-group K.representant-exists$ **by** *force*
have $hsumset: GK.sumset \{x\} (?\varphi ' G.sumset A B) = (\lambda a. ?\varphi (x' [\oplus$
 $a])) ' G.sumset A B$
proof
show $GK.sumset \{x\} (?\varphi ' G.sumset A B) \subseteq (\lambda a. ?\varphi (x' [\oplus a])) ' G.sumset A B$

```

proof
  fix  $y$  assume  $y \in GK.sumset \{x\}$  ( $?φ \text{ ' } G.sumset A B$ )
  then obtain  $z$  where  $z \in G.sumset A B$  and  $y = K.quotient-composition$ 
( $?φ x'$ ) ( $?φ z$ )
    using  $GK.sumset.cases \ hxφ$  by blast
    then show  $y \in (\lambda a. ?φ (x' [\oplus] a)) \text{ ' } G.sumset A B$ 
      using  $K.Class-commutes-with-composition \ G.composition-closed$ 
 $hx'G \ G.sumset.cases$ 
       $imageI$  by metis
    qed
  next
    show  $(\lambda a. ?φ (x' [\oplus] a)) \text{ ' } G.sumset A B \subseteq GK.sumset \{x\}$  ( $?φ \text{ ' }$ 
 $G.sumset A B$ )
    proof
      fix  $y$  assume  $y \in (\lambda a. ?φ (x' [\oplus] a)) \text{ ' } G.sumset A B$ 
      then obtain  $z$  where  $hz: z \in G.sumset A B$  and  $y = ?φ (x' [\oplus] z)$ 
by blast
        then have  $y = K.quotient-composition$  ( $?φ x'$ ) ( $?φ z$ ) using
 $K.Class-commutes-with-composition \ G.composition-closed \ hx'G$ 
 $G.sumset.cases$  by metis
        then show  $y \in GK.sumset \{x\}$  ( $?φ \text{ ' } G.sumset A B$ )
          using  $hxφ \ hz \ imageI \ hx \ GK.sumset.sumsetI \ K.natural.map-closed$ 
          by (metis  $G.composition-closed \ hx'G \ insertCI \ G.sumset.cases$ )
        qed
      qed
      have  $G.sumset \{x'\} (G.sumset A B) \subseteq G.sumset (G.sumset A B) \ ?K$ 
      proof
        fix  $y$  assume  $y \in G.sumset \{x'\} (G.sumset A B)$ 
        then obtain  $z$  where  $hz: z \in G.sumset A B$  and  $hy: y = x' [\oplus] z$ 
using  $G.sumset.cases$  by blast
        then have  $?φ (x' [\oplus] z) \in ?φ \text{ ' } G.sumset A B$  using  $hsum \ hsumset$ 
by blast
        then obtain  $w$  where  $hw: \{w\} \subseteq G.sumset A B$  and  $w \in G.sumset$ 
 $A B$ 
          and  $?φ (x' [\oplus] z) = ?φ w$  by auto
          then have  $(x' [\oplus] z) \in G.differenceset (G.sumset \{w\} \ ?K) \ ?K$ 
using  $hφ \ G.sumset-subset-carrier \ hx'G \ hz \ G.sumset-eq-subset-differenceset$ 
 $G.composition-closed \ G.stabilizer-is-nonempty \ G.stabilizer-subset-group$ 
 $G.sumset.cases$ 
           $K.Class-self \ G.differenceset-stabilizer-eq \ G.sumset-assoc$  by metis
          moreover have  $G.differenceset (G.sumset \{w\} \ ?K) \ ?K \subseteq G.sumset$ 
 $\{w\} \ ?K$ 
          using  $hw$  by (simp add: G.differenceset-stabilizer-eq \ G.sumset-assoc)
          ultimately show  $y \in G.sumset (G.sumset A B) \ ?K$  using  $hy \ hw$ 
 $G.sumset-mono \ subsetD$ 
           $subset-refl$  by blast
        qed
      moreover have  $G.sumset (G.sumset A B) \ ?K = G.sumset A B$ 

```

```

    using G.sumset-commute G.sumset-stabilizer-eq-self G.sumset-subset-carrier
  by auto
    ultimately have G.sumset {x'} (G.sumset A B) = G.sumset A B
      by (metis G.finite-sumset G.sumset-subset-carrier card-subset-eq
        G.card-sumset-singleton-subset-eq hA hB hx'G)
    then have x' ∈ ?K using hx'G by (meson empty-subsetI G.finite-sumset
hA hB insert-subset
      G.sumset-eq-sub-stabilizer G.sumset-subset-carrier)
    then show x ∈ {K.Class [0]} using hxφ
      by (metis K.Block-self K.Normal-def K.quotient.unit-closed insertCI)
    qed
  next
  show {K.Class [0]} ⊆ GK.stabilizer (K.Class ' G.sumset A B)
    using GK.zero-mem-stabilizer by auto
  qed
  interpret group-epimorphism ?φ G ([⊕]) [0] G.Factor-Group G ?K
    K.quotient-composition K.Class [0] ..
  interpret GKN: normal-subgroup-in-kernel K.Class G ([⊕]) [0] G.Factor-Group
G ?K
    K.quotient-composition K.Class [0] ?K
  proof
    show ?K ⊆ Ker using K.Block-self K.Normal-def K.quotient.unit-closed
  by blast
  qed
  have hsumK: card (G.sumset A B) = card ?K * card (?φ ' (G.sumset A
B))
  using G.finite-sumset hA hB G.sumset-subset-carrier G.Union-stabilizer-Class-eq

    G.sumset-subset-carrier K.Union-Coset-card-eq by simp
  have hGKsumset: GK.sumset (?φ ' A) (?φ ' B) = ?φ ' (G.sumset A B)
  proof
    show GK.sumset (?φ ' A) (?φ ' B) ⊆ ?φ ' G.sumset A B
  proof
    fix x assume x ∈ GK.sumset (?φ ' A) (?φ ' B)
    then obtain a b where ha: a ∈ A and hb: b ∈ B and
      x = K.quotient-composition (?φ a) (?φ b) using GK.sumset.cases by
blast
    then have x = ?φ (a [⊕] b) by (meson K.Class-commutes-with-composition
hAG hBG in-mono)
    then show x ∈ ?φ ' G.sumset A B using ha hb hAG hBG by blast
  qed
  next
  show ?φ ' G.sumset A B ⊆ GK.sumset (?φ ' A) (?φ ' B)
  proof
    fix x assume x ∈ ?φ ' G.sumset A B
    then obtain c where c ∈ G.sumset A B and x = ?φ c by blast
    then obtain a b where a ∈ A and b ∈ B and x = ?φ (a [⊕] b)
      using G.sumset.cases by metis
    then show x ∈ GK.sumset (?φ ' A) (?φ ' B) using GK.sumset.cases

```

```

      K.Class-commutes-with-composition hAG hBG in-mono
    by (smt (verit, best) GK.sumset.simps K.natural.map-closed imageI)
  qed
  have hAK: card (G.sumset A ?K) = card ?K * card (?φ ' A) using hAG
K.Union-Coset-card-eq hA
      G.sumset-stabilizer-eq-Class-Union G.Class-image-sumset-stabilizer-eq
    by (smt (verit, ccfv-threshold) card-0-eq G.card-sumset-0-iff G.finite-sumset
hAne hB hBG hBne
      G.stabilizer-finite G.sumset-subset-carrier)
    have hBK: card (G.sumset B ?K) = card ?K * card (?φ ' B) using hBG
K.Union-Coset-card-eq hB
      G.sumset-stabilizer-eq-Class-Union G.Class-image-sumset-stabilizer-eq
    by (smt (verit, ccfv-SIG) card-0-eq G.card-sumset-0-iff G.finite-sumset hA
hAG hAne hBne
      G.stabilizer-finite G.sumset-subset-carrier)
    have card (?φ ' A) ≤ card A by (simp add: card-image-le hA)
    moreover have card (?φ ' (G.sumset A B)) < card (G.sumset A B) using
hsumK hcardK-gt-1
      G.card-sumset-0-iff hA hB hAne hBne by (metis card-eq-0-iff card-image-le
hAG hBG
      le-neq-implies-less less-not-refl3 mult-cancel2 nat-mult-1)
    ultimately have card (GK.sumset (?φ ' A) (?φ ' B)) + card (?φ ' A) <
card (G.sumset A B) + card A
      using hGKsumset by auto
    then obtain m where m < n and card (GK.sumset (?φ ' A) (?φ ' B)) +
card (?φ ' A) = m
      using hcardsum by blast
    moreover have hφAsub: ?φ ' A ⊆ G.Factor-Group G ?K
    proof
      fix x assume x ∈ ?φ ' A then obtain a where a ∈ G and ?φ a = x
using hAG by blast
      then show x ∈ G.Factor-Group G ?K by blast
    qed
    moreover have hφBsub: ?φ ' B ⊆ G.Factor-Group G ?K
    proof
      fix x assume x ∈ ?φ ' B then obtain b where b ∈ G and ?φ b = x
using hBG by blast
      then show x ∈ G.Factor-Group G ?K by blast
    qed
    moreover have hφA: finite (?φ ' A) and hφB: finite (?φ ' B) and hφAne:
?φ ' A ≠ {} and
      hφBne: ?φ ' B ≠ {} using hA hB hAne hBne by auto
    moreover have additive-abelian-group (G.Factor-Group G ?K) K.quotient-composition
      (K.Class [0]) ..
    moreover have GK.stabilizer (GK.sumset (?φ ' A) (?φ ' B)) = {K.Class
[0]}
      using hstab0 hGKsumset by auto

```

ultimately have $hind\varphi$: $card (GK.sumset (?\varphi ' A) (?\varphi ' B)) \geq$
 $card (GK.sumset (?\varphi ' A) \{K.Class [0]\}) + card (GK.sumset (?\varphi ' B)$
 $\{K.Class [0]\}) - 1$
using $hindfactor$ [of m $?\varphi ' A$ $?\varphi ' B$] **by** $simp$
have $h\varphi sumA$: $GK.sumset (?\varphi ' A) \{K.Class [0]\} = ?\varphi ' A$
by ($simp$ add : $Int-absorb1$ $Int-commute$ $h\varphi Asub$)
have $h\varphi sumB$: $GK.sumset (?\varphi ' B) \{K.Class [0]\} = ?\varphi ' B$
by ($simp$ add : $Int-absorb1$ $Int-commute$ $h\varphi Bsub$)
have $card (G.sumset A ?K) + card (G.sumset B ?K) - card ?K =$
 $card ?K * (card (?\varphi ' A) + card (?\varphi ' B) - 1)$
using hAK hBK $add-mult-distrib2$ $diff-mult-distrib2$ $nat-mult-1-right$ **by**
 $presburger$
also have $\dots \leq card ?K * card (GK.sumset (?\varphi ' A) (?\varphi ' B))$
using $hind\varphi$ $h\varphi sumA$ $h\varphi sumB$ **by** $simp$
finally show $?thesis$ **by** ($simp$ add : $hGKsumset$ $hsumK$)
qed
qed
qed
thus $?thesis$ **using** $assms$ $hAne$ $hBne$ $additive-abelian-group-axioms$ **by** $blast$
qed

2.2 Strict version of Kneser's Theorem

We show a strict version of Kneser's Theorem as presented in Theorem 3.2 of [3].

theorem $Kneser-strict-aux$: **fixes** A and B **assumes** hAG : $A \subseteq G$ and hBG : $B \subseteq G$ and hA : $finite\ A$
and hB : $finite\ B$ and $hAne$: $A \neq \{\}$ and $hBne$: $B \neq \{\}$ and
 $hineq$: $card (sumset\ A\ B) > card (sumset\ A\ (stabilizer\ (sumset\ A\ B))) +$
 $card (sumset\ B\ (stabilizer\ (sumset\ A\ B))) - card (stabilizer\ (sumset\ A\ B))$
shows $card (sumset\ A\ B) \geq card\ A + card\ B$

proof –

let $?H = stabilizer (sumset\ A\ B)$
have $hfin$: $finite\ ?H$ **using** $stabilizer-subset-group$ $stabilizer-finite$ $sumset-subset-carrier$

 $finite-sumset$ $assms$ $sumset-is-empty-iff$ $sumset-stabilizer-eq-self$ **by** $metis$
have $card\ ?H\ dvd\ card (sumset\ A\ B)$ and $card\ ?H\ dvd (card (sumset\ A\ (stabilizer$
 $(sumset\ A\ B))) +$
 $card (sumset\ B\ (stabilizer (sumset\ A\ B))) - card\ ?H)$
using $card-stabilizer-divide-sumset$ hAG hBG $card-stabilizer-sumset-divide-sumset$
by $auto$
then have $card (sumset\ A\ B) \geq card (sumset\ A\ ?H) + card (sumset\ B\ ?H)$
using $hineq$
by ($metis$ $diff-le-mono2$ $dvd-add-right-iff$ $dvd-imp-le$ $le-diff-conv$ $less-imp-add-positive$)
moreover have $card\ A + card\ B \leq card (sumset\ A\ ?H) + card (sumset\ B\ ?H)$
using $card-le-sumset$ $sumset-commute$ $assms$ $stabilizer-subset-group$ $stabilizer-is-nonempty$

 $Int-emptyI$ $inf.orderE$ $add-mono$ $hfin$ **by** $metis$

ultimately show *?thesis* by *linarith*
qed

theorem *Kneser-strict*: fixes A and B assumes $A \subseteq G$ and $B \subseteq G$ and finite A and finite B
and $\text{stabilizer}(\text{sumset } A \ B) = H$ and $A \neq \{\}$ and $B \neq \{\}$ and $\text{card}(\text{sumset } A \ B) < \text{card } A + \text{card } B$
shows $\text{card}(\text{sumset } A \ B) = \text{card}(\text{sumset } A \ (\text{stabilizer}(\text{sumset } A \ B))) + \text{card}(\text{sumset } B \ (\text{stabilizer}(\text{sumset } A \ B))) - \text{card}(\text{stabilizer}(\text{sumset } A \ B))$
using *Kneser Kneser-strict-ax* *assms le-antisym nat-less-le* by *metis*

2.3 The CauchyDavenport Theorem

We show the CauchyDavenport Theorem as a corollary of Kneser's Theorem, following a comment on Theorem 3.2 in [3].

interpretation *Z-p: additive-abelian-group* $\{0..int((p :: nat)-1)\}$ $(\lambda x y. ((x + y) \text{ mod } int \ p))$ $0::int$
using *additive-abelian-group-def residue-group[of p]* by *fastforce*

theorem *Cauchy-Davenport*:

fixes $p :: nat$
assumes prime p and $A \neq \{\}$ and $B \neq \{\}$ and finite A and finite B and $A \subseteq \{0..p-1\}$ and $B \subseteq \{0..p-1\}$
shows $\text{card}(Z\text{-}p.\text{sumset } p \ A \ B) \geq \text{Min} \{p, \text{card } A + \text{card } B - 1\}$

proof(cases $Z\text{-}p.\text{stabilizer } p \ (Z\text{-}p.\text{sumset } p \ A \ B) = \{0\}$)

case *True*

moreover have $Z\text{-}p.\text{sumset } p \ A \ \{0\} = A$ and $Z\text{-}p.\text{sumset } p \ B \ \{0\} = B$ using *assms Z-p.sumset-D(1)* by *auto*

ultimately show *?thesis* using *Z-p.Kneser[of A p B]* *assms* by *fastforce*

next

case *hne: False*

let $?H = Z\text{-}p.\text{stabilizer } p \ (Z\text{-}p.\text{sumset } p \ A \ B)$

have $?H = \{0..int(p-1)\}$ using *hne Z-p.stabilizer-is-subgroup[of p Z-p.sumset p A B]*

residue-group-simple[OF assms(1)] by *blast*

moreover have $p \geq 2$ using *assms(1)* by (*simp add: prime-ge-2-nat*)

ultimately have $\text{card } ?H = p$ using *card-atLeastAtMost* by (*simp add: of-nat-diff*)

then have $p \leq \text{card} \ (Z\text{-}p.\text{sumset } p \ A \ B)$

using *Z-p.card-stabilizer-le card-0-eq* *assms Z-p.card-sumset-0-iff Z-p.sumset.cases*

Z-p.sumset-subset-carrier Z-p.finite-sumset by *metis*

then show *?thesis* by *auto*

qed

end

end

References

- [1] M. DeVos. A short proof of kneser's addition theorem for abelian groups. In M. B. Nathanson, editor, *Combinatorial and Additive Number Theory*, pages 39–41, New York, NY, 2014. Springer New York.
- [2] M. B. Nathanson. *Additive Number Theory: Inverse Problems and the Geometry of Sumsets*, volume 165 of *Graduate Texts in Mathematics*. Springer-Verlag, 1996.
- [3] I. Z. Ruzsa. Sumsets and structure, 2008. Course notes, available on <https://www.math.cmu.edu/users/af1p/Teaching/AdditiveCombinatorics/Additive-Combinatorics.pdf>.