

Impossibility of the Dissection of a Cube

Thomas Holme Surlykke

November 25, 2024

Abstract

This entry formalizes Littlewood’s argument [2], demonstrating that a 3-dimensional cube cannot be dissected into a finite collection of smaller cubes, each of a different size. The formalization addresses theorem #82, “Dissection of Cubes (J.E. Littlewood’s ‘elegant’ proof)” from Freek Wiedijk’s “100 Mathematical Theorems” list [1], and is based upon a prior formalization in Lean [3].

Contents

1	Basic definitions	2
1.1	point and cube definitions	2
1.2	Calculations with sets from cubes	3
1.2.1	Point membership	3
1.2.2	Cubes subset of each other, by <i>side</i>	4
2	Cubing	5
2.1	Properties of <i>is-dissection</i>	6
3	Hole	7
3.1	Definitions	8
3.2	Properties of a hole	8
3.3	Properties of cubes on a hole	9
4	Bottom of <i>unit-cube</i> is a hole	10
5	Minimum cube on hole is interior	12
5.1	Definition: Minimum cube on <i>h</i>	12
5.2	Minimum cube on hole is interior	13
6	Minimum cube of hole induces hole on top	17

7 The main result

19

theory *Cube-Dissection*

imports *Complex-Main HOL-Library.Disjoint-Sets HOL-Library.Infinite-Set*
begin

Proof that a cube can't be dissected into a finite number of subcubes of different size This formalization is heavily inspired by the Lean proof of the same fact, [3]. One goal of this project, is that by restricting to cubes of dimension 3 the logic will be easier to follow

1 Basic definitions

1.1 point and cube definitions

record *point* = *px*:: *real* *py*::*real* *pz*::*real*
record *cube* = *point*:: *point* *width*::*real*
datatype *axis* = *x* | *y* | *z*
abbreviation *coordinate* \equiv *case-axis px py pz*

abbreviation *is-valid* :: *cube* \Rightarrow *bool* **where** *is-valid* *c* \equiv (*width* *c* > 0)

Min value of cube along given axis

hide-const (**open**) *min*

abbreviation *min* :: *axis* \Rightarrow *cube* \Rightarrow *real* **where** *min* *ax* *c* \equiv *coordinate* *ax* (*point* *c*)

Max value (supremum) along given axis

hide-const (**open**) *max*

abbreviation *max* :: *axis* \Rightarrow *cube* \Rightarrow *real* **where** *max* *ax* *c* \equiv *min* *ax* *c* + *width* *c*

Sides of a cube. Half-open intervals, so that a dissection both is a cover, and consists of disjoint cubes

abbreviation *side* :: *axis* \Rightarrow *cube* \Rightarrow *real* *set* **where**

side *ax* *c* \equiv {*min* *ax* *c* ..< *max* *ax* *c*}

Sets of points generated from cubes

definition *to-set* :: *cube* \Rightarrow *point* *set* **where**

to-set *c* = {*p*. *px* *p* \in *side* *x* *c* \wedge *py* *p* \in *side* *y* *c* \wedge *pz* *p* \in *side* *z* *c*}

definition *bot* :: *cube* \Rightarrow *point* *set* **where**

bot *c* = {*p*. *px* *p* \in *side* *x* *c* \wedge *py* *p* \in *side* *y* *c* \wedge *pz* *p* = *min* *z* *c*}

definition *top* :: *cube* \Rightarrow *point* *set* **where**

top *c* = {*p*. *px* *p* \in *side* *x* *c* \wedge *py* *p* \in *side* *y* *c* \wedge *pz* *p* = *max* *z* *c*}

Moves a cube its width down (so top face to bottom face)

definition *shift-down* :: *cube* \Rightarrow *cube* **where**

shift-down *c* = *c* (*point* := *point* *c* (*pz* := *min* *z* *c* - *width* *c*))

1.2 Calculations with sets from cubes

A bunch of statements we need about how cubes can be compared by *side*

lemma *top-shift-down-eq-bot*: $top (shift\text{-}down\ c) = bot\ c$
unfolding *top-def bot-def shift-down-def* **by** *simp*

Sets not empty

lemma *non-empty*: $is\text{-}valid\ c \implies to\text{-}set\ c \neq \{\}$
unfolding *to-set-def* **by** *force*

lemma *top-non-empty*: $is\text{-}valid\ c \implies top\ c \neq \{\}$
unfolding *top-def* **by** (*auto intro!*: $exI[of\ \text{-}\ point\ c\ (\lambda\ pz :=\ max\ z\ c)]$)

min of a cube is in corresponding *side*

lemma *min-in-side*: $is\text{-}valid\ c \implies min\ ax\ c \in side\ ax\ c$
by *simp*

lemma *min-ne-max*: $is\text{-}valid\ c \implies min\ ax\ c \neq max\ ax\ c$
by *simp*

lemma *min-lt-max*: $is\text{-}valid\ c \implies min\ ax\ c < max\ ax\ c$
by *simp*

lemma *bot-subset*: $bot\ c \subseteq to\text{-}set\ c$
unfolding *bot-def to-set-def* **by**(*auto*)

1.2.1 Point membership

Points in a cube's set, by looking at membership of *side*

lemma *in-set-by-side*: $p \in to\text{-}set\ c \iff$
 $px\ p \in side\ x\ c \wedge py\ p \in side\ y\ c \wedge pz\ p \in side\ z\ c$
by(*cases p, simp add: to-set-def*)

lemma *in-set-by-side-2*: $(\lambda px=x0, py=y0, pz=z0) \in to\text{-}set\ c \iff$
 $x0 \in side\ x\ c \wedge y0 \in side\ y\ c \wedge z0 \in side\ z\ c$
by(*simp add: to-set-def*)

lemma *in-bot-by-side*: $p \in bot\ c \iff$
 $px\ p \in side\ x\ c \wedge py\ p \in side\ y\ c \wedge pz\ p = min\ z\ c$
by(*simp add: bot-def*)

lemma *in-bot-by-side-2*: $(\lambda px=x0, py=y0, pz=z0) \in bot\ c \iff$
 $x0 \in side\ x\ c \wedge y0 \in side\ y\ c \wedge z0 = min\ z\ c$
by(*simp add: in-bot-by-side*)

lemma *in-top-by-side*: $p \in top\ c \iff$
 $px\ p \in side\ x\ c \wedge py\ p \in side\ y\ c \wedge pz\ p = max\ z\ c$
by(*cases p, auto simp add: top-def*)

lemma *in-top-by-side-2*: $(\lambda px=x0, py=y0, pz=z0) \in top\ c \iff$
 $x0 \in side\ x\ c \wedge y0 \in side\ y\ c \wedge z0 = max\ z\ c$
by(*simp add: top-def*)

lemma *all-point-iff*: $(\forall p. P\ p) \iff (\forall x1\ y1\ z1. P\ (\lambda px = x1, py = y1, pz = z1))$

by (*metis point.cases*)

Intersection by *side*

lemma *set-intersect-by-side*: $to\text{-set } c1 \cap to\text{-set } c2 \neq \{\}$ \longleftrightarrow
 $side\ x\ c1 \cap side\ x\ c2 \neq \{\} \wedge side\ y\ c1 \cap side\ y\ c2 \neq \{\} \wedge side\ z\ c1 \cap side\ z\ c2$
 $\neq \{\}$
unfolding *set-eq-iff all-point-iff* **using** *in-set-by-side-2* **by** *blast*

lemma *bot-intersect-by-side*: $bot\ c1 \cap bot\ c2 \neq \{\}$
 $\longleftrightarrow side\ x\ c1 \cap side\ x\ c2 \neq \{\} \wedge side\ y\ c1 \cap side\ y\ c2 \neq \{\} \wedge min\ z\ c1 = min$
 $z\ c2$

proof(*intro iffI*)

assume $bot\ c1 \cap bot\ c2 \neq \{\}$

thus $side\ x\ c1 \cap side\ x\ c2 \neq \{\} \wedge side\ y\ c1 \cap side\ y\ c2 \neq \{\} \wedge min\ z\ c1 =$
 $min\ z\ c2$

using *in-bot-by-side* **by** *fastforce*

next

assume $side\ x\ c1 \cap side\ x\ c2 \neq \{\} \wedge side\ y\ c1 \cap side\ y\ c2 \neq \{\} \wedge min\ z\ c1 =$
 $min\ z\ c2$

thus $bot\ c1 \cap bot\ c2 \neq \{\}$ **using** *in-bot-by-side-2* **by** *blast*

qed

lemma *bot-top-intersect-by-side*: $bot\ c1 \cap top\ c2 \neq \{\}$
 $\longleftrightarrow side\ x\ c1 \cap side\ x\ c2 \neq \{\} \wedge side\ y\ c1 \cap side\ y\ c2 \neq \{\} \wedge min\ z\ c1 = max$
 $z\ c2$

proof(*intro iffI*)

assume $bot\ c1 \cap top\ c2 \neq \{\}$

thus $side\ x\ c1 \cap side\ x\ c2 \neq \{\} \wedge side\ y\ c1 \cap side\ y\ c2 \neq \{\} \wedge min\ z\ c1 =$
 $max\ z\ c2$

using *in-bot-by-side in-top-by-side* **by** *fastforce*

next

assume $side\ x\ c1 \cap side\ x\ c2 \neq \{\} \wedge side\ y\ c1 \cap side\ y\ c2 \neq \{\} \wedge min\ z\ c1 =$
 $max\ z\ c2$

thus $bot\ c1 \cap top\ c2 \neq \{\}$ **using** *in-bot-by-side-2 in-top-by-side-2* **by** *blast*

qed

1.2.2 Cubes subset of each other, by *side*

lemma *set-subset-by-side*: $to\text{-set } c1 \subseteq to\text{-set } c2 \longleftrightarrow$
 $side\ x\ c1 \subseteq side\ x\ c2 \wedge side\ y\ c1 \subseteq side\ y\ c2 \wedge side\ z\ c1 \subseteq side\ z\ c2$
unfolding *subset-eq Ball-def all-point-iff in-set-by-side* **by** *fastforce*

lemma *set-eq-by-side*: $to\text{-set } c1 = to\text{-set } c2 \longleftrightarrow$
 $side\ x\ c1 = side\ x\ c2 \wedge side\ y\ c1 = side\ y\ c2 \wedge side\ z\ c1 = side\ z\ c2$
using *set-subset-by-side* **by** *fast*

lemma *bot-eq-by-side*: $is\text{-valid } c1 \implies bot\ c1 = bot\ c2 \longleftrightarrow$
 $side\ x\ c1 = side\ x\ c2 \wedge side\ y\ c1 = side\ y\ c2 \wedge min\ z\ c1 = min\ z\ c2$

unfolding *bot-def set-eq-iff all-point-iff* **by** *fastforce*

lemma *bot-top-subset-by-side*: $is\text{-}valid\ c1 \implies bot\ c1 \subseteq top\ c2 \iff$
 $side\ x\ c1 \subseteq side\ x\ c2 \wedge side\ y\ c1 \subseteq side\ y\ c2 \wedge min\ z\ c1 = max\ z\ c2$
unfolding *top-def bot-def subset-eq Ball-def all-point-iff* **by** *fastforce*
lemma *bot-top-eq-by-side*: $is\text{-}valid\ c1 \implies bot\ c1 = top\ c2 \iff$
 $side\ x\ c1 = side\ x\ c2 \wedge side\ y\ c1 = side\ y\ c2 \wedge min\ z\ c1 = max\ z\ c2$
unfolding *top-def bot-def set-eq-iff Ball-def all-point-iff* **by** *fastforce*

lemma *width-eq-if-side-eq*: $\llbracket is\text{-}valid\ c1; side\ ax\ c1 = side\ ax\ c2 \rrbracket \implies width\ c1 =$
 $width\ c2$

by(*simp add:Ico-eq-Ico*)

to-set is injective

lemma *to-set-inj*:

assumes *is-valid c1 to-set c1 = to-set c2*

shows $c1 = c2$

proof –

from *assms(2)* **have** $*$: $side\ x\ c1 = side\ x\ c2\ side\ y\ c1 = side\ y\ c2\ side\ z\ c1 =$
 $side\ z\ c2$

by(*metis set-eq-by-side*)**+**

hence $min\ x\ c1 = min\ x\ c2\ min\ y\ c1 = min\ y\ c2\ min\ z\ c1 = min\ z\ c2$

by(*meson assms(1) Ico-eq-Ico less-add-same-cancel1*)**+**

hence $point\ c1 = point\ c2$ **by** *simp*

moreover from $*$ **have** $width\ c1 = width\ c2$ **by**(*metis assms(1) width-eq-if-side-eq*)

ultimately show $c1 = c2$ **by** *simp*

qed

Cube-Dissection.bot is also injective

lemma *bot-inj*: **assumes** *is-valid c1 bot c1 = bot c2* **shows** $c1 = c2$

proof –

from *assms* **have** $width\ c1 = width\ c2$ **by**(*metis assms bot-eq-by-side width-eq-if-side-eq*)

hence $to\text{-}set\ c1 = to\text{-}set\ c2$ **using** *assms bot-eq-by-side set-eq-by-side* **by** *auto*

thus $c1 = c2$ **by**(*metis assms(1) to-set-inj*)

qed

2 Cubing

We in this section introduce a dissection C of the unit cube

The cube we show there is no dissection of

definition *unit-cube* **where** $unit\text{-}cube = (\ point=(\ px=0, py=0, pz=0), width=1)$

lemma *min-unit-cube-0*: $min\ ax\ unit\text{-}cube = 0$

by(*metis unit-cube-def axis.exhaust axis.simps(7–9) cube.select-convs(1) point.select-convs(1–3)*)

lemma *unit-cube-valid[simp]*: *is-valid unit-cube*

by(*simp add: unit-cube-def*)

What we want to show doesn't exist. C is a set of cubes which satisfy:

1. All cubes are valid (width > 0)
2. All cubes are disjoint
3. The union of the cubes in C equal *unit-cube* (hence, all cubes are contained in *unit-cube*)
4. All cubes in C have different width
5. There are at least two cubes in C
6. There are a finite number of cubes in C

definition *is-dissection* :: *cube set* \Rightarrow *bool* **where**

is-dissection $C \iff$
 $(\forall c \in C. \text{is-valid } c)$
 $\wedge \text{disjoint (image to-set } C)$
 $\wedge \bigcup (\text{image to-set } C) = \text{to-set unit-cube}$
 $\wedge \text{inj-on width } C$ — All cubes are of different size
 $\wedge \text{card } C \geq 2$ — At least two cubes
 $\wedge \text{finite } C$

From now on, C is some fixed dissection of *unit-cube*, and 'dissection' refers to this fact

context *fixes* C **assumes** *dissection*: *is-dissection* C
begin

2.1 Properties of *is-dissection*

lemma *valid-if-dissection[simp]*: $c \in C \implies \text{is-valid } c$
using *dissection unfolding is-dissection-def* **by** *fast*

lemma *side-unit-cube*:
 $\text{side ax unit-cube} = \{0..<1\}$
by(*metis add.commute add.right-neutral cube.select-convs(2) min-unit-cube-0 unit-cube-def*)

lemma *subset-unit-cube-if-dissection*: $c \in C \implies \text{to-set } c \subseteq \text{to-set unit-cube}$
using *dissection unfolding is-dissection-def to-set-def* **by** *fast*

lemma *subset-unit-cube-by-side*:
 $c \in C \implies \text{side ax } c \subseteq \{0..<1\}$
by(*metis(full-types)subset-unit-cube-if-dissection set-subset-by-side side-unit-cube axis.exhaust*)

lemma *eq-iff-intersect*: $\llbracket c1 \in C; c2 \in C \rrbracket \implies c1 = c2 \iff \text{to-set } c1 \cap \text{to-set } c2 \neq \{\}$
using *dissection unfolding is-dissection-def disjoint-def*
by (*metis image-eqI inf.idem non-empty to-set-inj*)

Whenever we have a point in *unit-cube*, there exists a (unique) cube in C containing that point

lemma *obtain-cube*: $p \in \text{to-set } \textit{unit-cube} \implies \exists c \in C. p \in \text{to-set } c$
using *dissection unfolding is-dissection-def* **by** *blast*

If the top of c doesn't touch the top of *unit-cube*, then top of c must be covered by bottoms of cubes in C

lemma *top-cover-by-bot*:

assumes $c \in C \text{ max } z \ c < 1$

shows $\text{top } c \subseteq \bigcup (\text{image } \textit{bot } C)$

proof(*intro subsetI*)

fix p **assume** *p-in-top*: $p \in \text{top } c$

from *assms(1)* **have** $\text{side } x \ c \subseteq \{0..<1\}$ $\text{side } y \ c \subseteq \{0..<1\}$ $\text{side } z \ c \subseteq \{0..<1\}$

using *subset-unit-cube-by-side* **by** *blast+*

with *assms(2)* *p-in-top* **have** $px \ p \in \{0..<1\}$ $py \ p \in \{0..<1\}$ $pz \ p \in \{0..<1\}$

by(*simp add: in-top-by-side*)**+**

hence $p \in \text{to-set } \textit{unit-cube}$ **using** *in-set-by-side side-unit-cube* **by** *blast*

then obtain c' **where** $c' \in C \ p \in \text{to-set } c'$ **by**(*metis obtain-cube*)

moreover have $p \notin \text{to-set } c$ **using** *p-in-top in-top-by-side in-set-by-side* **by** *simp*

ultimately have $c' \neq c$ **by** *metis*

hence $\text{to-set } c' \cap \text{to-set } c = \{\}$ **by**(*metis assms(1) <c' ∈ C> eq-iff-intersect*)

have $p \in \text{bot } c'$

proof(*rule ccontr*)

assume $p \notin \text{bot } c'$

hence $\min z \ c' < pz \ p \ pz \ p < \max z \ c'$ **using** $\langle p \in \text{to-set } c' \rangle$

by(*simp-all add: in-set-by-side in-bot-by-side*)

hence $\text{side } z \ c' \cap \text{side } z \ c \neq \{\}$ **using** $\langle p \in \text{top } c \rangle$ **by**(*simp add: in-top-by-side*)

moreover have $\text{side } x \ c \cap \text{side } x \ c' \neq \{\}$ $\text{side } y \ c \cap \text{side } y \ c' \neq \{\}$

using $\langle p \in \text{top } c \rangle \langle p \in \text{to-set } c' \rangle$ **by**(*simp-all add: in-top-by-side in-set-by-side*

assms)

ultimately have $\text{to-set } c' \cap \text{to-set } c \neq \{\}$ **using** *set-intersect-by-side* **by** *simp*

thus *False* **using** $\langle \text{to-set } c' \cap \text{to-set } c = \{\} \rangle$ **by** *metis*

qed

thus $p \in \bigcup (\text{image } \textit{bot } C)$ **using** $\langle c' \in C \rangle$ **by** *blast*

qed

3 Hole

A hole h is a special kind of cube, where any cube whose bottom 'touches' the top of v must in fact have its bottom contained in the top of v . If $h \in C$, then this happens because all the other cubes surrounding h go up taller, forming a hole on top of v . Note that we don't require that $h \in C$, but this is only so we can prove that *unit-cube* shifted down by 1 is a hole - all other holes will in fact lie in C . The concept of a hole is inspired by the 'Valley' definition from [3]

3.1 Definitions

definition *is-hole* :: *cube* \Rightarrow *bool* **where**

is-hole *h* \longleftrightarrow

is-valid *h*

\wedge *top* *h* $\subseteq \bigcup (\text{image } \text{bot } C)$

$\wedge (\forall c \in C . \text{bot } c \cap \text{top } h \neq \{\} \longrightarrow \text{bot } c \subseteq \text{top } h)$

— *v* could be a cube in *C* (and most often is), but any other cube must be different width. Also, this assumption is not actually needed (as it follows from *v*, $c \in C$), but without it we have to do a special-case proof for the bottom of the *unit-cube*

$\wedge (\forall c \in C . c \neq h \longrightarrow \text{width } c \neq \text{width } h)$

Subset of *C* which are on a given hole *h*

definition *is-on-hole* :: *cube* \Rightarrow *cube* \Rightarrow *bool* **where**

is-on-hole *h* *c* $\longleftrightarrow \text{bot } c \subseteq \text{top } h$

definition *filter-on-hole* :: *cube* \Rightarrow *cube set* **where**

filter-on-hole *h* = *Set.filter* (*is-on-hole* *h*) *C*

3.2 Properties of a hole

Terminology: 'on hole' means cube *c* with: *Cube-Dissection.bot* *c* \subseteq *Cube-Dissection.top* *h*. 'in hole' means point *p* with: $p \in \text{Cube-Dissection.top } h$

local.filter-on-hole *h* $\subseteq C$

lemma *dissection-if-on-hole[simp]*: $c \in \text{filter-on-hole } h \Longrightarrow c \in C$

unfolding *filter-on-hole-def* **by** *simp*

Holes, and cubes on them, are valid

lemma *valid-if-hole[simp]*: *is-hole* *h* \Longrightarrow *is-valid* *h*

by (*metis is-hole-def*)

lemma *valid-if-on-hole[simp]*: $c \in \text{filter-on-hole } h \Longrightarrow \text{is-valid } c$

by *simp*

lemma *on-hole-finite*: *is-hole* *h* \Longrightarrow *finite* (*filter-on-hole* *h*)

by (*metis dissection is-dissection-def filter-on-hole-def finite-filter*)

lemma *on-hole-if-in-filter-on-hole*: $c \in \text{filter-on-hole } h \Longrightarrow \text{is-on-hole } h$ *c*

unfolding *filter-on-hole-def* **by** *simp*

lemma *on-hole-cover*: **assumes** *is-hole* *h* **shows** *top* *h* $\subseteq \bigcup (\text{image } \text{bot } (\text{filter-on-hole } h))$

proof

fix *p* **assume** $p \in \text{top } h$

then obtain *c* **where** $c \in C$ $p \in \text{bot } c$ **using** *assms* **unfolding** *is-hole-def* **by** *blast*

with $\langle p \in \text{top } h \rangle$ **have** $\text{bot } c \subseteq \text{top } h$ **using** *assms* **unfolding** *is-hole-def* **by** *blast*

hence $c \in \text{filter-on-hole } h$

by (*metis* $\langle c \in C \rangle$ *filter-on-hole-def is-on-hole-def member-filter*)

with $\langle p \in \text{bot } c \rangle$ **show** $p \in \bigcup (\text{image } \text{bot}(\text{filter-on-hole } h))$ **by** *blast*
qed

Whenever we have a point p in the top of a hole h , there exists a (unique) cube $c \in \text{local.filter-on-hole } h$, such that $p \in \text{Cube-Dissection.bot } c$

lemma *obtain-cube-if-in-hole*: $\llbracket \text{is-hole } h; p \in \text{top } h \rrbracket$
 $\implies \exists c \in \text{filter-on-hole } h . p \in \text{bot } c$
using *on-hole-cover* **by** *blast*

lemma *on-hole-inj-on-width*: $\text{is-hole } h \implies \text{inj-on width } (\text{filter-on-hole } h)$
using *dissection-if-on-hole inj-on-subset dissection*
is-dissection-def subset-iff **by** *metis*

3.3 Properties of cubes on a hole

lemma *neg-hole-if-on-hole*: $c \in \text{filter-on-hole } h \implies c \neq h$
using *valid-if-on-hole min-ne-max bot-top-subset-by-side is-on-hole-def*
on-hole-if-in-filter-on-hole **by** *blast*

lemma *subset-if-on-hole*: $c \in \text{filter-on-hole } h \implies \text{bot } c \subseteq \text{top } h$
unfolding *is-hole-def filter-on-hole-def is-on-hole-def* **by** *simp*

lemma *side-subset-if-on-hole*: $\llbracket c \in \text{filter-on-hole } h; ax \in \{x, y\} \rrbracket \implies \text{side } ax \ c \subseteq \text{side } ax \ h$
using *bot-top-subset-by-side subset-if-on-hole* **by** *auto*

lemma *min-z-eq-max-z-hole-if-on-hole*:
 $c \in \text{filter-on-hole } h \implies \text{min } z \ c = \text{max } z \ h$
using *subset-if-on-hole* **by**(*simp add: bot-top-subset-by-side*)

lemma *z-eq-if-on-hole*:
 $\llbracket c1 \in \text{filter-on-hole } h; c2 \in \text{filter-on-hole } h \rrbracket \implies \text{min } z \ c1 = \text{min } z \ c2$
using *min-z-eq-max-z-hole-if-on-hole* **by** *simp*

Do not need to care about z -coordinate

lemma *eq-iff-side-eq-if-on-hole*: $\llbracket c1 \in \text{filter-on-hole } h; c2 \in \text{filter-on-hole } h \rrbracket$
 $\implies c1 = c2 \iff \text{side } x \ c1 = \text{side } x \ c2 \wedge \text{side } y \ c1 = \text{side } y \ c2$
by(*meson valid-if-on-hole z-eq-if-on-hole bot-eq-by-side bot-inj*)

Disjointness-lemmas:

lemma *eq-iff-bot-intersect-if-on-hole*:
assumes $c1 \in \text{filter-on-hole } h \ c2 \in \text{filter-on-hole } h$
shows $c1 = c2 \iff \text{bot } c1 \cap \text{bot } c2 \neq \{\}$

proof –

from *assms* **have** $c1 = c2 \iff \text{to-set } c1 \cap \text{to-set } c2 \neq \{\}$ **by**(*simp add: eq-iff-intersect*)

moreover **have** $\text{min } z \ c1 = \text{min } z \ c2$ **by**(*metis assms z-eq-if-on-hole*)

hence $\text{to-set } c1 \cap \text{to-set } c2 \neq \{\} \iff \text{bot } c1 \cap \text{bot } c2 \neq \{\}$

using *set-intersect-by-side bot-intersect-by-side bot-subset subsetD* **by** *blast*

ultimately show $c1 = c2 \iff \text{bot } c1 \cap \text{bot } c2 \neq \{\}$ **by** *blast*
qed

lemma *eq-iff-side-intersect-if-on-hole*:

$\llbracket c1 \in \text{filter-on-hole } h; c2 \in \text{filter-on-hole } h \rrbracket$
 $\implies c1 = c2 \iff \text{side } x \ c1 \cap \text{side } x \ c2 \neq \{\} \wedge \text{side } y \ c1 \cap \text{side } y \ c2 \neq \{\}$
by (*meson z-eq-if-on-hole eq-iff-bot-intersect-if-on-hole bot-intersect-by-side*)

lemma *width-on-hole-lt-width-hole*:

assumes *is-hole* $h \ c \in \text{filter-on-hole } h$ **shows** $\text{width } c < \text{width } h$
proof (*rule ccontr*)
assume $\neg \text{width } c < \text{width } h$
hence $\text{width } c \geq \text{width } h$ **by** *simp*
moreover have $c \neq h$ **by** (*metis assms(2) neq-hole-if-on-hole*)
hence $\text{width } c \neq \text{width } h$ **using** *assms unfolding is-hole-def* **by** *simp*
ultimately have $\text{width } c > \text{width } h$ **by** *simp*
hence $\neg(\text{side } x \ c \subseteq \text{side } x \ h)$ **using** *assms less-le-not-le* **by** *fastforce*
hence $\neg(\text{bot } c \subseteq \text{top } h)$ **using** *assms by(auto simp add: bot-top-subset-by-side)*
moreover have $\text{bot } c \subseteq \text{top } h$ **by** (*metis assms(2) subset-if-on-hole*)
ultimately show *False* **by** *metis*
qed

lemma *strict-subset-if-on-hole*: **assumes** *is-hole* $h \ c \in \text{filter-on-hole } h$
shows $\text{bot } c \subset \text{top } h$

proof
show $\text{bot } c \subseteq \text{top } h$ **by** (*metis assms(2) subset-if-on-hole*)
next
have $\text{width } c \neq \text{width } h$ **using** *assms width-on-hole-lt-width-hole* **by** *fastforce*
hence $\text{side } x \ c \neq \text{side } x \ h$ **by** (*metis assms(1) valid-if-hole width-eq-if-side-eq*)
thus $\text{bot } c \neq \text{top } h$ **using** *assms bot-top-eq-by-side* **by** *simp*
qed

lemma *on-hole-non-empty*: *is-hole* $h \implies \text{filter-on-hole } h \neq \{\}$

using *on-hole-cover*
by (*metis Sup-empty image-empty subset-empty top-non-empty valid-if-hole*)

4 Bottom of *unit-cube* is a hole

lemma *bot-unit-cube-cover-by-bot*: $\text{bot } \text{unit-cube} \subseteq \bigcup (\text{image } \text{bot } C)$

proof
fix p
assume $p \in \text{bot } \text{unit-cube}$
hence $p \in \text{to-set } \text{unit-cube}$ **using** *bot-subset* **by** *auto*
then obtain c **where** $c \in C \ p \in \text{to-set } c$ **by** (*metis obtain-cube*)
from $\langle p \in \text{bot } \text{unit-cube} \rangle$ **have** $pz \ p = 0$ **by** (*metis min-unit-cube-0 in-bot-by-side*)
hence $\text{min } z \ c \leq 0$ **using** $\langle p \in \text{to-set } c \rangle$ **by** (*simp add: in-set-by-side*)
moreover have $\text{side } z \ c \subseteq \{0..<1\}$ **by** (*metis $\langle c \in C \rangle$ subset-unit-cube-by-side*)
hence $\text{min } z \ c \geq 0$ **using** $\langle c \in C \rangle$ **by** *fastforce*
ultimately have $\text{min } z \ c = 0$ **by** *simp*

hence $p \in \text{bot } c$ **using** $\langle p \in \text{to-set } c \rangle \langle \text{pz } p = 0 \rangle$ **by** (*simp add: in-set-by-side in-bot-by-side*)
thus $p \in \bigcup (\text{image bot } C)$ **using** $\langle c \in C \rangle$ **by** *auto*
qed

lemma *eq-if-width-eq-if-subset*:

assumes $\text{width } c1 = \text{width } c2$ $\text{to-set } c1 \subseteq \text{to-set } c2$

shows $\text{to-set } c1 = \text{to-set } c2$

proof –

from *assms(2)* **have** $\text{side } x \ c1 \subseteq \text{side } x \ c2$ $\text{side } y \ c1 \subseteq \text{side } y \ c2$ $\text{side } z \ c1 \subseteq \text{side } z \ c2$

using *set-subset-by-side* **by** *blast+*

hence $\text{side } x \ c1 = \text{side } x \ c2$ $\text{side } y \ c1 = \text{side } y \ c2$ $\text{side } z \ c1 = \text{side } z \ c2$

using *assms(1)* **by** *fastforce+*

thus $\text{to-set } c1 = \text{to-set } c2$ **using** *set-eq-by-side* **by** *blast*

qed

lemma *width-ne-one*:

assumes $c \in C$

shows $\text{width } c \neq 1$

proof (*rule ccontr*)

obtain c' **where** $c' \in C$ $c \neq c'$ **using** *assms dissection*

unfolding *is-dissection-def* **by** (*auto simp add: card-le-Suc-iff numeral-eq-Suc*)

hence $\text{to-set } c' \cap \text{to-set } c = \{\}$ **using** *assms* **by** (*metis eq-iff-intersect*)

assume $\neg \text{width } c \neq 1$

hence $\text{width } c = 1$ **by** *blast*

moreover **have** $\text{to-set } c \subseteq \text{to-set unit-cube}$ **by** (*metis assms subset-unit-cube-if-dissection*)

ultimately **have** $\text{to-set } c = \text{to-set unit-cube}$ **using** *eq-if-width-eq-if-subset*

unfolding *unit-cube-def* **by** *auto*

with $\langle c' \in C \rangle$ **have** $\text{to-set } c' \subseteq \text{to-set } c$ **by** (*metis subset-unit-cube-if-dissection*)

hence $\text{to-set } c' \cap \text{to-set } c \neq \{\}$ **using** $\langle c' \in C \rangle$ **by** (*simp add: non-empty Int-absorb2*)

moreover **have** $\text{to-set } c' \cap \text{to-set } c = \{\}$ **by** (*metis assms* $\langle c' \in C \rangle \langle c \neq c' \rangle$ *eq-iff-intersect*)

ultimately **show** *False* **by** *metis*

qed

Combines the previous lemmas, to show that the bottom of *unit-cube* is a hole

proposition *hole-unit-cube: is-hole (shift-down unit-cube)*

proof –

let $?b = \text{shift-down unit-cube}$ — cube with *point* (0,0,-1) and *width* 1

show $?thesis$ **unfolding** *is-hole-def*

proof (*intro conjI subsetI ballI impI*)

show *is-valid ?b* **by** (*simp add: shift-down-def*)

next

fix p **assume** $p \in \text{top } ?b$

hence $p \in \text{bot unit-cube}$ **by** (*simp add: top-shift-down-eq-bot*)

thus $p \in \bigcup (\text{image bot } C)$ **using** *bot-unit-cube-cover-by-bot* **by** *blast*

```

next
  fix c p assume c ∈ C bot c ∩ top ?b ≠ {} p ∈ bot c
  with ⟨bot c ∩ top ?b ≠ {}⟩ have min z c = 0
  by(metis top-shift-down-eq-bot bot-intersect-by-side min-unit-cube-0)
  hence pz p = 0 by(metis ⟨p ∈ bot c⟩ in-bot-by-side)
  moreover from ⟨p ∈ bot c⟩ have px p ∈ {0..<1} py p ∈ {0..<1}
  using ⟨c ∈ C⟩ subset-unit-cube-by-side in-bot-by-side by blast+
  ultimately have p ∈ bot unit-cube by(metis min-unit-cube-0 in-bot-by-side
side-unit-cube)
  thus p ∈ top ?b by(metis top-shift-down-eq-bot)
next
  fix c assume c ∈ C c ≠ ?b
  hence width c ≠ 1 by(metis width-ne-one)
  thus width c ≠ width ?b by(simp add: unit-cube-def shift-down-def)
qed
qed

```

5 Minimum cube on hole is interior

```

context
  fixes h assumes hole: is-hole h
begin

```

For this section, we fix a hole h , and define $cmin$ to be the smallest cube on this hole. Theorem *hole* refers to this fact. The goal of this section is then to show that $cmin$ itself is a hole.

5.1 Definition: Minimum cube on h

$cmin$ is the smallest cube on the hole h

```

definition cmin:: cube
  where cmin = (ARG-MIN width c . c ∈ filter-on-hole h)

```

```

lemma arg-min-exist: [finite C'; C' ≠ {}] ⇒ (ARG-MIN width c . c ∈ C') ∈ C'
  by(metis arg-min-on-def arg-min-if-finite(1))

```

This lemma also shows that $local.cmin$ exists

```

lemma cmin-on-h: cmin ∈ filter-on-hole h
  unfolding cmin-def by(metis hole arg-min-exist on-hole-finite on-hole-non-empty
)

```

```

lemma cmin-valid[simp]: is-valid cmin
  by(metis cmin-on-h valid-if-on-hole)

```

```

lemma arg-min-minimal: [finite C'; c ∈ C'] ⇒ width (ARG-MIN width c . c ∈
C') ≤ width c
  by(metis arg-min-least arg-min-on-def empty-iff)

```

lemma *cmin-minimal*: $c \in \text{filter-on-hole } h \implies \text{width } cmin \leq \text{width } c$
unfolding *cmin-def* **by**(*metis hole arg-min-minimal on-hole-finite*)

lemma *cmin-minimal-strict*:

assumes $c \in \text{filter-on-hole } h$ $c \neq cmin$

shows $\text{width } cmin < \text{width } c$

proof –

have $\text{width } cmin \leq \text{width } c$ **using** *assms(1)* **by**(*rule cmin-minimal*)

moreover have $\text{width } cmin \neq \text{width } c$

by (*metis assms on-hole-inj-on-width assms(2) cmin-on-h hole inj-on-contrad*)

ultimately show $\text{width } cmin < \text{width } c$ **by** *simp*

qed

lemma *cmin-max-z-neq-one*: $\max z cmin < 1$

proof –

let $?On-h = \text{filter-on-hole } h$

have $\text{bot } cmin \subset \text{top } h$ **by**(*metis hole cmin-on-h strict-subset-if-on-hole*)

then obtain c **where** $c \in ?On-h$ $c \neq cmin$ **using** *hole obtain-cube-if-in-hole* **by**
blast

hence $\min z cmin = \min z c$ **by**(*metis cmin-on-h $\langle c \in ?On-h \rangle$ z-eq-if-on-hole*)

hence $\max z cmin < \max z c$ **by**(*simp add: $\langle c \in ?On-h \rangle \langle c \neq cmin \rangle$ cmin-minimal-strict*)

moreover have $\text{side } z c \subseteq \{0..<1\}$

by(*metis $\langle c \in ?On-h \rangle$ dissection-if-on-hole subset-unit-cube-by-side*)

hence $\max z c \leq 1$ **using** $\langle c \in ?On-h \rangle$ *valid-if-on-hole* **by** (*simp add: less-le-not-le*)

ultimately show $\max z cmin < 1$ **by** *simp*

qed

5.2 Minimum cube on hole is interior

All squares on the boundary of h

definition *is-on-boundary* :: $\text{axis} \Rightarrow \text{cube} \Rightarrow \text{bool}$ **where**

is-on-boundary $ax\ c \iff \min ax\ h = \min ax\ c \vee \max ax\ h = \max ax\ c$

Shows that IF *local.cmin* is on a boundary ax , then we find some ax -coordinate r , which is further from the boundary than the edge of *local.cmin*, but closer than the edge of any other cube sufficiently close to the boundary.

lemma *cmin-on-boundary*:

assumes *is-on-boundary* $ax\ cmin$ $ax \in \{x, y\}$

shows $\exists r$.

$r \in (\text{side } ax\ h - (\text{side } ax\ cmin)) \wedge$

$(\forall c \in \text{filter-on-hole } h . c \neq cmin \implies \text{side } ax\ cmin \cap \text{side } ax\ c \neq \{ \} \implies r \in \text{side } ax\ c)$

proof –

let $?On-h = \text{filter-on-hole } h$

let $?c-2nd-min = \text{ARG-MIN width } c . c \in (\text{Set.remove } cmin\ ?On-h)$

have $\text{bot } cmin \subset \text{top } h$ **by**(*metis hole cmin-on-h strict-subset-if-on-hole*)

hence $\text{Set.remove } cmin\ ?On-h \neq \{ \}$ **using** *hole obtain-cube-if-in-hole* **by** *fastforce*

moreover have *finite* ($\text{Set.remove } cmin\ ?On-h$)

by (*metis hole on-hole-finite remove-def finite-Diff*)

ultimately have $?c\text{-}2\text{nd}\text{-}min \in \text{Set.remove } cmin \ ?On\text{-}h$ **by** $(metis \text{arg}\text{-}min\text{-}exist)$
hence $?c\text{-}2\text{nd}\text{-}min \in ?On\text{-}h$ **and** $width \ cmin < width \ ?c\text{-}2\text{nd}\text{-}min$
by $(simp, \text{simp add: } cmin\text{-}minimal\text{-}strict)$
then obtain ε **where** $\varepsilon > 0$ $width \ cmin + \varepsilon < width \ ?c\text{-}2\text{nd}\text{-}min$
by $(metis \text{field}\text{-}le\text{-}epsilon \ \text{less}\text{-}le\text{-}not\text{-}le \ \text{nle}\text{-}le)$
have $\varepsilon\text{-}prop: \forall c \in ?On\text{-}h. \ cmin \neq c \longrightarrow width \ cmin + \varepsilon < width \ c$
proof $(intro \ ballI \ impI)$
fix c **assume** $c \in ?On\text{-}h \ cmin \neq c$
hence $c \in \text{Set.remove } cmin \ ?On\text{-}h$ **by** $auto$
hence $width \ ?c\text{-}2\text{nd}\text{-}min \leq width \ c$
using $\langle \text{Set.remove } cmin \ ?On\text{-}h \neq \{\} \rangle \langle \text{finite } (\text{Set.remove } cmin \ ?On\text{-}h) \rangle$
 $arg\text{-}min\text{-}minimal$
by $blast$
thus $width \ cmin + \varepsilon < width \ c$ **using** $\langle width \ cmin + \varepsilon < width \ ?c\text{-}2\text{nd}\text{-}min \rangle$
by $simp$
qed
then show $?thesis$ **using** $assms(1)$ **unfolding** $is\text{-}on\text{-}boundary\text{-}def$
proof $(elim \ disjE)$
assume $min \ ax \ h = min \ ax \ cmin$
let $?r = max \ ax \ cmin + \varepsilon$
show $?thesis$
proof $(intro \ exI \ conjI \ ballI \ impI)$
have $?r < min \ ax \ cmin + width \ ?c\text{-}2\text{nd}\text{-}min$ **using** $\langle width \ cmin + \varepsilon < width \ ?c\text{-}2\text{nd}\text{-}min \rangle$ **by** $auto$
also have $\dots = min \ ax \ h + width \ ?c\text{-}2\text{nd}\text{-}min$ **using** $\langle min \ ax \ h = min \ ax \ cmin \rangle$ **by** $simp$
finally have $?r < max \ ax \ h$
using $\langle ?c\text{-}2\text{nd}\text{-}min \in ?On\text{-}h \rangle \ \text{hole} \ \text{width}\text{-}on\text{-}hole\text{-}lt\text{-}width\text{-}hole$ **by** $fastforce$
moreover have $min \ ax \ cmin < max \ ax \ cmin$ **using** $assms(1)$ **by** $(simp \ \text{add: } min\text{-}lt\text{-}max)$
hence $min \ ax \ h < ?r$ **using** $\langle \varepsilon > 0 \rangle \langle min \ ax \ h = min \ ax \ cmin \rangle$ **by** $linarith$
ultimately have $?r \in \text{side } ax \ h$ **by** $simp$
moreover from $\langle \varepsilon > 0 \rangle$ **have** $?r \notin \text{side } ax \ cmin$ **by** $simp$
ultimately show $?r \in \text{side } ax \ h - \text{side } ax \ cmin$ **by** $simp$
next
fix c **assume** $c \in ?On\text{-}h \ c \neq cmin \ \text{side } ax \ cmin \cap \text{side } ax \ c \neq \{\}$
from $\langle \text{side } ax \ cmin \cap \text{side } ax \ c \neq \{\} \rangle$ **have** $min \ ax \ c < max \ ax \ cmin$ **by** $auto$
also have $\dots < ?r$ **using** $\langle \varepsilon > 0 \rangle$ **by** $simp$
finally have $min \ ax \ c \leq ?r$ **by** $simp$
moreover have $bot \ c \subseteq top \ h$ **by** $(metis \langle c \in ?On\text{-}h \rangle \ \text{subset}\text{-}if\text{-}on\text{-}hole)$
hence $side \ ax \ c \subseteq side \ ax \ h$ **using** $\langle c \in ?On\text{-}h \rangle \ \text{assms}(2) \ \text{bot}\text{-}top\text{-}subset\text{-}by\text{-}side$
by $fastforce$
hence $min \ ax \ h \leq min \ ax \ c$ **using** $\langle c \in ?On\text{-}h \rangle$ **by** $fastforce$
hence $min \ ax \ cmin \leq min \ ax \ c$ **using** $\langle min \ ax \ h = min \ ax \ cmin \rangle$ **by** $simp$
hence $?r < max \ ax \ c$ **using** $\langle c \in ?On\text{-}h \rangle \langle c \neq cmin \rangle \ \varepsilon\text{-}prop$ **by** $auto$
ultimately show $?r \in \text{side } ax \ c$ **by** $simp$
qed
next
assume $max \ ax \ h = max \ ax \ cmin$

```

let ?r = min ax cmin - ε
show ?thesis
proof(intro exI conjI ballI impI)
  have min ax h = max ax h - width h by simp
  also have ... < max ax h - width ?c-2nd-min
    using hole width-on-hole-lt-width-hole ‹?c-2nd-min ∈ ?On-h› by simp
  also have ... = max ax cmin - width ?c-2nd-min using ‹max ax h = max
ax cmin› by simp
  also have ... < ?r using ‹width cmin + ε < width ?c-2nd-min› by argo
  finally have min ax h < ?r by simp
  moreover have min ax cmin < max ax cmin using assms(1) by(simp add:
min-lt-max)
  hence ?r < max ax h using ‹max ax h = max ax cmin› ‹ε > 0› by argo
  ultimately have ?r ∈ side ax h by simp
  moreover have ?r ∉ side ax cmin using ‹ε > 0› by simp
  ultimately show ?r ∈ side ax h - side ax cmin by simp
next
fix c assume c ∈ ?On-h c ≠ cmin side ax cmin ∩ side ax c ≠ {}
have ?r < min ax cmin using ‹ε > 0› by simp
also have ... < max ax c using ‹side ax cmin ∩ side ax c ≠ {}› by simp
finally have ?r < max ax c by simp
moreover have bot c ⊆ top h by(metis ‹c ∈ ?On-h› subset-if-on-hole)
hence side ax c ⊆ side ax h
  using ‹c ∈ ?On-h› assms(2) bot-top-subset-by-side by fastforce
hence max ax c ≤ max ax h using ‹c ∈ ?On-h› valid-if-on-hole by fastforce
hence max ax c ≤ max ax cmin using ‹max ax h = max ax cmin› by simp
hence min ax c ≤ ?r using ‹c ∈ ?On-h› ‹c ≠ cmin› ε-prop by auto
ultimately show ?r ∈ side ax c by simp
qed
qed
qed

```

Using the previous lemma, we show that *local.cmin* being on the boundary leads to a contradiction

lemma *cmin-not-on-boundary-by-axis*:

```

assumes ax ∈ {x, y}
shows ¬is-on-boundary ax cmin
proof(rule ccontr, unfold not-not)
let ?On-h = filter-on-hole h
assume is-on-boundary ax cmin
then obtain r where r ∈ (side ax h - side ax cmin) and
  r-prop: (∀ c ∈ ?On-h . c ≠ cmin → side ax cmin ∩ side ax c ≠ {}) → r ∈
side ax c)
  by(metis assms cmin-on-boundary)
hence r ∈ side ax h and r ∉ side ax cmin by blast+
let ?ax2 = if ax = x then y else x
let ?p1 = if ax = x then (px=r, py=min y cmin, pz=max z h) else (px=min x
cmin, py=r, pz=max z h)
have side ?ax2 cmin ⊆ side ?ax2 h using cmin-on-h side-subset-if-on-hole by

```

simp
hence $\min ?ax2\ cmin \in side\ ?ax2\ h$ **using** *min-in-side* **by** *fastforce*
hence $?p1 \in top\ h$ **using** $\langle r \in side\ ax\ h \rangle$ *in-top-by-side* **assms** **by** (*simp split: if-splits*)
then obtain $c1$ **where** $?p1 \in bot\ c1$ **and** $c1\text{-on-h}: c1 \in ?On\text{-}h$
using *hole obtain-cube-if-in-hole* **by** *blast*
hence $cmin \neq c1$ **using** $\langle r \notin side\ ax\ cmin \rangle$ *assms in-bot-by-side* **by** (*auto split: if-splits*)
hence $width\ cmin < width\ c1$ **by**(*metis* $\langle c1 \in ?On\text{-}h \rangle$ *cmin-minimal-strict*)
hence $\neg side\ ?ax2\ c1 \subseteq side\ ?ax2\ cmin$ **using** *c1-on-h valid-if-on-hole* **by** *fast-force*
then obtain s **where** $s \in side\ ?ax2\ c1$ $s \notin side\ ?ax2\ cmin$ **by** *blast*
let $?p2 = \text{if } ax = x \text{ then } (\!|px = \min\ x\ cmin, py = s, pz = \max\ z\ h|\!) \text{ else } (\!|px = s, py = \min\ y\ cmin, pz = \max\ z\ h|\!)$
have $side\ ax\ cmin \subseteq side\ ax\ h$ **using** *assms cmin-on-h side-subset-if-on-hole* **by** *blast*
hence $\min\ ax\ cmin \in side\ ax\ h$ **using** *min-in-side* **by** *fastforce*
moreover have $side\ ?ax2\ c1 \subseteq side\ ?ax2\ h$ **using** $\langle c1 \in ?On\text{-}h \rangle$ *side-subset-if-on-hole* **by** *simp*
hence $s \in side\ ?ax2\ h$ **using** $\langle s \in side\ ?ax2\ c1 \rangle$ **by** *blast*
ultimately have $?p2 \in top\ h$ **using** *in-top-by-side* **assms** **by** (*simp split: if-splits*)
then obtain $c2$ **where** $?p2 \in bot\ c2$ **and** $c2\text{-on-h}: c2 \in ?On\text{-}h$
using *hole obtain-cube-if-in-hole* **by** *blast*
let $?p3 = \text{if } ax = x \text{ then } (\!|px = r, py = s, pz = \max\ z\ h|\!) \text{ else } (\!|px = s, py = r, pz = \max\ z\ h|\!)$
— We show that *if* $ax = x$ *then* $(\!|px = r, py = s, pz = \max\ z\ h|\!)$ *else* $(\!|px = s, py = r, pz = \max\ z\ h|\!)$ is in the intersection of $c1$ and $c2$, but $p2$ is only in $c2$, contradicting disjointness
have $?p2 \notin bot\ cmin$ **using** $\langle s \notin side\ ?ax2\ cmin \rangle$ *in-bot-by-side* **by** (*auto split: if-splits*)
hence $cmin \neq c2$ **using** $\langle ?p2 \in bot\ c2 \rangle$ **by** *blast*
hence $r \in side\ ax\ c2$ **using** *r-prop c2-on-h min-in-side* $\langle ?p2 \in bot\ c2 \rangle$ *in-bot-by-side* **assms**
by *fastforce*
hence $?p3 \in bot\ c2$ **using** $\langle ?p2 \in bot\ c2 \rangle$ *in-bot-by-side* **assms** **by** (*simp split: if-splits*)
moreover have $?p3 \in bot\ c1$ **using** $\langle ?p1 \in bot\ c1 \rangle \langle s \in side\ ?ax2\ c1 \rangle$ *in-bot-by-side* **by** *auto*
ultimately have $c1 = c2$ **using** *c1-on-h c2-on-h eq-iff-bot-intersect-if-on-hole* **by** *blast*
moreover from $\langle s \notin side\ ?ax2\ cmin \rangle$ **have** $?p2 \notin to\text{-}set\ cmin$ **using** *in-set-by-side* **by** *auto*
hence $cmin \neq c2$ **using** *eq-iff-side-eq-if-on-hole c2-on-h* $\langle ?p2 \in bot\ c2 \rangle$ *bot-subset*

by *blast*
hence $to\text{-}set\ cmin \cap to\text{-}set\ c2 = \{\}$ **using** *cmin-on-h c2-on-h eq-iff-intersect* **by** *simp*
hence $side\ ?ax2\ cmin \cap side\ ?ax2\ c2 = \{\}$ **using** $\langle ?p2 \in bot\ c2 \rangle \langle cmin \neq c2 \rangle$ *c2-on-h* **assms** *cmin-on-h in-bot-by-side eq-iff-side-intersect-if-on-hole* **by** *fast-*

force
hence $\min ?ax2 \text{ cmin} \notin \text{side } ?ax2 \text{ c2}$ **using** *cmin-valid min-in-side* **by** *blast*
hence $?p1 \notin \text{bot } c2$ **using** *assms in-bot-by-side* **by** *fastforce*
hence $c1 \neq c2$ **using** $\langle ?p1 \in \text{bot } c1 \rangle$ **by** *blast*
ultimately show *False* **by** *blast*
qed

Previous result, written as inequalities instead

proposition *cmin-not-on-boundary*:

$\min x h < \min x \text{ cmin} \wedge \max x \text{ cmin} < \max x h$
 $\wedge \min y h < \min y \text{ cmin} \wedge \max y \text{ cmin} < \max y h$

proof –

have $\text{side } x \text{ cmin} \subseteq \text{side } x h \wedge \text{side } y \text{ cmin} \subseteq \text{side } y h$
using *cmin-on-h side-subset-if-on-hole* **by** *blast*
hence $\min x h \leq \min x \text{ cmin} \wedge \max x \text{ cmin} \leq \max x h \wedge$
 $\min y h \leq \min y \text{ cmin} \wedge \max y \text{ cmin} \leq \max y h$ **using** *cmin-valid* **by** *auto*
thus $\min x h < \min x \text{ cmin} \wedge \max x \text{ cmin} < \max x h \wedge$
 $\min y h < \min y \text{ cmin} \wedge \max y \text{ cmin} < \max y h$
using *cmin-not-on-boundary-by-axis unfolding is-on-boundary-def* **by** *force*
qed

6 Minimum cube of hole induces hole on top

The main result of this proof - the minimum cube on a hole is itself a hole!

proposition *hole-cmin*:

shows *is-hole cmin*

proof –

let $?On\text{-cmin} = \text{filter-on-hole } \text{cmin}$

show $?thesis$ **unfolding** *is-hole-def*

proof(*intro conjI ballI impI*)

show *is-valid cmin* **by** *simp*

next

show $\text{top } \text{cmin} \subseteq \bigcup (\text{image } \text{bot } C)$

by(*metis top-cover-by-bot cmin-max-z-neq-one cmin-on-h dissection-if-on-hole*)

next

fix c **assume** $c \in C \ c \neq \text{cmin}$

thus $\text{width } c \neq \text{width } \text{cmin}$

by(*metis dissection dissection-if-on-hole cmin-on-h inj-onD is-dissection-def*)

next

fix c **assume** $c \in C \ \text{bot } c \cap \text{top } \text{cmin} \neq \{\}$

hence $c\text{-valid}$: *is-valid* c **by** *simp*

show $\text{bot } c \subseteq \text{top } \text{cmin}$

proof(*rule ccontr*)

assume $\neg \text{bot } c \subseteq \text{top } \text{cmin}$

from $\langle \text{bot } c \cap \text{top } \text{cmin} \neq \{\} \rangle$ **have** $\min z c = \max z \text{ cmin}$ **using** *bot-top-intersect-by-side*

by *blast*

with $\langle \neg \text{bot } c \subseteq \text{top } \text{cmin} \rangle$ **obtain** ax **where** ax : $ax \in \{x, y\} \ \neg \text{side } ax \ c \subseteq$

side ax cmin
using *bot-top-subset-by-side c-valid* **by** (*meson insert-iff*)
let $?ax2 = \text{if } ax=x \text{ then } y \text{ else } x$
have $?ax2 \in \{x,y\}$ **by** *simp*
have $\min ax h < \min ax cmin \max ax cmin < \max ax h$
using *ax(1) cmin-not-on-boundary* **by** *blast+*
moreover **have** $\text{side } ax c \cap \text{side } ax cmin \neq \{\}$
using $\langle \text{bot } c \cap \text{top } cmin \neq \{\} \rangle$ *ax(1) bot-top-intersect-by-side* **by** *blast*
ultimately **have** $\text{side } ax c \cap \text{side } ax h - \text{side } ax cmin \neq \{\}$ **using** *ax(2)* **by**
fastforce
then obtain a **where** *a-prop*: $a \in \text{side } ax c \cap \text{side } ax h$ $a \notin \text{side } ax cmin$ **by**
blast
moreover from $\langle \text{bot } c \cap \text{top } cmin \neq \{\} \rangle$ **have** $\text{side } ?ax2 c \cap \text{side } ?ax2 cmin$
 $\neq \{\}$
using *bot-top-intersect-by-side* $\langle ?ax2 \in \{x,y\} \rangle$ **by** *simp*
then obtain b **where** *b-prop*: $b \in \text{side } ?ax2 c \cap \text{side } ?ax2 cmin$ **by** *blast*
hence *b-prop-2*: $b \in \text{side } ?ax2 c \cap \text{side } ?ax2 h$
using $\langle ?ax2 \in \{x,y\} \rangle$ *cmin-on-h side-subset-if-on-hole* **by** *blast*
let $?x0 = \text{if } ax=x \text{ then } a \text{ else } b$ **and** $?y0 = \text{if } ax=y \text{ then } a \text{ else } b$
have *xy0-props*: $?x0 \in \text{side } x c \cap \text{side } x h$ $?y0 \in \text{side } y c \cap \text{side } y h$
 $?x0 \notin \text{side } x cmin \vee ?y0 \notin \text{side } y cmin$
proof –
show $?x0 \in \text{side } x c \cap \text{side } x h$ **using** *a-prop b-prop ax(1)* $\langle ?ax2 \in \{x,y\} \rangle$
using *b-prop-2* **by** *presburger*
show $?y0 \in \text{side } y c \cap \text{side } y h$ **using** *a-prop b-prop-2 ax(1)* **by** *fastforce*
show $?x0 \notin \text{side } x cmin \vee ?y0 \notin \text{side } y cmin$ **using** *a-prop ax(1)* **by**
fastforce
qed
let $?pd = (\text{px}=?x0, \text{py}=?y0, \text{pz}=\max z h)$
let $?pu = (\text{px}=?x0, \text{py}=?y0, \text{pz}=\min z c)$
have $\max z h = \min z cmin$ **by** (*metis cmin-on-h min-z-eq-max-z-hole-if-on-hole*)
hence $?pd = (\text{px}=?x0, \text{py}=?y0, \text{pz}=\min z cmin)$ **by** *simp*
moreover **have** $\min z cmin < \min z c$ **using** *min-lt-max* $\langle \min z c = \max z$
 $cmin \rangle$ **by** *simp*
hence $\min z cmin \notin \text{side } z c$ **using** *in-set-by-side* **by** *simp*
ultimately **have** $?pd \notin \text{to-set } c$ **by** (*simp add: in-set-by-side*)
have $?pu \in \text{bot } c$ **using** *c-valid xy0-props in-bot-by-side* **by** *simp*
hence $?pu \in \text{to-set } c$ **using** *bot-subset* **by** *blast*
have $?pd \notin \text{bot } cmin$ **using** $\langle ?x0 \notin \text{side } x cmin \vee ?y0 \notin \text{side } y cmin \rangle$
in-bot-by-side **by** *auto*
have $?pd \in \text{top } h$ **using** *valid-if-hole xy0-props in-top-by-side* **by** *simp*
then obtain c' **where** $c' \in \text{filter-on-hole } h$ $?pd \in \text{bot } c'$
using *hole obtain-cube-if-in-hole* **by** *blast*
hence $c' \neq cmin$ **using** $\langle ?pd \notin \text{bot } cmin \rangle$ **by** *blast*
hence $\text{width } cmin < \text{width } c'$
using $\langle c' \in \text{filter-on-hole } h \rangle$ *cmin-minimal-strict* **by** *simp*
moreover **have** $\min z cmin = \min z c'$
by (*metis cmin-on-h* $\langle c' \in \text{filter-on-hole } h \rangle$ *z-eq-if-on-hole*)
ultimately **have** $\max z cmin < \max z c'$ **by** *simp*

hence $\min z c < \max z c'$ **by** (*metis* $\langle \min z c = \max z cmin \rangle$)
hence $\min z c \in \text{side } z c'$ **using** $\langle \min z cmin < \min z c \rangle \langle \min z cmin = \min z c' \rangle$ **by** *auto*
hence $?pu \in \text{to-set } c'$ **using** $\langle c' \in \text{filter-on-hole } h \rangle \langle ?pd \in \text{bot } c' \rangle$
in-bot-by-side in-set-by-side $\langle \min z c = \max z cmin \rangle$ **by** *simp*
hence $\text{to-set } c \cap \text{to-set } c' \neq \{\}$ **using** $\langle ?pu \in \text{to-set } c \rangle$ **by** *blast*
hence $c = c'$ **using** $\langle c' \in \text{filter-on-hole } h \rangle \langle c \in C \rangle$ *eq-iff-intersect* **by** *auto*
moreover **have** $c \neq c'$ **using** $\langle ?pd \notin \text{to-set } c \rangle \langle ?pd \in \text{bot } c' \rangle$ **using** *bot-subset*
by *blast*
ultimately show *False* **by** *blast*
qed
qed
qed

The main purpose of the previous result: From the proposition, *hole-cmin* when given the hole h induce another hole h' (i.e., *local.cmin*), which is in C and is strictly smaller.

lemma *recursive-step*: $\exists h'. h' \in C \wedge \text{is-hole } h' \wedge \text{width } h' < \text{width } h$

proof–

show *?thesis*

proof(*intro exI conjI*)

show $cmin \in C$ **using** *cmin-on-h* **by** *force*

show *is-hole cmin* **by**(*simp add: hole-cmin*)

show $\text{width } cmin < \text{width } h$ **by**(*simp add: hole cmin-on-h width-on-hole-lt-width-hole*)

qed

qed

Here we end the context in which h is some fixed hole (and hence also the specific *local.cmin*)

end

7 The main result

We combine the previous lemmas inductively as follows: 0: Start with the bottom of *unit-cube*, which we showed is a hole. n: For each hole, take the minimum cube on this hole, which is then a new hole, strictly smaller, and in C . Hence, C is infinite.

definition *next-hole*:: *cube* \Rightarrow *cube* **where**

next-hole $h = (\text{SOME } h' . h' \in C \wedge \text{is-hole } h' \wedge \text{width } h' < \text{width } h)$

lemma *next-hole-exist*: *is-hole* h

\Rightarrow *next-hole* $h \in C \wedge \text{is-hole } (\text{next-hole } h) \wedge \text{width } (\text{next-hole } h) < \text{width } h$

unfolding *next-hole-def* **by** (*rule someI-ex*) (*rule recursive-step*)

For following proof, we want the image of *nth-hole* to be contained in C , hence we start at $1::'a$ (= *Suc* 0). *nth-hole* is a function from \mathbb{N} to C

definition *nth-hole* :: *nat* \Rightarrow *cube* **where**

$nth-hole\ n = (next-hole \ \overset{\sim}{\sim} \ Suc\ n) \ (shift-down\ unit-cube)$

Each cube in the image of $local.nth-hole$ is a hole

lemma $nth-hole-is-hole: is-hole\ (nth-hole\ n)$
proof($induction\ n$)
 case 0
 have $nth-hole\ 0 = next-hole\ (shift-down\ unit-cube)$ **by**($simp\ add: nth-hole-def$)
 moreover have $is-hole\ (shift-down\ unit-cube)$ **by**($rule\ hole-unit-cube$)
 ultimately show $?case$ **by**($simp\ add: next-hole-exist$)
next
 case ($Suc\ n$)
 then show $?case$ **by**($simp\ add: nth-hole-def\ next-hole-exist$)
qed

$uminus$ is $uminus$, and $strict-mono$ means strictly increasing (not strictly monotonous, as the name might suggest)

lemma $nth-hole-strict-decreasing: strict-mono\ (uminus\ \circ\ width\ \circ\ nth-hole)$
proof($rule\ iffD2, rule\ strict-mono-Suc-iff, intro\ allI$)
 fix n
 have $is-hole\ (nth-hole\ n)$ **by** ($rule\ nth-hole-is-hole$)
 thus $(uminus\ \circ\ width\ \circ\ nth-hole)\ n < (uminus\ \circ\ width\ \circ\ nth-hole)\ (Suc\ n)$
 by($simp\ add: nth-hole-def\ next-hole-exist$)
qed

$local.nth-hole$ is injective

lemma $nth-hole-inj : inj\ nth-hole$
proof –
 have $strict-mono\ (uminus\ \circ\ width\ \circ\ nth-hole)$ **by**($rule\ nth-hole-strict-decreasing$)
 hence $inj\ (uminus\ \circ\ width\ \circ\ nth-hole)$ **by**($simp\ add: strict-mono-imp-inj-on$)
 thus $inj\ nth-hole$ **by**($simp\ add: inj-on-imageI2$)
qed

The image (range) of $local.nth-hole$ is contained in C

lemma $nth-hole-in: nth-hole\ n \in C$
proof($cases\ n$)
 case 0
 then show $?thesis$ **by**($simp\ add: hole-unit-cube\ nth-hole-def\ next-hole-exist$)
next
 case ($Suc\ nat$)
 then obtain m **where** $n = Suc\ m$ **by** $simp$
 have $is-hole\ (nth-hole\ m)$ **by**($simp\ add: nth-hole-is-hole$)
 hence $next-hole\ (nth-hole\ m) \in C$ **by**($simp\ add: next-hole-exist$)
 thus $nth-hole\ n \in C$ **by**($simp\ add: nth-hole-def\ \langle n = Suc\ m \rangle$)
qed

Same as previous lemma, but written with a quantifier

lemma $nth-hole-in-forall: \forall n . nth-hole\ n \in C$
by($simp\ add: nth-hole-in$)

The assumption made in this context (*is-dissection C*) leads to *False* (since *local.nth-hole* generates an infinite subset of *C*)

theorem *false-if-dissection: False*

proof –

have *inj nth-hole* **by**(*rule nth-hole-inj*)

moreover have $\forall n . \text{nth-hole } n \in C$ **by**(*rule nth-hole-in-forall*)

 — *local.nth-hole* injective, with infinite domain \mathbb{N} , and image contained in *C* together implies that *C* is infinite

ultimately have *infinite C* **by**(*metis finite-subset image-subset-iff range-inj-infinite*)

moreover have *finite C* **using** *dissection unfolding is-dissection-def* **by** *metis*

ultimately show *False* **by** *metis*

qed

end — Here we end the *is-dissection C* context

 Main result (spelling out the definition of *is-dissection*).

theorem *dissection-does-not-exist:*

$\nexists C. (\forall c \in C. \text{is-valid } c)$

$\wedge \text{disjoint (image to-set } C)$

$\wedge \bigcup (\text{image to-set } C) = \text{to-set unit-cube}$

$\wedge \text{inj-on width } C$

$\wedge \text{card } C \geq 2$

$\wedge \text{finite } C$

by(*metis is-dissection-def false-if-dissection*)

end

References

- [1] Formalizing 100 Theorems. <https://www.cs.ru.nl/~freek/100/>, accessed 2024-11-22.
- [2] J. E. Littlewood and B. Bollobas. *Littlewood's Miscellany*. Cambridge University Press, Cambridge [Cambridgeshire] ; New York, rev. ed edition, 1986.
- [3] F. van Doorn. Archive.Wiedijk100Theorems.CubingACube. https://leanprover-community.github.io/mathlib4_docs/Archive/Wiedijk100Theorems/CubingACube.html#Theorems100.%C2%AB82%C2%BB.cannot_cube_a_cube, accessed 2024-11-22.