

Distributed Distinct Elements

Emin Karayel

May 26, 2024

Abstract

This entry formalizes a randomized cardinality estimation data structure with asymptotically optimal space usage. It is inspired by the streaming algorithm presented by Błasiok [3] in 2018. His work closed the gap between the best-known lower bound and upper bound after a long line of research started by Flajolet and Martin [4] in 1984 and was the first to apply expander graphs (in addition to hash families) to the problem. The formalized algorithm has two improvements compared to the algorithm by Błasiok. It supports operation in parallel mode, and it relies on a simpler pseudo-random construction avoiding the use of code based extractors.

Contents

1	Introduction	2
2	Preliminary Results	2
3	Blind	8
4	Balls and Bins	8
5	Tail Bounds for Expander Walks	41
6	Inner Algorithm	52
7	Accuracy without cutoff	71
8	Cutoff Level	89
9	Accuracy with cutoff	97
10	Outer Algorithm	102

1 Introduction

The algorithm is described as functional data structures, given a seed which needs to be chosen uniformly from a initial segment of the natural numbers and globally, there are three functions:

- single - given the seed and an element from the universe computes a sketch for that singleton set
- merge - computes a sketch based on two input sketches and returns a sketch representing the union set
- estimate - computes an estimate for the cardinality of the set represented by a sketch

The main point is that a sketch requires $\mathcal{O}(\delta^{-2} \ln(\varepsilon^{-1}) + \ln n)$ space where n is the universe size, δ is the desired relative accuracy and ε is the desired failure probability. Note that it is easy to see that an exact solution would necessarily require $\mathcal{O}(n)$ bits.

The algorithm is split into two parts an inner algorithm, described in Section 6, which itself is already a full cardinality estimation algorithm, however its space usage is below optimal. The outer algorithm is introduced in Section 10, which runs mutiple copies of the inner algorithm with carefully chosen inner parameters.

As mentioned in the abstract the algorithm is inspired by the solution to the streaming version of the problem by Błasiok [3] in 2020. His work builds on a long line of reasrch starting in 1985 [4, 1, 2, 7, 11, 5].

In an earlier AFP entry [9] I have formalized an earlier cardinality estimation algorithm based on the work by Bar-Yossef et al. [2] in 2002. Since then I have addressed the existence of finite fields for higher prime powers and expander graphs [8, 10]. Building on these results, the formalization of this more advanced solution presented here became possible.

The solution described here improves on the algorithms described by Błasiok in two ways (without comprising its optimal space usage). It can be used in a parallel mode of operation. Moreover the pseudo-random construction used is simpler than the solution described by Błasiok — who uses an extractor based on Parvaresh-Vardy codes [6] to sample random walks in an expander graph, which are then sub-sampled and then the walks are used to sample seeds for hash functions. In the solution presented here neither the sub-sampling step nor the extractor is needed, instead a two-stage expander construction is used, this means that the nodes of the first expander correspond to the walks in a second expander graph. The latters nodes correspond to seeds of hash functions (as in Błasiok’s solution).

The modification needed to support a parallel mode of operation is a change in the failure strategy of the solution presented in Kane et al., which is the event when the data in the sketch reequires too much space. The main issue is that in the parallel case the number of states the algorithm might reach is not bounded by the universe size and thus an estimate they make for the probability of the failure event does not transfer to the parallel case. To solve that the algorithm in this work is more conservative. Instead of failing out-right it instead increases a cutoff threshold. For which it is then possible to show an upper estimate independent of the number of reached states.

2 Preliminary Results

This section contains various short preliminary results used in the sections below.

theory *Distributed-Distinct-Elements-Preliminary*

imports

Frequency-Moments.Frequency-Moments-Preliminary-Results

Universal-Hash-Families.Universal-Hash-Families-More-Product-PMF

Median-Method.Median

Expander-Graphs.Extra-Congruence-Method

Expander-Graphs.Constructive-Chernoff-Bound

Frequency-Moments.Landau-Ext

Stirling-Formula.Stirling-Formula

begin

unbundle *intro-cong-syntax*

lemma *pmf-rev-mono*:

assumes $\bigwedge x. x \in \text{set-pmf } p \implies x \notin Q \implies x \notin P$
shows $\text{measure } p \ P \leq \text{measure } p \ Q$
using *assms* **by** (*intro pmf-mono*) *blast*

lemma *pmf-exp-mono*:

fixes $f \ g :: 'a \Rightarrow \text{real}$
assumes $\text{integrable } (\text{measure-pmf } p) \ f \ \text{integrable } (\text{measure-pmf } p) \ g$
assumes $\bigwedge x. x \in \text{set-pmf } p \implies f \ x \leq g \ x$
shows $\text{integral}^L (\text{measure-pmf } p) \ f \leq \text{integral}^L (\text{measure-pmf } p) \ g$
using *assms* **by** (*intro integral-mono-AE AE-pmfI*) *auto*

lemma *pmf-markov*:

assumes $\text{integrable } (\text{measure-pmf } p) \ f \ c > 0$
assumes $\bigwedge x. x \in \text{set-pmf } p \implies f \ x \geq 0$
shows $\text{measure } p \ \{\omega. f \ \omega \geq c\} \leq (\int \omega. f \ \omega \ \partial p) / c$ (**is** $?L \leq ?R$)

proof –

have $a:AE \ \omega \ \text{in } (\text{measure-pmf } p). \ 0 \leq f \ \omega$
by (*intro AE-pmfI assms(3)*)
have $b:\{\} \in \text{measure-pmf.events } p$
unfolding *assms(1)* **by** *simp*

have $?L = \mathcal{P}(\omega \ \text{in } (\text{measure-pmf } p). \ f \ \omega \geq c)$
using *assms(1)* **by** *simp*

also have $\dots \leq ?R$

by (*intro integral-Markov-inequality-measure[OF - b] assms a*)

finally show *?thesis* **by** *simp*

qed

lemma *pair-pmf-prob-left*:

$\text{measure-pmf.prob } (\text{pair-pmf } A \ B) \ \{\omega. P \ (fst \ \omega)\} = \text{measure-pmf.prob } A \ \{\omega. P \ \omega\}$ (**is** $?L = ?R$)

proof –

have $?L = \text{measure-pmf.prob } (\text{map-pmf } fst \ (\text{pair-pmf } A \ B)) \ \{\omega. P \ \omega\}$
by (*subst measure-map-pmf*) *simp*
also have $\dots = ?R$
by (*subst map-fst-pair-pmf*) *simp*
finally show *?thesis* **by** *simp*

qed

lemma *pmf-exp-of-fin-function*:

assumes $\text{finite } A \ g \ \text{'set-pmf } p \subseteq A$
shows $(\int \omega. f \ (g \ \omega) \ \partial p) = (\sum y \in A. f \ y * \text{measure } p \ \{\omega. g \ \omega = y\})$
(**is** $?L = ?R$)

proof –

have $?L = \text{integral}^L (\text{map-pmf } g \ p) \ f$
using *integral-map-pmf assms* **by** *simp*
also have $\dots = (\sum a \in A. f \ a * \text{pmf } (\text{map-pmf } g \ p) \ a)$
using *assms*
by (*intro integral-measure-pmf-real*) *auto*
also have $\dots = (\sum y \in A. f \ y * \text{measure } p \ (g \ - \ \{y\}))$
unfolding *assms(1)* **by** (*intro-cong* $[\sigma_2 \ (*)]$ *more:sum.cong pmf-map*)
also have $\dots = ?R$
by (*intro sum.cong*) (*auto simp add: vimage-def*)
finally show *?thesis* **by** *simp*

qed

Cardinality rules for distinct/ordered pairs of a set without the finiteness constraint - to use in simplification:

lemma *card-distinct-pairs*:

$\text{card } \{x \in B \times B. \text{fst } x \neq \text{snd } x\} = \text{card } B^2 - \text{card } B$ (**is** $\text{card } ?L = ?R$)

proof (*cases finite B*)

case *True*

include *intro-cong-syntax*

have $\text{card } ?L = \text{card } (B \times B - (\lambda x. (x,x)) \text{ ` } B)$

by (*intro arg-cong[where f=card]*) *auto*

also have $\dots = \text{card } (B \times B) - \text{card } ((\lambda x. (x,x)) \text{ ` } B)$

by (*intro card-Diff-subset finite-imageI True image-subsetI*) *auto*

also have $\dots = ?R$

using *True* **by** (*intro-cong* [σ_2 (-)] *more: card-image*)

(*auto simp add:power2-eq-square inj-on-def*)

finally show *?thesis* **by** *simp*

next

case *False*

then obtain *p* **where** *p-in*: $p \in B$ **by** *fastforce*

have *False* **if** *finite ?L*

proof -

have $(\lambda x. (p,x)) \text{ ` } (B - \{p\}) \subseteq ?L$

using *p-in* **by** (*intro image-subsetI*) *auto*

hence *finite* $(\lambda x. (p,x)) \text{ ` } (B - \{p\})$

using *finite-subset that* **by** *auto*

hence *finite* $(B - \{p\})$

by (*rule finite-imageD*) (*simp add:inj-on-def*)

hence *finite B*

by *simp*

thus *False* **using** *False* **by** *simp*

qed

hence *infinite ?L* **by** *auto*

hence $\text{card } ?L = 0$ **by** *simp*

also have $\dots = ?R$

using *False* **by** *simp*

finally show *?thesis* **by** *simp*

qed

lemma *card-ordered-pairs'*:

fixes *M* :: (*'a* :: *linorder*) *set*

shows $\text{card } \{(x,y) \in M \times M. x < y\} = \text{card } M * (\text{card } M - 1) / 2$

proof (*cases finite M*)

case *True*

show *?thesis* **using** *card-ordered-pairs[OF True]* **by** *linarith*

next

case *False*

then obtain *p* **where** *p-in*: $p \in M$ **by** *fastforce*

let *?f* = $(\lambda x. \text{if } x < p \text{ then } (x,p) \text{ else } (p,x))$

have *False* **if** *finite* $\{(x,y) \in M \times M. x < y\}$ (**is** *finite ?X*)

proof -

have *?f* $\text{ ` } (M - \{p\}) \subseteq ?X$

using *p-in* **by** (*intro image-subsetI*) *auto*

hence *finite* $(?f \text{ ` } (M - \{p\}))$ **using** *that finite-subset* **by** *auto*

moreover have *inj-on* *?f* $(M - \{p\})$

by (*intro inj-onI*) (*metis Pair-inject*)

ultimately have *finite* $(M - \{p\})$

using *finite-imageD* **by** *blast*

hence *finite M*

using *finite-insert*[**where** $a=p$ **and** $A=M-\{p\}$] **by** *simp*
 thus *False* using *False* **by** *simp*
 qed
 hence *infinite ?X* **by** *auto*
 then show *?thesis* using *False* **by** *simp*
 qed

The following are versions of the mean value theorem, where the interval endpoints may be reversed.

lemma *MVT-symmetric*:

assumes $\bigwedge x. [\min a b \leq x; x \leq \max a b] \implies \text{DERIV } f x :> f' x$
shows $\exists z :: \text{real}. \min a b \leq z \wedge z \leq \max a b \wedge (f b - f a = (b - a) * f' z)$
proof –
 consider (a) $a < b$ | (b) $a = b$ | (c) $a > b$
 by *argo*
 then show *?thesis*
proof (*cases*)
 case a
 then obtain $z :: \text{real}$ **where** $r: a < z < b$ $f b - f a = (b - a) * f' z$
 using *assms MVT2*[**where** $a=a$ **and** $b=b$ **and** $f=f$ **and** $f'=f'$] **by** *auto*
 have $a \leq z < b$ using *r(1,2)* **by** *auto*
 thus *?thesis* using *a r(3)* **by** *auto*
 next
 case b
 then show *?thesis* **by** *auto*
 next
 case c
 then obtain $z :: \text{real}$ **where** $r: b < z < a$ $f a - f b = (a - b) * f' z$
 using *assms MVT2*[**where** $a=b$ **and** $b=a$ **and** $f=f$ **and** $f'=f'$] **by** *auto*
 have $f b - f a = (b - a) * f' z$ using *r* **by** *argo*
 moreover have $b \leq z < a$ using *r(1,2)* **by** *auto*
 ultimately show *?thesis* using *c* **by** *auto*
 qed
 qed

lemma *MVT-interval*:

fixes $I :: \text{real set}$
assumes *interval* I $a \in I$ $b \in I$
assumes $\bigwedge x. x \in I \implies \text{DERIV } f x :> f' x$
shows $\exists z. z \in I \wedge (f b - f a = (b - a) * f' z)$
proof –
 have $a: \min a b \in I$
 using *assms(2,3)* **by** (*cases* $a < b$) *auto*
 have $b: \max a b \in I$
 using *assms(2,3)* **by** (*cases* $a < b$) *auto*
 have $c: x \in \{\min a b.. \max a b\} \implies x \in I$ **for** x
 using *interval-def* *assms(1)* a b **by** *auto*
 have $[\min a b \leq x; x \leq \max a b] \implies \text{DERIV } f x :> f' x$ **for** x
 using *c* *assms(4)* **by** *auto*
 then obtain z **where** $z: z \geq \min a b$ $z \leq \max a b$ $f b - f a = (b - a) * f' z$
 using *MVT-symmetric* **by** *blast*
 have $z \in I$
 using *c* *z(1,2)* **by** *auto*
 thus *?thesis* using *z(3)* **by** *auto*
 qed

\ln is monotone on the positive numbers and thus commutes with \min and \max :

lemma *ln-min-swap*:

$x > (0::real) \implies (y > 0) \implies \ln (\min x y) = \min (\ln x) (\ln y)$
using *ln-less-cancel-iff* **by** *fastforce*

lemma *ln-max-swap*:

$x > (0::real) \implies (y > 0) \implies \ln (\max x y) = \max (\ln x) (\ln y)$
using *ln-le-cancel-iff* **by** *fastforce*

Loose lower bounds for the factorial fuction:.

lemma *fact-lower-bound*:

$\sqrt{2 * \pi * n} * (n / \exp(1)) ^ n \leq \text{fact } n$ (**is** $?L \leq ?R$)

proof (*cases* $n > 0$)

case *True*

have $\ln ?L = \ln (2 * \pi * n) / 2 + n * \ln n - n$

using *True* **by** (*simp add: ln-mult ln-sqrt ln-realpow ln-div algebra-simps*)

also have $\dots \leq \ln ?R$

by (*intro Stirling-Formula.ln-fact-bounds True*)

finally show *?thesis*

using *iffD1[OF ln-le-cancel-iff] True* **by** *simp*

next

case *False*

then show *?thesis* **by** *simp*

qed

lemma *fact-lower-bound-1*:

assumes $n > 0$

shows $(n / \exp 1) ^ n \leq \text{fact } n$ (**is** $?L \leq ?R$)

proof –

have $2 * \pi \geq 1$ **using** *pi-ge-two* **by** *auto*

moreover have $n \geq 1$ **using** *assms* **by** *simp*

ultimately have $2 * \pi * n \geq 1 * 1$

by (*intro mult-mono*) *auto*

hence $a : 2 * \pi * n \geq 1$ **by** *simp*

have $?L = 1 * ?L$ **by** *simp*

also have $\dots \leq \sqrt{2 * \pi * n} * ?L$

using *a* **by** (*intro mult-right-mono*) *auto*

also have $\dots \leq ?R$

using *fact-lower-bound* **by** *simp*

finally show *?thesis* **by** *simp*

qed

Rules to handle O-notation with multiple variables, where some filters may be towards zero:

lemma *real-inv-at-right-0-inf*:

$\forall_F x$ *in at-right* $(0::real)$. $c \leq 1 / x$

proof –

have $c \leq 1 / x$ **if** b : $x \in \{0 < .. < 1 / (\max c 1)\}$ **for** x

proof –

have $c * x \leq (\max c 1) * x$

using b **by** (*intro mult-right-mono, linarith, auto*)

also have $\dots \leq (\max c 1) * (1 / (\max c 1))$

using b **by** (*intro mult-left-mono*) *auto*

also have $\dots \leq 1$

by (*simp add:of-rat-divide*)

finally have $c * x \leq 1$ **by** *simp*

moreover have $0 < x$

using b **by** *simp*

ultimately show *?thesis* **by** (*subst pos-le-divide-eq, auto*)

qed
 thus ?thesis
 by (intro eventually-at-rightI[where b=1/(max c 1)], simp-all)
 qed

lemma bigo-prod-1:
 assumes $(\lambda x. f x) \in O[F](\lambda x. g x) \ G \neq \text{bot}$
 shows $(\lambda x. f (fst x)) \in O[F \times_F G](\lambda x. g (fst x))$
 proof –
 obtain c where a: $\forall_F x \text{ in } F. \text{norm } (f x) \leq c * \text{norm } (g x)$ and c-gt-0: $c > 0$
 using assms unfolding bigo-def by auto

 have $\exists c > 0. \forall_F x \text{ in } F \times_F G. \text{norm } (f (fst x)) \leq c * \text{norm } (g (fst x))$
 by (intro exI[where x=c] conjI c-gt-0 eventually-prod1' a assms(2))
 thus ?thesis
 unfolding bigo-def by simp
 qed

lemma bigo-prod-2:
 assumes $(\lambda x. f x) \in O[G](\lambda x. g x) \ F \neq \text{bot}$
 shows $(\lambda x. f (snd x)) \in O[F \times_F G](\lambda x. g (snd x))$
 proof –
 obtain c where a: $\forall_F x \text{ in } G. \text{norm } (f x) \leq c * \text{norm } (g x)$ and c-gt-0: $c > 0$
 using assms unfolding bigo-def by auto

 have $\exists c > 0. \forall_F x \text{ in } F \times_F G. \text{norm } (f (snd x)) \leq c * \text{norm } (g (snd x))$
 by (intro exI[where x=c] conjI c-gt-0 eventually-prod2' a assms(2))
 thus ?thesis
 unfolding bigo-def by simp
 qed

lemma eventually-inv:
 fixes P :: real \Rightarrow bool
 assumes eventually $(\lambda x. P (1/x))$ at-top
 shows eventually $(\lambda x. P x)$ (at-right 0)
 proof –
 obtain N where c:n $\geq N \implies P (1/n)$ for n
 using assms unfolding eventually-at-top-linorder by auto

define q where $q = \max 1 N$
 have d: $0 < 1 / q \ q > 0$
 unfolding q-def by auto

have P x if $x \in \{0 < .. < 1 / q\}$ for x
 proof –
 define n where $n = 1/x$
 have x-eq: $x = 1 / n$
 unfolding n-def using that by simp

have $N \leq q$ unfolding q-def by simp
 also have $\dots \leq n$
 unfolding n-def using that d by (simp add: divide-simps ac-simps)
 finally have $N \leq n$ by simp
 thus ?thesis
 unfolding x-eq by (intro c)
 qed

thus ?thesis

by (intro eventually-at-rightI[where b=1/q] d)
qed

lemma *bigO-inv*:
fixes $f\ g :: \text{real} \Rightarrow \text{real}$
assumes $(\lambda x. f\ (1/x)) \in O(\lambda x. g\ (1/x))$
shows $f \in O[\text{at-right } 0](g)$
using *assms eventually-inv unfolding bigO-def by auto*

unbundle *no-intro-cong-syntax*

3 Blind

Blind section added to preserve section numbers

end

4 Balls and Bins

The balls and bins model describes the probability space of throwing r balls into b bins. This section derives the expected number of bins hit by at least one ball, as well as the variance in the case that each ball is thrown independently. Further, using an approximation argument it is then possible to derive bounds for the same measures in the case when the balls are being thrown only k -wise independently. The proofs follow the reasoning described in [7, §A.1] but improve on the constants, as well as constraints.

theory *Distributed-Distinct-Elements-Balls-and-Bins*

imports

Distributed-Distinct-Elements-Preliminary

Discrete-Summation.Factorials

HOL-Combinatorics.Stirling

HOL-Computational-Algebra.Polynomial

HOL-Decision-Procs.Approximation

begin

hide-fact *Henstock-Kurzweil-Integration.integral-sum*

hide-fact *Henstock-Kurzweil-Integration.integral-mult-right*

hide-fact *Henstock-Kurzweil-Integration.integral-nonneg*

hide-fact *Henstock-Kurzweil-Integration.integral-cong*

unbundle *intro-cong-syntax*

lemma *sum-power-distrib*:

fixes $f :: 'a \Rightarrow \text{real}$

assumes *finite R*

shows $(\sum_{i \in R} f\ i) \wedge^s = (\sum xs \mid \text{set } xs \subseteq R \wedge \text{length } xs = s. (\prod x \leftarrow xs. f\ x))$

proof (*induction s*)

case 0

have $\{xs. xs = [] \wedge \text{set } xs \subseteq R\} = \{[]\}$

by (*auto simp add:set-eq-iff*)

then show *?case by simp*

next

case (*Suc s*)

have a :

$(\bigcup_{i \in R. (\#) i \in \{xs. \text{set } xs \subseteq R \wedge \text{length } xs = s\}}) = \{xs. \text{set } xs \subseteq R \wedge \text{length } xs = \text{Suc } s\}$

by (*subst lists-length-Suc-eq auto*)

have $\text{sum } f\ R \wedge^{\text{Suc } s} = (\text{sum } f\ R) * (\text{sum } f\ R) \wedge^s$

by *simp*

also have ... = $(\text{sum } f R) * (\sum xs \mid \text{set } xs \subseteq R \wedge \text{length } xs = s. (\prod x \leftarrow xs. f x))$
using *Suc* **by** *simp*
also have ... = $(\sum i \in R. (\sum xs \mid \text{set } xs \subseteq R \wedge \text{length } xs = s. (\prod x \leftarrow i\#xs. f x)))$
by (*subst sum-product*) *simp*
also have ... =
 $(\sum i \in R. (\sum xs \in (\lambda xs. i\#xs) ' \{xs. \text{set } xs \subseteq R \wedge \text{length } xs = s\}. (\prod x \leftarrow xs. f x)))$
by (*subst sum.reindex*) (*auto*)
also have ... = $(\sum xs \in (\bigcup i \in R. (\#) i ' \{xs. \text{set } xs \subseteq R \wedge \text{length } xs = s\}). (\prod x \leftarrow xs. f x))$
by (*intro sum.UNION-disjoint[symmetric]*) *assms ballI finite-imageI finite-lists-length-eq*
auto
also have ... = $(\sum xs \mid \text{set } xs \subseteq R \wedge \text{length } xs = \text{Suc } s. (\prod x \leftarrow xs. f x))$
by (*intro sum.cong a*) *auto*
finally show ?*case* **by** *simp*
qed

lemma *sum-telescope-eq*:

fixes $f :: \text{nat} \Rightarrow 'a :: \{\text{comm-ring-1}\}$
shows $(\sum k \in \{\text{Suc } m..n\}. f k - f (k - 1)) = \text{of-bool}(m \leq n) * (f n - f m)$
by (*cases m ≤ n, subst sum-telescope'', auto*)

An improved version of *diff-power-eq-sum*.

lemma *power-diff-sum*:

fixes $a b :: 'a :: \{\text{comm-ring-1}, \text{power}\}$
shows $a^k - b^k = (a - b) * (\sum i = 0..<k. a^i * b^{(k - 1 - i)})$

proof (*cases k*)

case 0

then show ?*thesis* **by** *simp*

next

case (*Suc nat*)

then show ?*thesis*

unfolding *Suc diff-power-eq-sum*

using *atLeast0LessThan diff-Suc-1* **by** *presburger*

qed

lemma *power-diff-est*:

assumes $(a :: \text{real}) \geq b$

assumes $b \geq 0$

shows $a^k - b^k \leq (a - b) * k * a^{(k-1)}$

proof –

have $a^k - b^k = (a - b) * (\sum i = 0..<k. a^i * b^{(k - 1 - i)})$

by (*rule power-diff-sum*)

also have ... $\leq (a - b) * (\sum i = 0..<k. a^i * a^{(k-1-i)})$

using *assms* **by** (*intro mult-left-mono sum-mono mult-right-mono power-mono, auto*)

also have ... = $(a - b) * (k * a^{(k-1)})$

by (*simp add:power-add[symmetric]*)

finally show ?*thesis* **by** *simp*

qed

lemma *power-diff-est-2*:

assumes $(a :: \text{real}) \geq b$

assumes $b \geq 0$

shows $a^k - b^k \geq (a - b) * k * b^{(k-1)}$

proof –

have $(a - b) * k * b^{(k-1)} = (a - b) * (\sum i = 0..<k. b^i * b^{(k-1-i)})$

by (*simp add:power-add[symmetric]*)

also have ... $\leq (a - b) * (\sum i = 0..<k. a^i * b^{(k-1-i)})$

using *assms*

by (*intro mult-left-mono sum-mono mult-right-mono power-mono*) *auto*

also have ... = $a^{\wedge}k - b^{\wedge}k$
 by (rule power-diff-sum[symmetric])
 finally show ?thesis by simp
 qed

lemma of-bool-prod:
 assumes finite R
 shows $(\prod j \in R. \text{of-bool}(f j)) = (\text{of-bool}(\forall j \in R. f j) :: \text{real})$
 using assms by (induction R rule:finite-induct) auto

Additional results about falling factorials:

lemma ffact-nonneg:
 fixes x :: real
 assumes $k - 1 \leq x$
 shows $\text{ffact } k x \geq 0$
 using assms unfolding prod-ffact[symmetric]
 by (intro prod-nonneg ballI) simp

lemma ffact-pos:
 fixes x :: real
 assumes $k - 1 < x$
 shows $\text{ffact } k x > 0$
 using assms unfolding prod-ffact[symmetric]
 by (intro prod-pos ballI) simp

lemma ffact-mono:
 fixes x y :: real
 assumes $k - 1 \leq x \leq y$
 shows $\text{ffact } k x \leq \text{ffact } k y$
 using assms
 unfolding prod-ffact[symmetric]
 by (intro prod-mono) auto

lemma ffact-of-nat-nonneg:
 fixes x :: 'a :: {comm-ring-1, linordered-nonzero-semiring}
 assumes $x \in \mathbb{N}$
 shows $\text{ffact } k x \geq 0$

proof –
 obtain y where y-def: $x = \text{of-nat } y$
 using assms(1) Nats-cases by auto
 have $(0 :: 'a) \leq \text{of-nat } (\text{ffact } k y)$
 by simp
 also have ... = $\text{ffact } k x$
 by (simp add:of-nat-ffact y-def)
 finally show ?thesis by simp
 qed

lemma ffact-suc-diff:
 fixes x :: ('a :: comm-ring-1)
 shows $\text{ffact } k x - \text{ffact } k (x-1) = \text{of-nat } k * \text{ffact } (k-1) (x-1)$ (is ?L = ?R)

proof (cases k)
 case 0
 then show ?thesis by simp
 next
 case (Suc n)
 hence ?L = $\text{ffact } (\text{Suc } n) x - \text{ffact } (\text{Suc } n) (x-1)$ by simp
 also have ... = $x * \text{ffact } n (x-1) - ((x-1)\text{-of-nat } n) * \text{ffact } n (x-1)$
 by (subst (1) ffact-Suc, simp add: ffact-Suc-rev)

also have ... = *of-nat* (Suc n) * *ffact* n (x-1)
by (*simp add: algebra-simps*)
also have ... = *of-nat* k * *ffact* (k-1) (x-1) **using** *Suc* **by** *simp*
finally show ?thesis **by** *simp*
qed

lemma *ffact-bound*:

ffact k (n::nat) ≤ n ^ k

proof –

have *ffact* k n = (∏ i=0..*k*. (n-i))
unfolding *prod-ffact-nat[symmetric]*

by *simp*

also have ... ≤ (∏ i=0..*k*. n)

by (*intro prod-mono*) *auto*

also have ... = n ^ k

by *simp*

finally show ?thesis **by** *simp*

qed

lemma *fact-moment-binomial*:

fixes n :: nat **and** α :: real

assumes α ∈ {0..1}

defines p ≡ *binomial-pmf* n α

shows (∫ ω. *ffact* s (real ω) ∂p) = *ffact* s (real n) * α ^ s (**is** ?L = ?R)

proof (*cases* s ≤ n)

case *True*

have ?L = (∑ k≤n. (real (n choose k) * α ^ k * (1 - α) ^ (n - k)) * real (*ffact* s k))

unfolding *p-def* **using** *assms* **by** (*subst expectation-binomial-pmf'*) (*auto simp add: of-nat-ffact*)

also have ... = (∑ k ∈ {0+s..(n-s)+s}. (real (n choose k) * α ^ k * (1 - α) ^ (n - k)) * *ffact* s k)

using *True* *ffact-nat-triv* **by** (*intro sum-mono-neutral-cong-right*) *auto*

also have ... = (∑ k=0..n-s. α ^ s * real (n choose (k+s)) * α ^ k * (1-α) ^ (n-(k+s)) * *ffact* s (k+s))

by (*subst sum.atLeastAtMost-shift-bounds*, *simp add: algebra-simps power-add*)

also have ... = α ^ s * (∑ k≤n-s. real (n choose (k+s)) * *ffact* s (k+s) * α ^ k * (1-α) ^ ((n-s)-k))

using *atMost-atLeast0* **by** (*simp add: sum-distrib-left algebra-simps cong: sum.cong*)

also have ... = α ^ s * (∑ k≤n-s. real (n choose (k+s)) * *fact* (k+s) / *fact* k * α ^ k * (1-α) ^ ((n-s)-k))

using *real-of-nat-div[OF fact-dvd[OF le-add1]]*

by (*subst fact-div-fact-ffact-nat[symmetric]*, *auto*)

also have ... = α ^ s * (∑ k≤n-s.

(*fact* n / *fact* (n-s)) * *fact* (n-s) / (*fact* ((n-s)-k) * *fact* k) * α ^ k * (1-α) ^ ((n-s)-k))

using *True* **by** (*intro arg-cong2[where f=(*)]* *sum.cong*)

(*auto simp add: binomial-fact algebra-simps*)

also have ... = α ^ s * (*fact* n / *fact* (n - s)) *

(∑ k≤n-s. *fact* (n-s) / (*fact* ((n-s)-k) * *fact* k) * α ^ k * (1-α) ^ ((n-s)-k))

by (*simp add: sum-distrib-left algebra-simps*)

also have ... = α ^ s * (*fact* n / *fact* (n - s)) * (∑ k≤n-s. ((n-s) choose k) * α ^ k * (1-α) ^ ((n-s)-k))

using *True* **by** (*intro-cong [σ₂(*)]* *more: sum.cong*) (*auto simp add: binomial-fact*)

also have ... = α ^ s * real (*fact* n *div fact* (n - s)) * (α + (1-α)) ^ (n-s)

using *True* *real-of-nat-div[OF fact-dvd]* **by** (*subst binomial-ring*, *simp*)

also have ... = α ^ s * real (*ffact* s n)

by (*subst fact-div-fact-ffact-nat[OF True]*, *simp*)

also have ... = ?R

by (*subst of-nat-ffact*, *simp*)

finally show ?thesis **by** *simp*

next

case *False*

have ?L = (∑ k≤n. (real (n choose k) * α ^ k * (1 - α) ^ (n - k)) * real (*ffact* s k))

unfolding *p-def* **using** *assms* **by** (*subst expectation-binomial-pmf'*) (*auto simp add:of-nat-ffact*)
also have $\dots = (\sum k \leq n. (\text{real } (n \text{ choose } k) * \alpha^k * (1 - \alpha)^{(n - k)}) * \text{real } 0)$
using *False*
by (*intro-cong* [σ_2 (*), σ_1 *of-nat*] *more: sum.cong ffact-nat-triv*) *auto*
also have $\dots = 0$ **by** *simp*
also have $\dots = \text{real } (\text{ffact } s \ n) * \alpha^s$
using *False* **by** (*subst ffact-nat-triv, auto*)
also have $\dots = ?R$
by (*subst of-nat-ffact, simp*)
finally show *?thesis* **by** *simp*
qed

The following describes polynomials of a given maximal degree as a subset of the functions, similar to the subsets \mathbb{Z} or \mathbb{Q} as subsets of larger number classes.

definition *Polynomials* (\mathbb{P})

where *Polynomials* $k = \{f. \exists p. f = \text{poly } p \wedge \text{degree } p \leq k\}$

lemma *Polynomials-mono*:

assumes $s \leq t$

shows $\mathbb{P} \ s \subseteq \mathbb{P} \ t$

using *assms* **unfolding** *Polynomials-def* **by** *auto*

lemma *Polynomials-addI*:

assumes $f \in \mathbb{P} \ k \ g \in \mathbb{P} \ k$

shows $(\lambda \omega. f \ \omega + g \ \omega) \in \mathbb{P} \ k$

proof –

obtain $pf \ pg$ **where** *fg-def*: $f = \text{poly } pf \ \text{degree } pf \leq k \ g = \text{poly } pg \ \text{degree } pg \leq k$

using *assms* **unfolding** *Polynomials-def* **by** *blast*

hence $\text{degree } (pf + pg) \leq k \ (\lambda x. f \ x + g \ x) = \text{poly } (pf + pg)$

using *degree-add-le* **by** *auto*

thus *?thesis* **unfolding** *Polynomials-def* **by** *auto*

qed

lemma *Polynomials-diffI*:

fixes $f \ g :: 'a :: \text{comm-ring} \Rightarrow 'a$

assumes $f \in \mathbb{P} \ k \ g \in \mathbb{P} \ k$

shows $(\lambda x. f \ x - g \ x) \in \mathbb{P} \ k$

proof –

obtain $pf \ pg$ **where** *fg-def*: $f = \text{poly } pf \ \text{degree } pf \leq k \ g = \text{poly } pg \ \text{degree } pg \leq k$

using *assms* **unfolding** *Polynomials-def* **by** *blast*

hence $\text{degree } (pf - pg) \leq k \ (\lambda x. f \ x - g \ x) = \text{poly } (pf - pg)$

using *degree-diff-le* **by** *auto*

thus *?thesis* **unfolding** *Polynomials-def* **by** *auto*

qed

lemma *Polynomials-idI*:

$(\lambda x. x) \in (\mathbb{P} \ 1 :: ('a :: \text{comm-ring-1} \Rightarrow 'a) \ \text{set})$

proof –

have $(\lambda x. x) = \text{poly } [; 0, (1 :: 'a) ;]$

by (*intro ext, auto*)

also have $\dots \in \mathbb{P} \ 1$

unfolding *Polynomials-def* **by** *auto*

finally show *?thesis* **by** *simp*

qed

lemma *Polynomials-constI*:

$(\lambda x. c) \in \mathbb{P} \ k$

proof –

have $(\lambda x. c) = \text{poly } [: c :]$
by *(intro ext, simp)*
also have $\dots \in \mathbb{P} k$
unfolding *Polynomials-def* **by** *auto*
finally show *?thesis* **by** *simp*
qed

lemma *Polynomials-multI*:
fixes $f g :: 'a :: \{\text{comm-ring}\} \Rightarrow 'a$
assumes $f \in \mathbb{P} s \ g \in \mathbb{P} t$
shows $(\lambda x. f x * g x) \in \mathbb{P} (s+t)$

proof –

obtain $pf \ pg$ **where** *xy-def*: $f = \text{poly } pf \ \text{degree } pf \leq s \ g = \text{poly } pg \ \text{degree } pg \leq t$
using *assms* **unfolding** *Polynomials-def* **by** *blast*

have $\text{degree } (pf * pg) \leq \text{degree } pf + \text{degree } pg$
by *(intro degree-mult-le)*
also have $\dots \leq s + t$
using *xy-def* **by** *(intro add-mono) auto*
finally have $\text{degree } (pf * pg) \leq s+t$ **by** *simp*
moreover have $(\lambda x. f x * g x) = \text{poly } (pf * pg)$
using *xy-def* **by** *auto*
ultimately show *?thesis* **unfolding** *Polynomials-def* **by** *auto*

qed

lemma *Polynomials-composeI*:
fixes $f g :: 'a :: \{\text{comm-semiring-0}, \text{semiring-no-zero-divisors}\} \Rightarrow 'a$
assumes $f \in \mathbb{P} s \ g \in \mathbb{P} t$
shows $(\lambda x. f (g x)) \in \mathbb{P} (s*t)$

proof –

obtain $pf \ pg$ **where** *xy-def*: $f = \text{poly } pf \ \text{degree } pf \leq s \ g = \text{poly } pg \ \text{degree } pg \leq t$
using *assms* **unfolding** *Polynomials-def* **by** *blast*

have $\text{degree } (pf \circ_p pg) = \text{degree } pf * \text{degree } pg$
by *(intro degree-pcompose)*

also have $\dots \leq s * t$
using *xy-def* **by** *(intro mult-mono) auto*

finally have $\text{degree } (pf \circ_p pg) \leq s * t$
by *simp*

moreover have $(\lambda x. f (g x)) = \text{poly } (pf \circ_p pg)$
unfolding *xy-def*

by *(intro ext poly-pcompose[symmetric])*

ultimately show *?thesis* **unfolding** *Polynomials-def* **by** *auto*

qed

lemma *Polynomials-const-left-multI*:
fixes $c :: 'a :: \{\text{comm-ring}\}$
assumes $f \in \mathbb{P} k$

shows $(\lambda x. c * f x) \in \mathbb{P} k$

proof –

have $(\lambda x. c * f x) \in \mathbb{P} (0+k)$

by *(intro Polynomials-multI Polynomials-constI assms)*

thus *?thesis* **by** *simp*

qed

lemma *Polynomials-const-right-multI*:
fixes $c :: 'a :: \{\text{comm-ring}\}$
assumes $f \in \mathbb{P} k$

shows $(\lambda x. f x * c) \in \mathbb{P} k$

proof –
have $(\lambda x. f x * c) \in \mathbb{P} (k+0)$
by (*intro Polynomials-multI Polynomials-constI assms*)
thus *?thesis* **by** *simp*
qed

lemma *Polynomials-const-divI*:

fixes $c :: 'a :: \{field\}$
assumes $f \in \mathbb{P} k$
shows $(\lambda x. f x / c) \in \mathbb{P} k$

proof –
have $(\lambda x. f x * (1/c)) \in \mathbb{P} (k+0)$
by (*intro Polynomials-multI Polynomials-constI assms*)
thus *?thesis* **by** *simp*
qed

lemma *Polynomials-ffact*: $(\lambda x. \text{ffact } s (x - y)) \in (\mathbb{P} s :: ('a :: comm-ring-1 \Rightarrow 'a) \text{ set})$

proof (*induction s arbitrary: y*)
case 0
then show *?case*
using *Polynomials-constI* [**where** $c=1$] **by** *simp*
next
case (*Suc s*)
have $(\lambda (x :: 'a). \text{ffact } (Suc s) (x-y)) = (\lambda x. (x-y) * \text{ffact } s (x - (y+1)))$
by (*simp add: ffact-Suc algebra-simps*)
also have $\dots \in \mathbb{P} (1+s)$
by (*intro Polynomials-multI Suc Polynomials-diffI Polynomials-idI Polynomials-constI*)
finally show *?case* **by** *simp*
qed

lemmas *Polynomials-intros* =

Polynomials-const-divI
Polynomials-composeI
Polynomials-const-left-multI
Polynomials-const-right-multI
Polynomials-multI
Polynomials-addI
Polynomials-diffI
Polynomials-idI
Polynomials-constI
Polynomials-ffact

definition $C_2 :: real$ **where** $C_2 = 7.5$

definition $C_3 :: real$ **where** $C_3 = 16$

A locale fixing the sets of balls and bins

locale *balls-and-bins-abs* =
fixes $R :: 'a \text{ set}$ **and** $B :: 'b \text{ set}$
assumes *fin-B*: *finite B* **and** *B-ne*: $B \neq \{\}$
assumes *fin-R*: *finite R*
begin

Independent balls and bins space:

definition Ω
where $\Omega = \text{prod-pmf } R (\lambda \cdot. \text{pmf-of-set } B)$

lemma *set-pmf- Ω* : $\text{set-pmf } \Omega = R \rightarrow_E B$
unfolding *Ω -def set-prod-pmf* [*OF fin-R*]

by (simp add: comp-def set-pmf-of-set[OF B-ne fin-B])

lemma card-B-gt-0: card B > 0
 using B-ne fin-B by auto

lemma card-B-ge-1: card B ≥ 1
 using card-B-gt-0 by simp

definition Z j ω = real (card {i. i ∈ R ∧ ω i = (j::'b)})

definition Y ω = real (card (ω ' R))

definition μ = real (card B) * (1 - (1-1/real (card B))^{card R})

Factorial moments for the random variable describing the number of times a bin will be hit:

lemma fact-moment-balls-and-bins:

assumes J ⊆ B J ≠ {}

shows (∫ ω. ffact s (∑ j ∈ J. Z j ω) ∂Ω) =
 ffact s (real (card R)) * (real (card J) / real (card B))^s
 (is ?L = ?R)

proof -

let ?α = real (card J) / real (card B)

let ?q = binomial-pmf (card R) ?α

let ?Y = (λω. card {r ∈ R. ω r ∈ J})

have fin-J: finite J

using finite-subset assms(1) fin-B by auto

have Z-sum-eq: (∑ j ∈ J. Z j ω) = real (?Y ω) for ω

proof -

have ?Y ω = card (∪ j ∈ J. {r ∈ R. ω r = j})

by (intro arg-cong[where f=card]) auto

also have ... = (∑ i ∈ J. card {r ∈ R. ω r = i})

using fin-R fin-J by (intro card-UN-disjoint) auto

finally have ?Y ω = (∑ j ∈ J. card {r ∈ R. ω r = j}) by simp

thus ?thesis

unfolding Z-def of-nat-sum[symmetric] by simp

qed

have card-J: card J ≤ card B

using assms(1) fin-B card-mono by auto

have α-range: ?α ≥ 0 ?α ≤ 1

using card-J card-B-gt-0 by auto

have pmf (map-pmf (λω. ω ∈ J) (pmf-of-set B)) x = pmf (bernoulli-pmf ?α) x
 (is ?L1=?R1) for x

proof -

have ?L1 = real (card (B ∩ {ω. (ω ∈ J) = x})) / real (card B)

using B-ne fin-B

by (simp add: pmf-map measure-pmf-of-set vimage-def)

also have ... = (if x then (card J) else (card (B - J))) / real (card B)

using Int-absorb1[OF assms(1)] by (auto simp add: Diff-eq Int-def)

also have ... = (if x then (card J) / card B else (real (card B) - card J) / real (card B))

using card-J fin-J assms(1) by (simp add: of-nat-diff card-Diff-subset)

also have ... = (if x then ?α else (1 - ?α))

using card-B-gt-0 by (simp add: divide-simps)

also have ... = ?R1

using α-range by auto

finally show ?thesis by simp

qed
hence $c:\text{map-pmf } (\lambda\omega. \omega \in J) (\text{pmf-of-set } B) = \text{bernoulli-pmf } ?\alpha$
by $(\text{intro pmf-eqI}) \text{ simp}$

have $\text{map-pmf } (\lambda\omega. \lambda r \in R. \omega r \in J) \Omega = \text{prod-pmf } R (\lambda-. (\text{map-pmf } (\lambda\omega. \omega \in J) (\text{pmf-of-set } B)))$
unfolding $\text{map-pmf-def } \Omega\text{-def } \text{restrict-def}$ **using** fin-R
by $(\text{subst Pi-pmf-bind}[\text{where } d'=\text{undefined}]) \text{ auto}$
also have $\dots = \text{prod-pmf } R (\lambda-. \text{bernoulli-pmf } ?\alpha)$
unfolding c **by** simp
finally have $b:\text{map-pmf } (\lambda\omega. \lambda r \in R. \omega r \in J) \Omega = \text{prod-pmf } R (\lambda-. \text{bernoulli-pmf } ?\alpha)$
by simp

have $\text{map-pmf } ?Y \Omega = \text{map-pmf } ((\lambda\omega. \text{card } \{r \in R. \omega r\}) \circ (\lambda\omega. \lambda r \in R. \omega r \in J)) \Omega$
unfolding comp-def
by $(\text{intro map-pmf-cong } \text{arg-cong}[\text{where } f=\text{card}]) (\text{auto } \text{simp } \text{add:comp-def})$
also have $\dots = (\text{map-pmf } (\lambda\omega. \text{card } \{r \in R. \omega r\}) \circ \text{map-pmf } (\lambda\omega. \lambda r \in R. \omega r \in J)) \Omega$
by $(\text{subst map-pmf-compose}[\text{symmetric}]) \text{ auto}$
also have $\dots = \text{map-pmf } (\lambda\omega. \text{card } \{r \in R. \omega r\}) (\text{prod-pmf } R (\lambda-. (\text{bernoulli-pmf } ?\alpha)))$
unfolding $\text{comp-def } b$ **by** simp
also have $\dots = ?q$
using $\alpha\text{-range}$ **by** $(\text{intro binomial-pmf-altdef}'[\text{symmetric}] \text{fin-R}) \text{ auto}$
finally have $a:\text{map-pmf } ?Y \Omega = ?q$
by simp

have $?L = (\int \omega. \text{ffact } s (\text{real } (?Y \omega)) \partial\Omega)$
unfolding $Z\text{-sum-eq}$ **by** simp
also have $\dots = (\int \omega. \text{ffact } s (\text{real } \omega) \partial(\text{map-pmf } ?Y \Omega))$
by simp
also have $\dots = (\int \omega. \text{ffact } s (\text{real } \omega) \partial?q)$
unfolding a **by** simp
also have $\dots = ?R$
using $\alpha\text{-range}$ **by** $(\text{subst fact-moment-binomial}, \text{auto})$
finally show $?thesis$ **by** simp

qed

Expectation and variance for the number of distinct bins that are hit by at least one ball in the fully independent model. The result for the variance is improved by a factor of 4 w.r.t. the paper.

lemma

shows $\text{exp-balls-and-bins: measure-pmf.expectation } \Omega Y = \mu (\text{is } ?AL = ?AR)$
and $\text{var-balls-and-bins: measure-pmf.variance } \Omega Y \leq \text{card } R * (\text{real } (\text{card } R) - 1) / \text{card } B$
(is $?BL \leq ?BR)$

proof –

let $?b = \text{real } (\text{card } B)$
let $?r = \text{card } R$
define $Z :: 'b \Rightarrow ('a \Rightarrow 'b) \Rightarrow \text{real}$
where $Z = (\lambda i \omega. \text{of-bool}(i \notin \omega \text{ ' } R))$
define α **where** $\alpha = (1 - 1 / ?b) \wedge ?r$
define β **where** $\beta = (1 - 2 / ?b) \wedge ?r$

have $\text{card } (B \times B \cap \{x. \text{fst } x = \text{snd } x\}) = \text{card } ((\lambda x. (x,x)) \text{ ' } B)$
by $(\text{intro } \text{arg-cong}[\text{where } f=\text{card}]) \text{ auto}$
also have $\dots = \text{card } B$
by $(\text{intro } \text{card-image}, \text{simp } \text{add:inj-on-def})$
finally have $d: \text{card } (B \times B \cap \{x. \text{fst } x = \text{snd } x\}) = \text{card } B$
by simp
hence $\text{count-1: real } (\text{card } (B \times B \cap \{x. \text{fst } x = \text{snd } x\})) = \text{card } B$

by *simp*

have $\text{card } B + \text{card } (B \times B \cap -\{x. \text{fst } x = \text{snd } x\}) =$
 $\text{card } (B \times B \cap \{x. \text{fst } x = \text{snd } x\}) + \text{card } (B \times B \cap -\{x. \text{fst } x = \text{snd } x\})$
by (*subst d*) *simp*

also have $\dots = \text{card } ((B \times B \cap \{x. \text{fst } x = \text{snd } x\}) \cup (B \times B \cap -\{x. \text{fst } x = \text{snd } x\}))$
using *finite-subset[OF - finite-cartesian-product[OF fin-B fin-B]]*
by (*intro card-Un-disjoint[symmetric]*) *auto*

also have $\dots = \text{card } (B \times B)$
by (*intro arg-cong[where f=card]*) *auto*

also have $\dots = \text{card } B^{\wedge 2}$
unfolding *card-cartesian-product* **by** (*simp add:power2-eq-square*)

finally have $\text{card } B + \text{card } (B \times B \cap -\{x. \text{fst } x = \text{snd } x\}) = \text{card } B^{\wedge 2}$ **by** *simp*

hence *count-2*: $\text{real } (\text{card } (B \times B \cap -\{x. \text{fst } x = \text{snd } x\})) = \text{real } (\text{card } B)^{\wedge 2} - \text{card } B$
by (*simp add:algebra-simps flip: of-nat-add of-nat-power*)

hence *finite* (*set-pmf* Ω)
unfolding *set-pmf- Ω*
using *fin-R fin-B* **by** (*auto intro!:finite-PiE*)

hence *int*: *integrable* (*measure-pmf* Ω) *f*
for $f :: ('a \Rightarrow 'b) \Rightarrow \text{real}$
by (*intro integrable-measure-pmf-finite*) *simp*

have *a*: *prob-space.indep-vars* (*measure-pmf* Ω) (*λi . discrete*) ($\lambda x \omega. \omega x$) *R*
unfolding Ω -*def* **using** *indep-vars-Pi-pmf[OF fin-R]* **by** *metis*

have *b*: $(\int \omega. \text{of-bool } (\omega ' R \subseteq A) \partial\Omega) = (\text{real } (\text{card } (B \cap A)) / \text{real } (\text{card } B))^{\wedge \text{card } R}$
(is ?L = ?R) **for** *A*

proof –

have $?L = (\int \omega. (\prod j \in R. \text{of-bool}(\omega j \in A)) \partial\Omega)$
by (*intro Bochner-Integration.integral-cong ext*)
(auto simp add: of-bool-prod[OF fin-R])

also have $\dots = (\prod j \in R. (\int \omega. \text{of-bool}(\omega j \in A) \partial\Omega))$
using *fin-R*
by (*intro prob-space.indep-vars-lebesgue-integral[OF prob-space-measure-pmf] int*
prob-space.indep-vars-compose2[OF prob-space-measure-pmf a]) *auto*

also have $\dots = (\prod j \in R. (\int \omega. \text{of-bool}(\omega \in A) \partial(\text{map-pmf } (\lambda \omega. \omega j) \Omega)))$
by *simp*

also have $\dots = (\prod j \in R. (\int \omega. \text{of-bool}(\omega \in A) \partial(\text{pmf-of-set } B)))$
unfolding Ω -*def* **by** (*subst Pi-pmf-component[OF fin-R]*) *simp*

also have $\dots = ((\sum \omega \in B. \text{of-bool } (\omega \in A)) / \text{real } (\text{card } B))^{\wedge \text{card } R}$
by (*simp add: integral-pmf-of-set[OF B-ne fin-B]*)

also have $\dots = ?R$
unfolding *of-bool-def sum.If-cases[OF fin-B]* **by** *simp*

finally show *?thesis* **by** *simp*

qed

have *Z-exp*: $(\int \omega. Z i \omega \partial\Omega) = \alpha$ **if** $i \in B$ **for** *i*

proof –

have $\text{real } (\text{card } (B \cap -\{i\})) = \text{real } (\text{card } (B - \{i\}))$
by (*intro-cong [σ_1 card, σ_1 of-nat]*) *auto*

also have $\dots = \text{real } (\text{card } B - \text{card } \{i\})$
using *that* **by** (*subst card-Diff-subset*) *auto*

also have $\dots = \text{real } (\text{card } B) - \text{real } (\text{card } \{i\})$
using *fin-B that* **by** (*intro of-nat-diff card-mono*) *auto*

finally have *c*: $\text{real } (\text{card } (B \cap -\{i\})) = \text{real } (\text{card } B) - 1$
by *simp*

have $(\int \omega. Z i \omega \partial \Omega) = (\int \omega. \text{of-bool}(\omega \text{ ' } R \subseteq - \{i\}) \partial \Omega)$
unfolding $Z\text{-def}$ **by** simp
also have $\dots = (\text{real}(\text{card}(B \cap -\{i\})) / \text{real}(\text{card } B))^{\wedge \text{card } R}$
by $(\text{intro } b)$
also have $\dots = ((\text{real}(\text{card } B) - 1) / \text{real}(\text{card } B))^{\wedge \text{card } R}$
by $(\text{subst } c)$ simp
also have $\dots = \alpha$
unfolding $\alpha\text{-def}$ **using** $\text{card-}B\text{-gt-0}$
by $(\text{simp add:divide-eq-eq diff-divide-distrib})$
finally show $?thesis$
by simp
qed

have $Z\text{-prod-exp: } (\int \omega. Z i \omega * Z j \omega \partial \Omega) = (\text{if } i = j \text{ then } \alpha \text{ else } \beta)$
if $i \in B \ j \in B$ **for** $i \ j$
proof –

have $\text{real}(\text{card}(B \cap -\{i,j\})) = \text{real}(\text{card}(B - \{i,j\}))$
by $(\text{intro-cong } [\sigma_1 \text{ card}, \sigma_1 \text{ of-nat}]) \text{ auto}$
also have $\dots = \text{real}(\text{card } B - \text{card } \{i,j\})$
using that **by** $(\text{subst card-Diff-subset}) \text{ auto}$
also have $\dots = \text{real}(\text{card } B) - \text{real}(\text{card } \{i,j\})$
using $\text{fin-}B$ **that** **by** $(\text{intro of-nat-diff card-mono}) \text{ auto}$
finally have $c: \text{real}(\text{card}(B \cap -\{i,j\})) = \text{real}(\text{card } B) - \text{card } \{i,j\}$
by simp

have $(\int \omega. Z i \omega * Z j \omega \partial \Omega) = (\int \omega. \text{of-bool}(\omega \text{ ' } R \subseteq - \{i,j\}) \partial \Omega)$
unfolding $Z\text{-def of-bool-conj[symmetric]}$
by $(\text{intro integral-cong ext}) \text{ auto}$
also have $\dots = (\text{real}(\text{card}(B \cap -\{i,j\})) / \text{real}(\text{card } B))^{\wedge \text{card } R}$
by $(\text{intro } b)$
also have $\dots = ((\text{real}(\text{card } B) - \text{card } \{i,j\}) / \text{real}(\text{card } B))^{\wedge \text{card } R}$
by $(\text{subst } c)$ simp
also have $\dots = (\text{if } i = j \text{ then } \alpha \text{ else } \beta)$
unfolding $\alpha\text{-def } \beta\text{-def}$ **using** $\text{card-}B\text{-gt-0}$
by $(\text{simp add:divide-eq-eq diff-divide-distrib})$
finally show $?thesis$ **by** simp
qed

have $Y\text{-eq: } Y \omega = (\sum i \in B. 1 - Z i \omega)$ **if** $\omega \in \text{set-pmf } \Omega$ **for** ω
proof –

have $\text{set-pmf } \Omega \subseteq \text{Pi } R (\lambda-. B)$
using $\text{set-pmf-}\Omega$ **by** $(\text{simp add:PiE-def})$
hence $\omega \text{ ' } R \subseteq B$
using that **by** auto
hence $Y \omega = \text{card}(B \cap \omega \text{ ' } R)$
unfolding $Y\text{-def}$ **using** Int-absorb1 **by** metis
also have $\dots = (\sum i \in B. \text{of-bool}(i \in \omega \text{ ' } R))$
unfolding $\text{of-bool-def sum.If-cases[OF fin-}B]$ **by** (simp)
also have $\dots = (\sum i \in B. 1 - Z i \omega)$
unfolding $Z\text{-def}$ **by** (intro sum.cong) $(\text{auto simp add:of-bool-def})$
finally show $Y \omega = (\sum i \in B. 1 - Z i \omega)$ **by** simp
qed

have $Y\text{-sq-eq: } (Y \omega)^2 = (\sum (i,j) \in B \times B. 1 - Z i \omega - Z j \omega + Z i \omega * Z j \omega)$
if $\omega \in \text{set-pmf } \Omega$ **for** ω
unfolding $Y\text{-eq[OF that]}$ $\text{power2-eq-square sum-product sum.cartesian-product}$
by (intro sum.cong) $(\text{auto simp add:algebra-simps})$

have *measure-pmf.expectation* $\Omega Y = (\int \omega. (\sum i \in B. 1 - Z i \omega) \partial\Omega)$
using *Y-eq* **by** (*intro integral-cong-AE AE-pmfI*) *auto*
also have $\dots = (\sum i \in B. 1 - (\int \omega. Z i \omega \partial\Omega))$
using *int* **by** *simp*
also have $\dots = ?b * (1 - \alpha)$
using *Z-exp* **by** *simp*
also have $\dots = ?AR$
unfolding *α -def* *μ -def* **by** *simp*
finally show $?AL = ?AR$ **by** *simp*

have *measure-pmf.variance* $\Omega Y = (\int \omega. Y \omega^2 \partial\Omega) - (\int \omega. Y \omega \partial\Omega)^2$
using *int* **by** (*subst measure-pmf.variance-eq*) *auto*
also have $\dots =$
 $(\int \omega. (\sum i \in B \times B. 1 - Z (fst i) \omega - Z (snd i) \omega + Z (fst i) \omega * Z (snd i) \omega) \partial\Omega) -$
 $(\int \omega. (\sum i \in B. 1 - Z i \omega) \partial\Omega)^2$
using *Y-eq* *Y-sq-eq*
by (*intro-cong* [$\sigma_2(-)$, σ_2 *power*] *more: integral-cong-AE AE-pmfI*) (*auto simp add:case-prod-beta*)
also have $\dots =$
 $(\sum i \in B \times B. (\int \omega. (1 - Z (fst i) \omega - Z (snd i) \omega + Z (fst i) \omega * Z (snd i) \omega) \partial\Omega)) -$
 $(\sum i \in B. (\int \omega. (1 - Z i \omega) \partial\Omega))^2$
by (*intro-cong* [$\sigma_2(-)$, σ_2 *power*] *more: integral-sum int*)
also have $\dots =$
 $(\sum i \in B \times B. (\int \omega. (1 - Z (fst i) \omega - Z (snd i) \omega + Z (fst i) \omega * Z (snd i) \omega) \partial\Omega)) -$
 $(\sum i \in B \times B. (\int \omega. (1 - Z (fst i) \omega) \partial\Omega) * (\int \omega. (1 - Z (snd i) \omega) \partial\Omega))$
unfolding *power2-eq-square* *sum-product* *sum.cartesian-product*
by (*simp add:case-prod-beta*)
also have $\dots = (\sum (i,j) \in B \times B. (\int \omega. (1 - Z i \omega - Z j \omega + Z i \omega * Z j \omega) \partial\Omega) -$
 $(\int \omega. (1 - Z i \omega) \partial\Omega) * (\int \omega. (1 - Z j \omega) \partial\Omega))$
by (*subst sum-subtractf[symmetric]*, *simp add:case-prod-beta*)
also have $\dots = (\sum (i,j) \in B \times B. (\int \omega. Z i \omega * Z j \omega \partial\Omega) - (\int \omega. Z i \omega \partial\Omega) * (\int \omega. Z j \omega$
 $\partial\Omega))$
using *int* **by** (*intro sum.cong refl*) (*simp add:algebra-simps case-prod-beta*)
also have $\dots = (\sum i \in B \times B. (if\ fst\ i =\ snd\ i\ then\ \alpha - \alpha^2\ else\ \beta - \alpha^2))$
by (*intro sum.cong refl*)
(simp add:Z-exp Z-prod-exp mem-Times-iff case-prod-beta power2-eq-square)
also have $\dots = ?b * (\alpha - \alpha^2) + (?b^2 - card\ B) * (\beta - \alpha^2)$
using *count-1* *count-2* *finite-cartesian-product* *fin-B* **by** (*subst sum.If-cases*) *auto*
also have $\dots = ?b^2 * (\beta - \alpha^2) + ?b * (\alpha - \beta)$
by (*simp add:algebra-simps*)
also have $\dots = ?b * ((1 - 1 / ?b)^{?r} - (1 - 2 / ?b)^{?r}) - ?b^2 * (((1 - 1 / ?b)^2)^{?r} - (1 - 2 / ?b)^{?r})$
unfolding *β -def* *α -def*
by (*simp add: power-mult[symmetric] algebra-simps*)
also have $\dots \leq card\ R * (real\ (card\ R) - 1) / card\ B$ (**is** $?L \leq ?R$)
proof (*cases ?b ≥ 2*)
case *True*
have $?L \leq$
 $?b * (((1 - 1 / ?b) - (1 - 2 / ?b)) * ?r * (1 - 1 / ?b)^{?r - 1}) -$
 $?b^2 * (((1 - 1 / ?b)^2) - ((1 - 2 / ?b))) * ?r * ((1 - 2 / ?b)^{?r - 1})$
using *True*
by (*intro diff-mono mult-left-mono power-diff-est-2 power-diff-est divide-right-mono*)
(auto simp add:power2-eq-square algebra-simps)
also have $\dots = ?b * ((1 / ?b) * ?r * (1 - 1 / ?b)^{?r - 1}) - ?b^2 * ((1 / ?b^2)^{?r} * ((1 - 2 / ?b)^{?r - 1}))$
by (*intro arg-cong2[where f=(-)] arg-cong2[where f=(*)] refl*)
(auto simp add:algebra-simps power2-eq-square)
also have $\dots = ?r * ((1 - 1 / ?b)^{?r - 1} - ((1 - 2 / ?b)^{?r - 1}))$
by (*simp add:algebra-simps*)
also have $\dots \leq ?r * (((1 - 1 / ?b) - (1 - 2 / ?b)) * (?r - 1) * (1 - 1 / ?b)^{?r - 1 - 1})$
using *True* **by** (*intro mult-left-mono power-diff-est*) *(auto simp add:algebra-simps)*

also have $\dots \leq ?r * ((1/?b) * (?r - 1) * 1^{(?r - 1 - 1)})$
using *True* **by** (*intro mult-left-mono mult-mono power-mono*) *auto*
also have $\dots = ?R$
using *card-B-gt-0* **by** *auto*
finally show $?L \leq ?R$ **by** *simp*
next
case *False*
hence $?b = 1$ **using** *card-B-ge-1* **by** *simp*
thus $?L \leq ?R$
by (*cases card R = 0*) *auto*
qed
finally show *measure-pmf.variance* $\Omega Y \leq \text{card } R * (\text{real } (\text{card } R) - 1) / \text{card } B$
by *simp*
qed

definition *lim-balls-and-bins* $k p = ($
prob-space.k-wise-indep-vars (measure-pmf p) k (*λ* . *discrete*) (*$\lambda x \omega. \omega x$*) *R* \wedge
 $(\forall x. x \in R \longrightarrow \text{map-pmf } (\lambda \omega. \omega x) p = \text{pmf-of-set } B))$

lemma *indep*:

assumes *lim-balls-and-bins* $k p$
shows *prob-space.k-wise-indep-vars (measure-pmf p) k* (*λ* . *discrete*) (*$\lambda x \omega. \omega x$*) *R* \wedge
using *assms lim-balls-and-bins-def* **by** *simp*

lemma *ran*:

assumes *lim-balls-and-bins* $k p x \in R$
shows *map-pmf* (*$\lambda \omega. \omega x$*) $p = \text{pmf-of-set } B$
using *assms lim-balls-and-bins-def* **by** *simp*

lemma *Z-integrable*:

fixes $f :: \text{real} \Rightarrow \text{real}$
assumes *lim-balls-and-bins* $k p$
shows *integrable* $p (\lambda \omega. f (Z i \omega))$
unfolding *Z-def* **using** *fin-R card-mono*
by (*intro integrable-pmf-iff-bounded* [**where** $C = \text{Max } (\text{abs } 'f' \text{ real } \{..\text{card } R\})$])
fastforce

lemma *Z-any-integrable-2*:

fixes $f :: \text{real} \Rightarrow \text{real}$
assumes *lim-balls-and-bins* $k p$
shows *integrable* $p (\lambda \omega. f (Z i \omega + Z j \omega))$

proof –

have $q: \text{real } (\text{card } A) + \text{real } (\text{card } B) \in \text{real } \{..2 * \text{card } R\}$ **if** $A \subseteq R B \subseteq R$ **for** $A B$

proof –

have $\text{card } A + \text{card } B \leq \text{card } R + \text{card } R$
by (*intro add-mono card-mono fin-R that*)
also have $\dots = 2 * \text{card } R$ **by** *simp*

finally show *?thesis* **by** *force*

qed

thus *?thesis*

unfolding *Z-def* **using** *fin-R card-mono abs-triangle-ineq*
by (*intro integrable-pmf-iff-bounded* [**where** $C = \text{Max } (\text{abs } 'f' \text{ real } \{..2 * \text{card } R\})$]) *Max-ge*
finite-imageI imageI *auto*

qed

lemma *hit-count-prod-exp*:

assumes $j1 \in B j2 \in B s+t \leq k$

assumes *lim-balls-and-bins* $k p$
defines $L \equiv \{(xs,ys). \text{ set } xs \subseteq R \wedge \text{ set } ys \subseteq R \wedge$
 $(\text{ set } xs \cap \text{ set } ys = \{\}) \vee j1 = j2) \wedge \text{ length } xs = s \wedge \text{ length } ys = t\}$
shows $(\int \omega. Z j1 \omega \widehat{s} * Z j2 \omega \widehat{t} \partial p) =$
 $(\sum (xs,ys) \in L. (1/\text{real } (\text{card } B)) \widehat{\wedge} (\text{card } (\text{set } xs \cup \text{set } ys)))$
 $(\text{is } ?L = ?R)$
proof –
define $W1 :: 'a \Rightarrow ('a \Rightarrow 'b) \Rightarrow \text{real}$
where $W1 = (\lambda i \omega. \text{ of-bool } (\omega i = j1) :: \text{real})$
define $W2 :: 'a \Rightarrow ('a \Rightarrow 'b) \Rightarrow \text{real}$
where $W2 = (\lambda i \omega. \text{ of-bool } (\omega i = j2) :: \text{real})$
define $\tau :: 'a \text{ list} \times 'a \text{ list} \Rightarrow 'a \Rightarrow 'b$
where $\tau = (\lambda l x. \text{ if } x \in \text{ set } (\text{fst } l) \text{ then } j1 \text{ else } j2)$

have $\tau\text{-check-1: } \tau l x = j1 \text{ if } x \in \text{ set } (\text{fst } l) \text{ and } l \in L \text{ for } x l$
using that unfolding $\tau\text{-def } L\text{-def}$ **by** *auto*
have $\tau\text{-check-2: } \tau l x = j2 \text{ if } x \in \text{ set } (\text{snd } l) \text{ and } l \in L \text{ for } x l$
using that unfolding $\tau\text{-def } L\text{-def}$ **by** *auto*
have $\tau\text{-check-3: } \tau l x \in B \text{ for } x l$
using *assms(1,2)* **unfolding** $\tau\text{-def}$ **by** *simp*

have $Z1\text{-eq: } Z j1 \omega = (\sum i \in R. W1 i \omega) \text{ for } \omega$
using *fin-R* **unfolding** $Z\text{-def } W1\text{-def}$
by *(simp add:of-bool-def sum.If-cases Int-def)*

have $Z2\text{-eq: } Z j2 \omega = (\sum i \in R. W2 i \omega) \text{ for } \omega$
using *fin-R* **unfolding** $Z\text{-def } W2\text{-def}$
by *(simp add:of-bool-def sum.If-cases Int-def)*

define α **where** $\alpha = 1 / \text{real } (\text{card } B)$

have $a: (\int \omega. (\prod x \leftarrow a. W1 x \omega) * (\prod y \leftarrow b. W2 y \omega) \partial p) = 0 \text{ (is } ?L1 = 0)$
if $x \in \text{ set } a \cap \text{ set } b \text{ } j1 \neq j2 \text{ length } a = s \text{ length } b = t \text{ for } x a b$
proof –
have $(\prod x \leftarrow a. W1 x \omega) * (\prod y \leftarrow b. W2 y \omega) = 0 \text{ for } \omega$
proof –
have $W1 x \omega = 0 \vee W2 y \omega = 0$
unfolding $W1\text{-def } W2\text{-def}$ **using that by** *simp*
hence $(\prod x \leftarrow a. W1 x \omega) = 0 \vee (\prod y \leftarrow b. W2 y \omega) = 0$
unfolding *prod-list-zero-iff* **using that(1)** **by** *auto*
thus *?thesis* **by** *simp*
qed
hence $?L1 = (\int \omega. 0 \partial p)$
by *(intro arg-cong2[where f=measure-pmf.expectation]) auto*
also have $\dots = 0$
by *simp*
finally show *?thesis* **by** *simp*
qed

have $b: \text{prob-space.indep-vars } p (\lambda-. \text{discrete}) (\lambda i \omega. \omega i) (\text{set } (\text{fst } x) \cup \text{set } (\text{snd } x))$
if $x \in L \text{ for } x$
proof –
have $\text{card } (\text{set } (\text{fst } x) \cup \text{set } (\text{snd } x)) \leq \text{card } (\text{set } (\text{fst } x)) + \text{card } (\text{set } (\text{snd } x))$
by *(intro card-Un-le)*
also have $\dots \leq \text{length } (\text{fst } x) + \text{length } (\text{snd } x)$
by *(intro add-mono card-length)*
also have $\dots = s + t$
using that $L\text{-def}$ **by** *auto*

also have $\dots \leq k$ **using** $assms(3)$ **by** $simp$
finally have $card (set (fst x) \cup set (snd x)) \leq k$ **by** $simp$
moreover have $set (fst x) \cup set (snd x) \subseteq R$
using $that L-def$ **by** $auto$
ultimately show $?thesis$
by ($intro prob-space.k-wise-indep-vars-subset[OF prob-space-measure-pmf indep[OF assms(4)]]$)
 $auto$
qed

have $c: (\int \omega. of_bool (\omega x = z) \partial p) = \alpha$ (**is** $?L1 = -$)
if $z \in B$ $x \in R$ **for** $x z$
proof –
have $?L1 = (\int \omega. indicator \{\omega. \omega x = z\} \omega \partial p)$
unfolding $indicator-def$ **by** $simp$
also have $\dots = measure p \{\omega. \omega x = z\}$
by $simp$
also have $\dots = measure (map_pmf (\lambda \omega. \omega x) p) \{z\}$
by ($subst measure-map-pmf$) ($simp add:vimage-def$)
also have $\dots = measure (pmf-of-set B) \{z\}$
using $that$ **by** ($subst ran[OF assms(4)]$) $auto$
also have $\dots = 1 / card B$
using $fin-B$ **that** **by** ($subst measure-pmf-of-set$) $auto$
also have $\dots = \alpha$
unfolding $\alpha-def$ **by** $simp$
finally show $?thesis$ **by** $simp$
qed

have $d: abs x \leq 1 \implies abs y \leq 1 \implies abs (x*y) \leq 1$ **for** $x y :: real$
by ($simp add:abs-mult mult-le-one$)
have $e: (\bigwedge x. x \in set xs \implies abs x \leq 1) \implies abs(prod-list xs) \leq 1$ **for** $xs :: real list$
using d **by** ($induction xs, simp, simp$)

have $?L = (\int \omega. (\sum j \in R. W1 j \omega) \wedge^s * (\sum j \in R. W2 j \omega) \wedge^t \partial p)$
unfolding $Z1-eq Z2-eq$ **by** $simp$
also have $\dots = (\int \omega. (\sum xs \mid set xs \subseteq R \wedge length xs = s. (\prod x \leftarrow xs. W1 x \omega)) * (\sum ys \mid set ys \subseteq R \wedge length ys = t. (\prod y \leftarrow ys. W2 y \omega)) \partial p)$
unfolding $sum-power-distrib[OF fin-R]$ **by** $simp$
also have $\dots = (\int \omega. (\sum l \in \{xs. set xs \subseteq R \wedge length xs = s\} \times \{ys. set ys \subseteq R \wedge length ys = t\}. (\prod x \leftarrow fst l. W1 x \omega) * (\prod y \leftarrow snd l. W2 y \omega)) \partial p)$
by ($intro arg-cong[where f=integral^L p]$)
 $(simp add: sum-product sum.cartesian-product case-prod-beta)$
also have $\dots = (\sum l \in \{xs. set xs \subseteq R \wedge length xs = s\} \times \{ys. set ys \subseteq R \wedge length ys = t\}. (\int \omega. (\prod x \leftarrow fst l. W1 x \omega) * (\prod y \leftarrow snd l. W2 y \omega) \partial p))$
unfolding $W1-def W2-def$
by ($intro integral-sum integrable-pmf-iff-bounded[where C=1] d e$) $auto$
also have $\dots = (\sum l \in L. (\int \omega. (\prod x \leftarrow fst l. W1 x \omega) * (\prod y \leftarrow snd l. W2 y \omega) \partial p))$
unfolding $L-def$ **using** a **by** ($intro sum.mono-neutral-right finite-cartesian-product finite-lists-length-eq fin-R$) $auto$
also have $\dots = (\sum l \in L. (\int \omega. (\prod x \leftarrow fst l. of_bool(\omega x = \tau l x)) * (\prod y \leftarrow snd l. of_bool(\omega y = \tau l y)) \partial p))$
unfolding $W1-def W2-def$ **using** $\tau-check-1 \tau-check-2$
by ($intro sum.cong arg-cong[where f=integral^L p] ext arg-cong2[where f=(*)] arg-cong[where f=prod-list]$) $auto$
also have $\dots = (\sum l \in L. (\int \omega. (\prod x \leftarrow (fst l @ snd l). of_bool(\omega x = \tau l x)) \partial p))$
by $simp$
also have $\dots = (\sum l \in L. (\int \omega. (\prod x \in set (fst l @ snd l). of_bool(\omega x = \tau l x) \wedge count-list (fst l @ snd l) x) \partial p))$

unfolding *prod-list-eval* **by** *simp*
also have ... = $(\sum l \in L. (\int \omega. (\prod x \in \text{set } (fst\ l) \cup \text{set } (snd\ l). \text{of-bool}(\omega\ x = \tau\ l\ x) \wedge \text{count-list } (fst\ l @ snd\ l)\ x) \partial\ p))$
by *simp*
also have ... = $(\sum l \in L. (\int \omega. (\prod x \in \text{set } (fst\ l) \cup \text{set } (snd\ l). \text{of-bool}(\omega\ x = \tau\ l\ x)) \partial\ p))$
using *count-list-gr-1*
by (*intro sum.cong arg-cong*[**where** $f = \text{integral}^L\ p$] *ext prod.cong*) *force+*
also have ... = $(\sum l \in L. (\prod x \in \text{set } (fst\ l) \cup \text{set } (snd\ l). (\int \omega. \text{of-bool}(\omega\ x = \tau\ l\ x) \partial\ p)))$
by (*intro sum.cong prob-space.indep-vars-lebesgue-integral*[*OF prob-space-measure-pmf*]
integrable-pmf-iff-bounded[**where** $C = 1$]
prob-space.indep-vars-compose2[*OF prob-space-measure-pmf b*]) *auto*
also have ... = $(\sum l \in L. (\prod x \in \text{set } (fst\ l) \cup \text{set } (snd\ l). \alpha))$
using τ -*check-3* **unfolding** *L-def* **by** (*intro sum.cong prod.cong c*) *auto*
also have ... = $(\sum l \in L. \alpha \wedge (\text{card } (\text{set } (fst\ l) \cup \text{set } (snd\ l))))$
by *simp*
also have ... = ?*R*
unfolding *L-def* α -*def* **by** (*simp add:case-prod-beta*)
finally show ?*thesis* **by** *simp*
qed

lemma *hit-count-prod-pow-eq*:

assumes $i \in B\ j \in B$
assumes *lim-balls-and-bins* $k\ p$
assumes *lim-balls-and-bins* $k\ q$
assumes $s + t \leq k$
shows $(\int \omega. (Z\ i\ \omega) \wedge^s * (Z\ j\ \omega) \wedge^t \partial\ p) = (\int \omega. (Z\ i\ \omega) \wedge^s * (Z\ j\ \omega) \wedge^t \partial\ q)$
unfolding *hit-count-prod-exp*[*OF assms(1,2,5,3)*]
unfolding *hit-count-prod-exp*[*OF assms(1,2,5,4)*]
by *simp*

lemma *hit-count-sum-pow-eq*:

assumes $i \in B\ j \in B$
assumes *lim-balls-and-bins* $k\ p$
assumes *lim-balls-and-bins* $k\ q$
assumes $s \leq k$
shows $(\int \omega. (Z\ i\ \omega + Z\ j\ \omega) \wedge^s \partial\ p) = (\int \omega. (Z\ i\ \omega + Z\ j\ \omega) \wedge^s \partial\ q)$
(is ?*L* = ?*R***)**

proof –

have *q2*: $|Z\ i\ x \wedge^l * Z\ j\ x \wedge^{(s-l)}| \leq \text{real } (\text{card } R \wedge^s)$
if $l \in \{..s\}$ **for** $s\ i\ j\ l\ x$

proof –

have $|Z\ i\ x \wedge^l * Z\ j\ x \wedge^{(s-l)}| \leq Z\ i\ x \wedge^l * Z\ j\ x \wedge^{(s-l)}$

unfolding *Z-def* **by** *auto*

also have ... $\leq \text{real } (\text{card } R) \wedge^l * \text{real } (\text{card } R) \wedge^{(s-l)}$

unfolding *Z-def*

by (*intro mult-mono power-mono of-nat-mono card-mono fin-R*) *auto*

also have ... = $\text{real } (\text{card } R) \wedge^s$ **using** *that*

by (*subst power-add*[*symmetric*]) *simp*

also have ... = $\text{real } (\text{card } R \wedge^s)$

by *simp*

finally show ?*thesis* **by** *simp*

qed

have ?*L* = $(\int \omega. (\sum l \leq s. \text{real } (s\ \text{choose } l) * (Z\ i\ \omega \wedge^l * Z\ j\ \omega \wedge^{(s-l)})) \partial\ p)$

by (*subst binomial-ring*) (*simp add:algebra-simps*)

also have ... = $(\sum l \leq s. (\int \omega. \text{real } (s\ \text{choose } l) * (Z\ i\ \omega \wedge^l * Z\ j\ \omega \wedge^{(s-l)})) \partial\ p)$

by (*intro integral-sum integrable-mult-right*

integrable-pmf-iff-bounded[**where** $C = \text{card } R \wedge^s$] *q2*) *auto*

also have ... = $(\sum l \leq s. \text{real } (s \text{ choose } l) * (\int \omega. (Z i \omega^l * Z j \omega^{s-l}) \partial p))$
by *(intro sum.cong integral-mult-right integrable-pmf-iff-bounded[where C=card R^s] q2) auto*
also have ... = $(\sum l \leq s. \text{real } (s \text{ choose } l) * (\int \omega. (Z i \omega^l * Z j \omega^{s-l}) \partial q))$
using *assms(5)*
by *(intro-cong [\sigma_2 (*)] more: sum.cong hit-count-prod-pow-eq[OF assms(1-4)]) auto*
also have ... = $(\sum l \leq s. (\int \omega. \text{real } (s \text{ choose } l) * (Z i \omega^l * Z j \omega^{s-l}) \partial q))$
by *(intro sum.cong integral-mult-right[symmetric] integrable-pmf-iff-bounded[where C=card R^s] q2) auto*
also have ... = $(\int \omega. (\sum l \leq s. \text{real } (s \text{ choose } l) * (Z i \omega^l * Z j \omega^{s-l}))) \partial q$
by *(intro integral-sum[symmetric] integrable-mult-right integrable-pmf-iff-bounded[where C=card R^s] q2) auto*
also have ... = ?R
by *(subst binomial-ring) (simp add: algebra-simps)*
finally show ?thesis **by** *simp*
qed

lemma *hit-count-sum-poly-eq:*

assumes $i \in B \ j \in B$
assumes *lim-balls-and-bins k p*
assumes *lim-balls-and-bins k q*
assumes $f \in \mathbb{P} \ k$
shows $(\int \omega. f (Z i \omega + Z j \omega) \partial p) = (\int \omega. f (Z i \omega + Z j \omega) \partial q)$
(is ?L = ?R)

proof –

obtain *fp* **where** *f-def: f = poly fp degree fp ≤ k*
using *assms(5) unfolding Polynomials-def by auto*

have ?L = $(\sum d \leq \text{degree } fp. (\int \omega. \text{poly.coeff } fp \ d * (Z i \omega + Z j \omega)^d \partial p))$
unfolding *f-def poly-altdef*
by *(intro integral-sum integrable-mult-right Z-any-integrable-2[OF assms(3)])*
also have ... = $(\sum d \leq \text{degree } fp. \text{poly.coeff } fp \ d * (\int \omega. (Z i \omega + Z j \omega)^d \partial p))$
by *(intro sum.cong integral-mult-right Z-any-integrable-2[OF assms(3)])*
simp
also have ... = $(\sum d \leq \text{degree } fp. \text{poly.coeff } fp \ d * (\int \omega. (Z i \omega + Z j \omega)^d \partial q))$
using *f-def*
by *(intro sum.cong arg-cong2[where f=(*)] hit-count-sum-pow-eq[OF assms(1-4)]) auto*
also have ... = $(\sum d \leq \text{degree } fp. (\int \omega. \text{poly.coeff } fp \ d * (Z i \omega + Z j \omega)^d \partial q))$
by *(intro sum.cong) auto*
also have ... = ?R
unfolding *f-def poly-altdef by (intro integral-sum[symmetric] integrable-mult-right Z-any-integrable-2[OF assms(4)])*
finally show ?thesis **by** *simp*

qed

lemma *hit-count-poly-eq:*

assumes $b \in B$
assumes *lim-balls-and-bins k p*
assumes *lim-balls-and-bins k q*
assumes $f \in \mathbb{P} \ k$
shows $(\int \omega. f (Z b \omega) \partial p) = (\int \omega. f (Z b \omega) \partial q)$ **(is** ?L = ?R)

proof –

have $a: (\lambda a. f (a / 2)) \in \mathbb{P} (k * 1)$
by *(intro Polynomials-composeI[OF assms(4)] Polynomials-intros)*
have ?L = $\int \omega. f ((Z b \omega + Z b \omega) / 2) \partial p$
by *simp*
also have ... = $\int \omega. f ((Z b \omega + Z b \omega) / 2) \partial q$

using a **by** $(\text{intro hit-count-sum-poly-eq}[OF \text{ assms}(1,1,2,3)]) \text{ simp}$
also have $\dots = ?R$ **by** simp
finally show $?thesis$ **by** simp
qed

lemma *lim-balls-and-bins-from-ind-balls-and-bins:*

lim-balls-and-bins $k \ \Omega$

proof –

have $\text{prob-space.indep-vars} (\text{measure-pmf} \ \Omega) (\lambda-. \text{discrete}) (\lambda x \ \omega. \ \omega \ x) \ R$
unfolding $\Omega\text{-def}$ **using** $\text{indep-vars-Pi-pmf}[OF \ \text{fin-R}]$ **by** metis
hence $\text{prob-space.indep-vars} (\text{measure-pmf} \ \Omega) (\lambda-. \text{discrete}) (\lambda x \ \omega. \ \omega \ x) \ J$ **if** $J \subseteq R$ **for** J
using $\text{prob-space.indep-vars-subset}[OF \ \text{prob-space-measure-pmf} - \text{that}]$ **by** auto
hence $a:\text{prob-space.k-wise-indep-vars} (\text{measure-pmf} \ \Omega) \ k (\lambda-. \text{discrete}) (\lambda x \ \omega. \ \omega \ x) \ R$
by $(\text{simp add:prob-space.k-wise-indep-vars-def}[OF \ \text{prob-space-measure-pmf}])$

have $b:\text{map-pmf} (\lambda \omega. \ \omega \ x) \ \Omega = \text{pmf-of-set} \ B$ **if** $x \in R$ **for** x
using that **unfolding** $\Omega\text{-def}$ $\text{Pi-pmf-component}[OF \ \text{fin-R}]$ **by** simp

show $?thesis$

using $a \ b \ \text{fin-R} \ \text{fin-B}$ **unfolding** $\text{lim-balls-and-bins-def}$ **by** auto

qed

lemma *hit-count-factorial-moments:*

assumes $a:j \in B$

assumes $s \leq k$

assumes $\text{lim-balls-and-bins} \ k \ p$

shows $(\int \omega. \ \text{ffact} \ s \ (Z \ j \ \omega) \ \partial p) = \text{ffact} \ s \ (\text{real} \ (\text{card} \ R)) * (1 / \text{real} \ (\text{card} \ B))^{\wedge} s$
(is $?L = ?R$ **)**

proof –

have $(\lambda x. \ \text{ffact} \ s \ (x-0::\text{real})) \in \mathbb{P} \ s$

by $(\text{intro Polynomials-intros})$

hence $b:\text{ffact} \ s \in (\mathbb{P} \ k :: (\text{real} \Rightarrow \text{real}) \ \text{set})$

using $\text{Polynomials-mono}[OF \ \text{assms}(2)]$ **by** auto

have $?L = (\int \omega. \ \text{ffact} \ s \ (Z \ j \ \omega) \ \partial \Omega)$

by $(\text{intro hit-count-poly-eq}[OF \ a \ \text{assms}(3) \ \text{lim-balls-and-bins-from-ind-balls-and-bins}] \ b)$

also have $\dots = (\int \omega. \ \text{ffact} \ s \ (\sum i \in \{j\}. \ Z \ i \ \omega) \ \partial \Omega)$

by simp

also have $\dots = \text{ffact} \ s \ (\text{real} \ (\text{card} \ R)) * (\text{real} \ (\text{card} \ \{j\}) / \text{real} \ (\text{card} \ B))^{\wedge} s$

using $\text{assms}(1)$

by $(\text{intro fact-moment-balls-and-bins} \ \text{fin-R} \ \text{fin-B}) \ \text{auto}$

also have $\dots = ?R$

by simp

finally show $?thesis$ **by** simp

qed

lemma *hit-count-factorial-moments-2:*

assumes $a:i \in B \ j \in B$

assumes $i \neq j \ s \leq k \ \text{card} \ R \leq \text{card} \ B$

assumes $\text{lim-balls-and-bins} \ k \ p$

shows $(\int \omega. \ \text{ffact} \ s \ (Z \ i \ \omega + Z \ j \ \omega) \ \partial p) \leq 2^{\wedge} s$

(is $?L \leq ?R$ **)**

proof –

have $(\lambda x. \ \text{ffact} \ s \ (x-0::\text{real})) \in \mathbb{P} \ s$

by $(\text{intro Polynomials-intros})$

hence $b:\text{ffact} \ s \in (\mathbb{P} \ k :: (\text{real} \Rightarrow \text{real}) \ \text{set})$

using $\text{Polynomials-mono}[OF \ \text{assms}(4)]$ **by** auto

have *or-distrib*: $(a \wedge b) \vee (a \wedge c) \longleftrightarrow a \wedge (b \vee c)$ **for** $a\ b\ c$
by *auto*
have $?L = (\int \omega. \text{ffact } s\ (Z\ i\ \omega + Z\ j\ \omega)\ \partial\Omega)$
by (*intro hit-count-sum-poly-eq*[*OF a assms(6) lim-balls-and-bins-from-ind-balls-and-bins*] *b*)
also have $\dots = (\int \omega. \text{ffact } s\ ((\sum t \in \{i,j\}. Z\ t\ \omega))\ \partial\Omega)$
using *assms(3)* **by** *simp*
also have $\dots = \text{ffact } s\ (\text{real } (\text{card } R)) * (\text{real } (\text{card } \{i,j\}) / \text{real } (\text{card } B)) \wedge^s$
using *assms(1,2)*
by (*intro fact-moment-balls-and-bins fin-R fin-B*) *auto*
also have $\dots = \text{real } (\text{ffact } s\ (\text{card } R)) * (\text{real } (\text{card } \{i,j\}) / \text{real } (\text{card } B)) \wedge^s$
by (*simp add:of-nat-ffact*)
also have $\dots \leq (\text{card } R) \wedge^s * (\text{real } (\text{card } \{i,j\}) / \text{real } (\text{card } B)) \wedge^s$
by (*intro mult-mono of-nat-mono ffact-bound, simp-all*)
also have $\dots \leq (\text{card } B) \wedge^s * (\text{real } (2) / \text{real } (\text{card } B)) \wedge^s$
using *assms(3)*
by (*intro mult-mono of-nat-mono power-mono assms(5), simp-all*)
also have $\dots = ?R$
using *card-B-gt-0* **by** (*simp add:divide-simps*)
finally show *?thesis* **by** *simp*
qed

lemma *balls-and-bins-approx-helper*:

fixes $x :: \text{real}$
assumes $x \geq 2$
assumes $\text{real } k \geq 5 * x / \ln x$
shows $k \geq 2$
and $2^{k+3} / \text{fact } k \leq (1 / \exp x) \wedge^2$
and $2 / \text{fact } k \leq 1 / (\exp 1 * \exp x)$

proof –

have *ln-inv*: $\ln x = - \ln (1 / x)$ **if** $x > 0$ **for** $x :: \text{real}$
using *that* **by** (*subst ln-div, auto*)

have *apx*:

$\exp 1 \leq (5 :: \text{real})$
 $4 * \ln 4 \leq (2 - 2 * \exp 1 / 5) * \ln (450 :: \text{real})$
 $\ln 8 * 2 \leq (450 :: \text{real})$
 $4 / 5 * 2 * \exp 1 + \ln (5 / 4) * \exp 1 \leq (5 :: \text{real})$
 $\exp 1 \leq (2 :: \text{real}) \wedge^4$
by (*approximation 10*)**+**

have $2 \leq 5 * (x / (x-1))$
using *assms(1)* **by** (*simp add:divide-simps*)
also have $\dots \leq 5 * (x / \ln x)$
using *assms(1)*
by (*intro mult-left-mono divide-left-mono ln-le-minus-one mult-pos-pos*) *auto*
also have $\dots \leq k$ **using** *assms(2)* **by** *simp*
finally show *k-ge-2*: $k \geq 2$ **by** *simp*

have $\ln x * (2 * \exp 1) = \ln (((4/5) * x) * (5/4)) * (2 * \exp 1)$
by *simp*
also have $\dots = \ln ((4/5) * x) * (2 * \exp 1) + \ln((5/4)) * (2 * \exp 1)$
using *assms(1)* **by** (*subst ln-mult, simp-all add:algebra-simps*)
also have $\dots < (4/5) * x * (2 * \exp 1) + \ln (5/4) * (x * \exp 1)$
using *assms(1)* **by** (*intro add-less-le-mono mult-strict-right-mono ln-less-self*
mult-left-mono mult-right-mono) (*auto simp add:algebra-simps*)
also have $\dots = ((4/5) * 2 * \exp 1 + \ln(5/4) * \exp 1) * x$
by (*simp add:algebra-simps*)
also have $\dots \leq 5 * x$

using *assms(1) apx(4)* **by** (*intro mult-right-mono, simp-all*)
finally have $1: \ln x * (2 * \exp 1) \leq 5 * x$ **by** *simp*

have $\ln 8 \leq 3 * x - 5 * x * \ln(2 * \exp 1 / 5 * \ln x) / \ln x$
proof (*cases x ∈ {2..450}*)
case *True*
then show *?thesis* **by** (*approximation 10 splitting: x=10*)
next
case *False*
hence *x-ge-450*: $x \geq 450$ **using** *assms(1)* **by** *simp*

have $4 * \ln 4 \leq (2 - 2 * \exp 1 / 5) * \ln(450::\text{real})$
using *apx(2)* **by** (*simp*)
also have $\dots \leq (2 - 2 * \exp 1 / 5) * \ln x$
using *x-ge-450 apx(1)*
by (*intro mult-left-mono iffD2[OF ln-le-cancel-iff], simp-all*)
finally have $(2 - 2 * \exp 1 / 5) * \ln x \geq 4 * \ln 4$ **by** *simp*
hence $2 * \exp 1 / 5 * \ln x + 0 \leq 2 * \exp 1 / 5 * \ln x + ((2 - 2 * \exp 1 / 5) * \ln x - 4 * \ln 4)$
by (*intro add-mono*) *auto*
also have $\dots = 4 * (1/2) * \ln x - 4 * \ln 4$
by (*simp add:algebra-simps*)
also have $\dots = 4 * (\ln(x \text{ powr } (1/2)) - \ln 4)$
using *x-ge-450* **by** (*subst ln-powr, auto*)
also have $\dots = 4 * (\ln(x \text{ powr } (1/2)/4))$
using *x-ge-450* **by** (*subst ln-div auto*)
also have $\dots < 4 * (x \text{ powr } (1/2)/4)$
using *x-ge-450* **by** (*intro mult-strict-left-mono ln-less-self*) *auto*
also have $\dots = x \text{ powr } (1/2)$ **by** *simp*
finally have $2 * \exp 1 / 5 * \ln x \leq x \text{ powr } (1/2)$ **by** *simp*
hence $\ln(2 * \exp 1 / 5 * \ln x) \leq \ln(x \text{ powr } (1/2))$
using *x-ge-450 ln-le-cancel-iff* **by** *simp*
hence $0: \ln(2 * \exp 1 / 5 * \ln x) / \ln x \leq 1/2$
using *x-ge-450* **by** (*subst (asm) ln-powr, auto*)
have $\ln 8 \leq 3 * x - 5 * x * (1/2)$
using *x-ge-450 apx(3)* **by** *simp*
also have $\dots \leq 3 * x - 5 * x * (\ln(2 * \exp 1 / 5 * \ln x) / \ln x)$
using *x-ge-450* **by** (*intro diff-left-mono mult-left-mono 0*) *auto*
finally show *?thesis* **by** *simp*

qed

hence $2 * x + \ln 8 \leq 2 * x + (3 * x - 5 * x * \ln(2 * \exp 1 / 5 * \ln x) / \ln x)$
by (*intro add-mono, auto*)
also have $\dots = 5 * x + 5 * x * \ln(5 / (2 * \exp 1 * \ln x)) / \ln x$
using *assms(1)* **by** (*subst ln-inv*) (*auto simp add:algebra-simps*)
also have $\dots = 5 * x * (\ln x + \ln(5 / (2 * \exp 1 * \ln x))) / \ln x$
using *assms(1)* **by** (*simp add:algebra-simps add-divide-distrib*)
also have $\dots = 5 * x * (\ln(5 * x / (2 * \exp 1 * \ln x))) / \ln x$
using *assms(1)* **by** (*subst ln-mult[symmetric], auto*)
also have $\dots = (5 * x / \ln x) * \ln((5 * x / \ln x) / (2 * \exp 1))$
by (*simp add:algebra-simps*)
also have $\dots \leq k * \ln(k / (2 * \exp 1))$
using *assms(1,2) 1 k-ge-2*
by (*intro mult-mono iffD2[OF ln-le-cancel-iff] divide-right-mono*)
auto
finally have $k * \ln(k / (2 * \exp 1)) \geq 2 * x + \ln 8$ **by** *simp*
hence $k * \ln(2 * \exp 1 / k) \leq -2 * x - \ln 8$
using *k-ge-2* **by** (*subst ln-inv, auto*)
hence $\ln((2 * \exp 1 / k) \text{ powr } k) \leq \ln(\exp(-2 * x)) - \ln 8$

```

    using k-ge-2 by (subst ln-powr, auto)
  also have ... = ln(exp(-2*x)/8)
    by (simp add:ln-div)
  finally have ln ((2*exp 1/k) powr k) ≤ ln (exp(-2*x)/8) by simp
  hence 1: (2*exp 1/k) powr k ≤ exp(-2*x)/8
    using k-ge-2 assms(1) by (subst (asm) ln-le-cancel-iff) auto
  have 2^(k+3)/fact k ≤ 2^(k+3)/(k / exp 1)^k
    using k-ge-2 by (intro divide-left-mono fact-lower-bound-1) auto
  also have ... = 8 * 2^k * (exp 1 / k)^k
    by (simp add:power-add algebra-simps power-divide)
  also have ... = 8 * (2*exp 1/k) powr k
    using k-ge-2 powr-realpow
    by (simp add:power-mult-distrib[symmetric])
  also have ... ≤ 8 * (exp(-2*x)/8)
    by (intro mult-left-mono 1) auto
  also have ... = exp((-x)*2)
    by simp
  also have ... = exp(-x)^2
    by (subst exp-powr[symmetric], simp)
  also have ... = (1/exp x)^2
    by (simp add: exp-minus inverse-eq-divide)
  finally show 2:2^(k+3)/fact k ≤ (1/exp x)^2 by simp

  have (2::real)/fact k = (2^(k+3)/fact k)/(2^(k+2))
    by (simp add:divide-simps power-add)
  also have ... ≤ (1/exp x)^2/(2^(k+2))
    by (intro divide-right-mono 2, simp)
  also have ... ≤ (1/exp x)^1/(2^(k+2))
    using assms(1) by (intro divide-right-mono power-decreasing) auto
  also have ... ≤ (1/exp x)^1/(2^4)
    using k-ge-2 by (intro divide-left-mono power-increasing) auto
  also have ... ≤ (1/exp x)^1/exp(1)
    using k-ge-2 apx(5) by (intro divide-left-mono) auto
  also have ... = 1/(exp 1 * exp x) by simp
  finally show (2::real)/fact k ≤ 1/(exp 1 * exp x) by simp

```

qed

Bounds on the expectation and variance in the k-wise independent case. Here the independence assumption is improved by a factor of two compared to the result in the paper.

lemma

```

  assumes card R ≤ card B
  assumes  $\bigwedge c. \text{lim-balls-and-bins } (k+1) (p c)$ 
  assumes  $\varepsilon \in \{0 < .. 1/\text{exp}(2)\}$ 
  assumes  $k \geq 5 * \ln(\text{card } B / \varepsilon) / \ln(\ln(\text{card } B / \varepsilon))$ 
  shows
    exp-approx:  $|\text{measure-pmf.expectation } (p \text{ True}) Y - \text{measure-pmf.expectation } (p \text{ False}) Y| \leq$ 
       $\varepsilon * \text{real } (\text{card } R) \text{ (is ?A) and}$ 
    var-approx:  $|\text{measure-pmf.variance } (p \text{ True}) Y - \text{measure-pmf.variance } (p \text{ False}) Y| \leq \varepsilon^2$ 
      (is ?B)

```

proof –

```

  let ?p1 = p False
  let ?p2 = p True

```

```

  have exp (2::real) = 1 / (1/exp 2) by simp
  also have ... ≤ 1 /  $\varepsilon$ 
    using assms(3) by (intro divide-left-mono) auto
  also have ... ≤ real (card B) /  $\varepsilon$ 
    using assms(3) card-B-gt-0 by (intro divide-right-mono) auto

```

finally have $\exp 2 \leq \text{real } (\text{card } B) / \varepsilon$ **by** *simp*
hence *k-condition-h*: $2 \leq \ln (\text{card } B / \varepsilon)$
using *assms(3) card-B-gt-0* **by** (*subst ln-ge-iff*) *auto*
have *k-condition-h-2*: $0 < \text{real } (\text{card } B) / \varepsilon$
using *assms(3) card-B-gt-0* **by** (*intro divide-pos-pos*) *auto*

note *k-condition* = *balls-and-bins-approx-helper*[*OF k-condition-h assms(4)*]

define $\varphi :: \text{real} \Rightarrow \text{real}$ **where** $\varphi = (\lambda x. \text{min } x \ 1)$

define *f* **where** $f = (\lambda x. 1 - (-1)^{\wedge}k / \text{real } (\text{fact } k) * \text{ffact } k (x-1))$
define *g* **where** $g = (\lambda x. \varphi x - f x)$
have *φ-exp*: $\varphi x = f x + g x$ **for** *x*
unfolding *g-def* **by** *simp*

have *k-ge-2*: $k \geq 2$
using *k-condition(1)* **by** *simp*

define γ **where** $\gamma = 1 / \text{real } (\text{fact } k)$
have *γ-nonneg*: $\gamma \geq 0$
unfolding *γ-def* **by** *simp*

have *k-le-k-plus-1*: $k \leq k+1$
by *simp*

have $f \in \mathbb{P} k$
unfolding *f-def* **by** (*intro Polynomials-intros*)
hence *f-poly*: $f \in \mathbb{P} (k+1)$
using *Polynomials-mono*[*OF k-le-k-plus-1*] **by** *auto*

have *g-diff*: $|g x - g (x-1)| = \text{ffact } (k-1) (x-2) / \text{fact } (k-1)$
if $x \geq k$ **for** *x* :: *real*
proof –
have $x \geq 2$ **using** *k-ge-2* **that** **by** *simp*
hence $\varphi x = \varphi (x-1)$
unfolding *φ-def* **by** *simp*
hence $|g x - g (x-1)| = |f (x-1) - f x|$
unfolding *g-def* **by** (*simp add:algebra-simps*)
also have $\dots = |(-1)^{\wedge}k / \text{real } (\text{fact } k) * (\text{ffact } k (x-2) - \text{ffact } k (x-1))|$
unfolding *f-def* **by** (*simp add:algebra-simps*)
also have $\dots = 1 / \text{real } (\text{fact } k) * |\text{ffact } k (x-1) - \text{ffact } k ((x-1)-1)|$
by (*simp add:abs-mult*)
also have $\dots = 1 / \text{real } (\text{fact } k) * \text{real } k * |\text{ffact } (k-1) (x-2)|$
by (*subst ffact-suc-diff, simp add:abs-mult*)
also have $\dots = |\text{ffact } (k-1) (x-2)| / \text{fact } (k-1)$
using *k-ge-2* **by** (*subst fact-reduce*) *auto*
also have $\dots = \text{ffact } (k-1) (x-2) / \text{fact } (k-1)$
unfolding *ffact-eq-fact-mult-binomial* **using** *that k-ge-2*
by (*intro arg-cong2*[**where** $f=(/)$]) *abs-of-nonneg ffact-nonneg*) *auto*
finally show *?thesis* **by** *simp*

qed

have *f-approx-φ*: $f x = \varphi x$ **if** *f-approx-φ-1*: $x \in \text{real } \{0..k\}$ **for** *x*
proof (*cases x = 0*)
case *True*
hence $f x = 1 - (-1)^{\wedge}k / \text{real } (\text{fact } k) * (\prod_{i=0..<k.} - (\text{real } i+1))$
unfolding *f-def prod-ffact*[*symmetric*] **by** (*simp add:algebra-simps*)

also have ... = $1 - (-1)^k / \text{real } (\text{fact } k) * ((\prod_{i=0..<k} (-1)::\text{real}) * (\prod_{i=0..<k} \text{real } i+1))$

by (subst prod.distrib[symmetric]) simp

also have ... = $1 - (-1)^k / \text{real } (\text{fact } k) * ((-1)^k * (\prod_{i \in (\lambda x. x + 1) \text{ ' } \{0..<k\}. \text{real } i}))$

by (subst prod.reindex, auto simp add:inj-on-def comp-def algebra-simps)

also have ... = $1 - (-1)^k / \text{real } (\text{fact } k) * ((-1)^k * (\prod_{i \in \{1..k\}. \text{real } i}))$

by (intro arg-cong2[where f=(-)] arg-cong2[where f=(*)] prod.cong refl) auto

also have ... = 0

unfolding fact-prod by simp

also have ... = φx

using True φ -def by simp

finally show ?thesis by simp

next

case False

hence a: $x \geq 1$ using that by auto

obtain x' where x' -def: $x' \in \{0..k\}$ $x = \text{real } x'$

using f-approx- φ -1 by auto

hence $x' - 1 \in \{0..<k\}$ using k-ge-2 by simp

moreover have $x - \text{real } 1 = \text{real } (x' - 1)$

using False x' -def(2) by simp

ultimately have b: $x - 1 = \text{real } (x' - 1)$ $x' - 1 < k$

by auto

have $f x = 1 - (-1)^k / \text{real } (\text{fact } k) * \text{real } (\text{ffact } k (x' - 1))$

unfolding f-def b of-nat-ffact by simp

also have ... = 1

using b by (subst ffact-nat-triv, auto)

also have ... = φx

unfolding φ -def using a by auto

finally show ?thesis by simp

qed

have q2: $|Z i x^l * Z j x^{(s-l)}| \leq \text{real } (\text{card } R)^s$

if $l \in \{..s\}$ for $s i j l x$

proof -

have $|Z i x^l * Z j x^{(s-l)}| \leq Z i x^l * Z j x^{(s-l)}$

unfolding Z-def by auto

also have ... $\leq \text{real } (\text{card } R)^l * \text{real } (\text{card } R)^{(s-l)}$

unfolding Z-def

by (intro mult-mono power-mono of-nat-mono card-mono fin-R) auto

also have ... = $\text{real } (\text{card } R)^s$ using that

by (subst power-add[symmetric]) simp

also have ... = $\text{real } (\text{card } R)^s$

by simp

finally show ?thesis by simp

qed

have q: $\text{real } (\text{card } A) + \text{real } (\text{card } B) \in \text{real } \{..2 * \text{card } R\}$ if $A \subseteq R$ $B \subseteq R$ for $A B$

proof -

have $\text{card } A + \text{card } B \leq \text{card } R + \text{card } R$

by (intro add-mono card-mono fin-R that)

also have ... = $2 * \text{card } R$ by simp

finally show ?thesis by force

qed

have g-eq-0-iff-2: $\text{abs } (g x) * y = 0$ if $x \in \mathbb{Z}$ $x \geq 0$ $x \leq k$ for $x y :: \text{real}$

proof -

have $\exists x'. x = \text{real-of-int } x' \wedge x' \leq k \wedge x' \geq 0$

using *that Ints-def by fastforce*
hence $\exists x'. x = \text{real } x' \wedge x' \leq k$
by (*metis nat-le-iff of-nat-nat*)
hence $x \in \text{real } \{0..k\}$
by *auto*
hence $g x = 0$
unfolding *g-def using f-approx-φ by simp*
thus *?thesis by simp*
qed

have *g-bound-abs*: $|\int \omega. g (f \omega) \partial p| \leq (\int \omega. \text{ffact } (k+1) (f \omega) \partial p) * \gamma$
(is *?L ≤ ?R*)
if $\text{range } f \subseteq \text{real } \{..m\}$ **for** m **and** $p :: ('a \Rightarrow 'b) \text{ pmf}$ **and** $f :: ('a \Rightarrow 'b) \Rightarrow \text{real}$
proof –
have *f-any-integrable*:
integrable p (λω. h (f ω)) for h :: real ⇒ real
using *that*
by (*intro integrable-pmf-iff-bounded[where C=Max (abs ' h ' real ' {..m})]*
Max-ge finite-imageI imageI) *auto*

have *f-val*: $f \omega \in \text{real } \{..m\}$ **for** ω **using** *that by auto*
hence *f-nat*: $f \omega \in \mathbb{N}$ **for** ω
unfolding *Nats-def by auto*

have *f-int*: $f \omega \geq \text{real } y + 1$ **if** $f \omega > \text{real } y$ **for** $y \omega$
proof –
obtain x **where** *x-def*: $f \omega = \text{real } x$ $x \leq m$ **using** *f-val by auto*
hence $y < x$ **using** *that by simp*
hence $y + 1 \leq x$ **by** *simp*
then show *?thesis using x-def by simp*
qed

have *f-nonneg*: $f \omega \geq 0$ **for** ω
proof –
obtain x **where** *x-def*: $f \omega = \text{real } x$ $x \leq m$ **using** *f-val by auto*
hence $x \geq 0$ **by** *simp*
then show *?thesis using x-def by simp*
qed

have $\neg(\text{real } x \leq f \omega)$ **if** $x > m$ **for** $x \omega$
proof –
obtain x **where** *x-def*: $f \omega = \text{real } x$ $x \leq m$ **using** *f-val by auto*
then show *?thesis using x-def that by simp*
qed

hence *max-Z1*: $\text{measure } p \{\omega. \text{real } x \leq f \omega\} = 0$ **if** $x > m$ **for** x
using *that by auto*

have $?L \leq (\int \omega. |g (f \omega)| \partial p)$
by (*intro integral-abs-bound*)
also have $\dots = (\sum y \in \text{real } \{..m\}. |g y| * \text{measure } p \{\omega. f \omega = y\})$
using *that by (intro pmf-exp-of-fin-function) auto*
also have $\dots = (\sum y \in \{..m\}. |g (\text{real } y)| * \text{measure } p \{\omega. f \omega = \text{real } y\})$
by (*subst sum.reindex*) (*auto simp add:comp-def*)
also have $\dots = (\sum y \in \{..m\}. |g (\text{real } y)| * (\text{measure } p (\{\omega. f \omega = \text{real } y\} \cup \{\omega. f \omega > y\}) - \text{measure } p \{\omega. f \omega > y\}))$
by (*subst measure-Union*) *auto*
also have $\dots = (\sum y \in \{..m\}. |g (\text{real } y)| * (\text{measure } p \{\omega. f \omega \geq y\} - \text{measure } p \{\omega. f \omega >$

$y\}))$
by (*intro sum.cong arg-cong2[where f=(*)] arg-cong2[where f=(-)] arg-cong[where f=measure p]*) *auto*
also have ... = $(\sum_{y \in \{..m\}} |g(\text{real } y)| * \text{measure } p \{\omega. f \ \omega \geq y\}) -$
 $(\sum_{y \in \{..m\}} |g(\text{real } y)| * \text{measure } p \{\omega. f \ \omega > y\})$
by (*simp add:algebra-simps sum-subtractf*)
also have ... = $(\sum_{y \in \{..m\}} |g(\text{real } y)| * \text{measure } p \{\omega. f \ \omega \geq y\}) -$
 $(\sum_{y \in \{..m\}} |g(\text{real } y)| * \text{measure } p \{\omega. f \ \omega \geq \text{real } (y+1)\})$
using *f-int*
by (*intro sum.cong arg-cong2[where f=(-)] arg-cong2[where f=(*)] arg-cong[where f=measure p]*) *fastforce+*
also have ... = $(\sum_{y \in \{..m\}} |g(\text{real } y)| * \text{measure } p \{\omega. f \ \omega \geq \text{real } y\}) -$
 $(\sum_{y \in \text{Suc } \{..m\}} |g(\text{real } y - 1)| * \text{measure } p \{\omega. f \ \omega \geq \text{real } y\})$
by (*subst sum.reindex*) (*auto simp add:comp-def*)
also have ... = $(\sum_{y \in \{..m\}} |g(\text{real } y)| * \text{measure } p \{\omega. f \ \omega \geq \text{real } y\}) -$
 $(\sum_{y \in \{1..m\}} |g(\text{real } y - 1)| * \text{measure } p \{\omega. f \ \omega \geq \text{real } y\})$
using *max-Z1 image-Suc-atMost*
by (*intro arg-cong2[where f=(-)] sum.mono-neutral-cong*) *auto*
also have ... = $(\sum_{y \in \{k+1..m\}} |g(\text{real } y)| * \text{measure } p \{\omega. f \ \omega \geq y\}) -$
 $(\sum_{y \in \{k+1..m\}} |g(\text{real } y - 1)| * \text{measure } p \{\omega. f \ \omega \geq y\})$
using *k-ge-2*
by (*intro arg-cong2[where f=(-)] sum.mono-neutral-cong-right ballI g-eq-0-iff-2*)
auto
also have ... = $(\sum_{y \in \{k+1..m\}} (|g(\text{real } y)| - |g(\text{real } y - 1)|) * \text{measure } p \{\omega. f \ \omega \geq y\})$
by (*simp add:algebra-simps sum-subtractf*)
also have ... $\leq (\sum_{y \in \{k+1..m\}} |g(\text{real } y) - g(\text{real } y - 1)| * \text{measure } p \{\omega. \text{ffact } (k+1) (f \ \omega) \geq \text{ffact } (k+1) (\text{real } y)\})$
using *ffact-mono* **by** (*intro sum-mono mult-mono pmf-mono*) *auto*
also have ... = $(\sum_{y \in \{k+1..m\}} (\text{ffact } (k-1) (\text{real } y - 2) / \text{fact } (k-1)) * \text{measure } p \{\omega. \text{ffact } (k+1) (f \ \omega) \geq \text{ffact } (k+1) (\text{real } y)\})$
by (*intro sum.cong, simp-all add: g-diff*)
also have ... $\leq (\sum_{y \in \{k+1..m\}} (\text{ffact } (k-1) (\text{real } y - 2) / \text{fact } (k-1)) * ((f \ \omega. \text{ffact } (k+1) (f \ \omega) \partial p) / \text{ffact } (k+1) (\text{real } y)))$
using *k-ge-2 f-nat*
by (*intro sum-mono mult-left-mono pmf-markov f-any-integrable divide-nonneg-pos ffact-of-nat-nonneg ffact-pos*) *auto*
also have ... = $(\int \omega. \text{ffact } (k+1) (f \ \omega) \partial p) / \text{fact } (k-1) * (\sum_{y \in \{k+1..m\}} \text{ffact } (k-1) (\text{real } y - 2) / \text{ffact } (\text{Suc } (\text{Suc } (k-1))) (\text{real } y))$
using *k-ge-2* **by** (*simp add:algebra-simps sum-distrib-left*)
also have ... = $(\int \omega. \text{ffact } (k+1) (f \ \omega) \partial p) / \text{fact } (k-1) * (\sum_{y \in \{k+1..m\}} \text{ffact } (k-1) (\text{real } y - 2) / (\text{real } y * (\text{real } y - 1) * \text{ffact } (k-1) (\text{real } y - 2)))$
by (*subst ffact-Suc, subst ffact-Suc, simp*)
also have ... = $(\int \omega. \text{ffact } (k+1) (f \ \omega) \partial p) / \text{fact } (k-1) * (\sum_{y \in \{k+1..m\}} 1 / (\text{real } y * (\text{real } y - 1)))$
using *order.strict-implies-not-eq[OF ffact-pos]* *k-ge-2*
by (*intro arg-cong2[where f=(*)] sum.cong*) *auto*
also have ... = $(\int \omega. \text{ffact } (k+1) (f \ \omega) \partial p) / \text{fact } (k-1) * (\sum_{y \in \{\text{Suc } k..m\}} 1 / (\text{real } y - 1) - 1 / (\text{real } y))$
using *k-ge-2* **by** (*intro arg-cong2[where f=(*)] sum.cong*) (*auto simp add: divide-simps*)
also have ... = $(\int \omega. \text{ffact } (k+1) (f \ \omega) \partial p) / \text{fact } (k-1) * (\sum_{y \in \{\text{Suc } k..m\}} (-1 / (\text{real } y)) - (-1 / (\text{real } (y - 1))))$
using *k-ge-2* **by** (*intro arg-cong2[where f=(*)] sum.cong*) (*auto*)
also have ... = $(\int \omega. \text{ffact } (k+1) (f \ \omega) \partial p) / \text{fact } (k-1) * (\text{of-bool } (k \leq m) * (1 / \text{real } k - 1 / \text{real } m))$
by (*subst sum-telescope-eq, auto*)
also have ... $\leq (\int \omega. \text{ffact } (k+1) (f \ \omega) \partial p) / \text{fact } (k-1) * (1 / \text{real } k)$
using *k-ge-2 f-nat*
by (*intro mult-left-mono divide-nonneg-nonneg integral-nonneg*)

$\text{ffact-of-nat-nonneg}$ auto
also have ... = ?R
using $k\text{-ge-2}$ **unfolding** $\gamma\text{-def}$ **by** (cases k) (auto simp add: algebra-simps)
finally show ?thesis **by** simp
qed

have $z1\text{-g-bound}$: $|\int \omega. g (Z i \omega) \partial p c| \leq (\text{real} (\text{card } R) / \text{real} (\text{card } B)) * \gamma$
(is ?L1 \leq ?R1) **if** $i \in B$ **for** $i c$
proof –

have ?L1 \leq $(\int \omega. \text{ffact} (k+1) (Z i \omega) \partial p c) * \gamma$
unfolding $Z\text{-def}$ **using** fin-R
by (intro $g\text{-bound-abs}$ [**where** $m1 = \text{card } R$]) (auto intro!: imageI card-mono)
also have ... = $\text{ffact} (k+1) (\text{real} (\text{card } R)) * (1 / \text{real} (\text{card } B))^{\wedge(k+1)} * \gamma$
using *that* **by** ($\text{subst hit-count-factorial-moments}[OF - - \text{assms}(2)]$, simp-all)
also have ... = $\text{real} (\text{ffact} (k+1) (\text{card } R)) * (1 / \text{real} (\text{card } B))^{\wedge(k+1)} * \gamma$
by ($\text{simp add: of-nat-ffact}$)
also have ... $\leq \text{real} (\text{card } R)^{\wedge(k+1)} * (1 / \text{real} (\text{card } B))^{\wedge(k+1)} * \gamma$
using $\gamma\text{-nonneg}$
by (intro $\text{mult-right-mono of-nat-mono ffact-bound}$, simp-all)
also have ... $\leq (\text{real} (\text{card } R) / \text{real} (\text{card } B))^{\wedge(k+1)} * \gamma$
by ($\text{simp add: divide-simps}$)
also have ... $\leq (\text{real} (\text{card } R) / \text{real} (\text{card } B))^{\wedge 1} * \gamma$
using $\text{assms}(1)$ card-B-gt-0 $\gamma\text{-nonneg}$ **by** (intro $\text{mult-right-mono power-decreasing}$) auto
also have ... = ?R1 **by** simp
finally show ?thesis **by** simp
qed

have $g\text{-add-bound}$: $|\int \omega. g (Z i \omega + Z j \omega) \partial p c| \leq 2^{\wedge(k+1)} * \gamma$
(is ?L1 \leq ?R1) **if** $ij\text{-in-B}$: $i \in B j \in B i \neq j$ **for** $i j c$
proof –

have ?L1 \leq $(\int \omega. \text{ffact} (k+1) (Z i \omega + Z j \omega) \partial p c) * \gamma$
unfolding $Z\text{-def}$ **using** $\text{assms}(1)$
by (intro $g\text{-bound-abs}$ [**where** $m1 = 2 * \text{card } R$]) (auto intro!: $\text{imageI } q$)
also have ... $\leq 2^{\wedge(k+1)} * \gamma$
by (intro $\gamma\text{-nonneg mult-right-mono hit-count-factorial-moments-2}[OF \text{that}(1,2,3) - \text{assms}(1,2)]$)
auto
finally show ?thesis **by** simp
qed

have $Z\text{-poly-diff}$:
 $|\int \omega. \varphi (Z i \omega) \partial^?p1 - \int \omega. \varphi (Z i \omega) \partial^?p2| \leq 2 * ((\text{real} (\text{card } R) / \text{card } B) * \gamma)$
(is ?L $\leq 2 * ?R$) **if** $i \in B$ **for** i
proof –

note $Z\text{-poly-eq} =$
 $\text{hit-count-poly-eq}[OF \text{that } \text{assms}(2)[\text{of True}] \text{assms}(2)[\text{of False}] f\text{-poly}]$

have ?L = $|\int \omega. f (Z i \omega) \partial^?p1 + \int \omega. g (Z i \omega) \partial^?p1 - \int \omega. f (Z i \omega) \partial^?p2 - \int \omega. g (Z i \omega) \partial^?p2|$
using $Z\text{-integrable}[OF \text{assms}(2)]$ **unfolding** $\varphi\text{-exp}$ **by** simp
also have ... = $|\int \omega. g (Z i \omega) \partial^?p1 + (- \int \omega. g (Z i \omega) \partial^?p2)|$
by ($\text{subst } Z\text{-poly-eq}$) auto
also have ... $\leq |\int \omega. g (Z i \omega) \partial^?p1| + |\int \omega. g (Z i \omega) \partial^?p2|$
by simp
also have ... $\leq ?R + ?R$
by (intro $\text{add-mono } z1\text{-g-bound}$ *that*)
also have ... = $2 * ?R$
by ($\text{simp add: algebra-simps}$)

finally show ?thesis by simp
qed

have Z-poly-diff-2: $|\int \omega. \varphi (Z i \omega) \partial^?p1 - \int \omega. \varphi (Z i \omega) \partial^?p2| \leq 2 * \gamma$
(is ?L ≤ ?R) if $i \in B$ for i

proof –

have ?L ≤ 2 * ((real (card R) / real (card B)) * γ)

by (intro Z-poly-diff that)

also have ... ≤ 2 * (1 * γ)

using assms fin-B that γ -nonneg card-gt-0-iff

by (intro mult-mono that iffD2[OF pos-divide-le-eq]) auto

also have ... = ?R by simp

finally show ?thesis by simp

qed

have Z-poly-diff-3: $|\int \omega. \varphi (Z i \omega + Z j \omega) \partial^?p2 - \int \omega. \varphi (Z i \omega + Z j \omega) \partial^?p1| \leq$
 $2^{(k+2)*\gamma}$

(is ?L ≤ ?R) if $i \in B j \in B i \neq j$ for $i j$

proof –

note Z-poly-eq-2 =

hit-count-sum-poly-eq[OF that(1,2) assms(2)[of True] assms(2)[of False] f-poly]

have ?L = $|\int \omega. f (Z i \omega + Z j \omega) \partial^?p2 + \int \omega. g (Z i \omega + Z j \omega) \partial^?p2 -$
 $\int \omega. f (Z i \omega + Z j \omega) \partial^?p1 - \int \omega. g (Z i \omega + Z j \omega) \partial^?p1|$

using Z-any-integrable-2[OF assms(2)] unfolding φ -exp by simp

also have ... = $|\int \omega. g (Z i \omega + Z j \omega) \partial^?p2 + (- \int \omega. g (Z i \omega + Z j \omega) \partial^?p1)|$

by (subst Z-poly-eq-2) auto

also have ... ≤ $|\int \omega. g (Z i \omega + Z j \omega) \partial^?p1| + |\int \omega. g (Z i \omega + Z j \omega) \partial^?p2|$

by simp

also have ... ≤ $2^{(k+1)*\gamma} + 2^{(k+1)*\gamma}$

by (intro add-mono g-add-bound that)

also have ... = ?R

by (simp add: algebra-simps)

finally show ?thesis by simp

qed

have Y-eq: $Y \omega = (\sum i \in B. \varphi (Z i \omega))$ if $\omega \in \text{set-pmf } (p c)$ for $c \omega$

proof –

have $\omega \text{ ' } R \subseteq B$

proof (rule image-subsetI)

fix x assume $a: x \in R$

have $\omega x \in \text{set-pmf } (\text{map-pmf } (\lambda \omega. \omega x) (p c))$

using that by (subst set-map-pmf) simp

also have ... = $\text{set-pmf } (\text{pmf-of-set } B)$

by (intro arg-cong[where $f = \text{set-pmf}$] assms ran[OF assms(2)] a)

also have ... = B

by (intro set-pmf-of-set fin-B B-ne)

finally show $\omega x \in B$ by simp

qed

hence $(\omega \text{ ' } R) = B \cap \omega \text{ ' } R$

by auto

hence $Y \omega = \text{card } (B \cap \omega \text{ ' } R)$

unfolding Y-def by auto

also have ... = $(\sum i \in B. \text{of-bool } (i \in \omega \text{ ' } R))$

unfolding of-bool-def using fin-B by (subst sum.If-cases) auto

also have ... = $(\sum i \in B. \text{of-bool } (\text{card } \{r \in R. \omega r = i\} > 0))$

using fin-R by (intro sum.cong arg-cong[where $f = \text{of-bool}$])

(auto simp add:card-gt-0-iff)
 also have ... = $(\sum i \in B. \varphi(Z i \omega))$
 unfolding φ -def Z-def by (intro sum.cong) (auto simp add:of-bool-def)
 finally show ?thesis by simp
 qed

let $?\varphi 2 = (\lambda x y. \varphi x + \varphi y - \varphi (x+y))$
 let $?Bd = \{x \in B \times B. \text{fst } x \neq \text{snd } x\}$

have $Y\text{-sq-eq}' : Y \omega \wedge 2 = (\sum i \in ?Bd. ?\varphi 2 (Z (\text{fst } i) \omega) (Z (\text{snd } i) \omega)) + Y \omega$
 (is $?L = ?R$) if $\omega \in \text{set-pmf } (p \ c)$ for $c \ \omega$

proof –

have a: $\varphi (Z x \omega) = \text{of-bool}(\text{card } \{r \in R. \omega \ r = x\} > 0)$ for x

unfolding φ -def Z-def by auto

have b: $\varphi (Z x \omega + Z y \omega) =$

$\text{of-bool}(\text{card } \{r \in R. \omega \ r = x\} > 0 \vee \text{card } \{r \in R. \omega \ r = y\} > 0)$ for $x \ y$

unfolding φ -def Z-def by auto

have c: $\varphi (Z x \omega) * \varphi (Z y \omega) = ?\varphi 2 (Z x \omega) (Z y \omega)$ for $x \ y$

unfolding a b of-bool-def by auto

have d: $\varphi (Z x \omega) * \varphi (Z x \omega) = \varphi (Z x \omega)$ for x

unfolding a of-bool-def by auto

have $?L = (\sum i \in B \times B. \varphi (Z (\text{fst } i) \omega) * \varphi (Z (\text{snd } i) \omega))$

unfolding Y -eq[OF that] power2-eq-square sum-product sum.cartesian-product

by (simp add:case-prod-beta)

also have ... = $(\sum i \in ?Bd \cup \{x \in B \times B. \text{fst } x = \text{snd } x\}. \varphi (Z (\text{fst } i) \omega) * \varphi (Z (\text{snd } i) \omega))$

by (intro sum.cong refl) auto

also have ... = $(\sum i \in ?Bd. \varphi (Z (\text{fst } i) \omega) * \varphi (Z (\text{snd } i) \omega)) +$

$(\sum i \in \{x \in B \times B. \text{fst } x = \text{snd } x\}. \varphi (Z (\text{fst } i) \omega) * \varphi (Z (\text{snd } i) \omega))$

using assms fin-B by (intro sum.union-disjoint, auto)

also have ... = $(\sum i \in ?Bd. ?\varphi 2 (Z (\text{fst } i) \omega) (Z (\text{snd } i) \omega)) +$

$(\sum i \in \{x \in B \times B. \text{fst } x = \text{snd } x\}. \varphi (Z (\text{fst } i) \omega) * \varphi (Z (\text{fst } i) \omega))$

unfolding c by (intro arg-cong2[where f=(+)] sum.cong) auto

also have ... = $(\sum i \in ?Bd. ?\varphi 2 (Z (\text{fst } i) \omega) (Z (\text{snd } i) \omega)) +$

$(\sum i \in \text{fst } \{x \in B \times B. \text{fst } x = \text{snd } x\}. \varphi (Z i \omega) * \varphi (Z i \omega))$

by (subst sum.reindex, auto simp add:inj-on-def)

also have ... = $(\sum i \in ?Bd. ?\varphi 2 (Z (\text{fst } i) \omega) (Z (\text{snd } i) \omega)) + (\sum i \in B. \varphi (Z i \omega))$

using d by (intro arg-cong2[where f=(+)] sum.cong refl d) (auto simp add:image-iff)

also have ... = $?R$

unfolding Y -eq[OF that] by simp

finally show ?thesis by simp

qed

have $|\text{integral}^L ?p1 \ Y - \text{integral}^L ?p2 \ Y| =$

$|(\int \omega. (\sum i \in B. \varphi(Z i \omega)) \partial ?p1) - (\int \omega. (\sum i \in B. \varphi(Z i \omega)) \partial ?p2)|$

by (intro arg-cong[where f=abs] arg-cong2[where f=(-)]

integral-cong-AE AE-pmfI Y-eq) auto

also have ... =

$|(\sum i \in B. (\int \omega. \varphi(Z i \omega) \partial ?p1)) - (\sum i \in B. (\int \omega. \varphi(Z i \omega) \partial ?p2))|$

by (intro arg-cong[where f=abs] arg-cong2[where f=(-)]

integral-sum Z-integrable[OF assms(2)])

also have ... = $|(\sum i \in B. (\int \omega. \varphi(Z i \omega) \partial ?p1) - (\int \omega. \varphi(Z i \omega) \partial ?p2))|$

by (subst sum-subtractf) simp

also have ... $\leq (\sum i \in B. |(\int \omega. \varphi(Z i \omega) \partial ?p1) - (\int \omega. \varphi(Z i \omega) \partial ?p2)|)$

by simp

also have ... $\leq (\sum i \in B. 2 * ((\text{real } (\text{card } R) / \text{real } (\text{card } B)) * \gamma))$

by (intro sum-mono Z-poly-diff)

also have ... $\leq 2 * \text{real } (\text{card } R) * \gamma$

using γ -nonneg by (simp)
 finally have Y -exp-diff-1: $|\text{integral}^L \text{?}p1 Y - \text{integral}^L \text{?}p2 Y| \leq 2 * \text{real} (\text{card } R) * \gamma$
 by simp

have $|\text{integral}^L \text{?}p1 Y - \text{integral}^L \text{?}p2 Y| \leq (2 / \text{fact } k) * \text{real} (\text{card } R)$
 using Y -exp-diff-1 by (simp add: algebra-simps γ -def)
 also have $\dots \leq 1 / (\text{exp } 1 * (\text{real} (\text{card } B) / \varepsilon)) * \text{card } R$
 using k -condition(3) k -condition-h-2 by (intro mult-right-mono) auto
 also have $\dots = \varepsilon / (\text{exp } 1 * \text{real} (\text{card } B)) * \text{card } R$
 by simp
 also have $\dots \leq \varepsilon / (1 * 1) * \text{card } R$
 using $\text{assms}(3)$ card-B-gt-0
 by (intro mult-right-mono divide-left-mono mult-mono) auto
 also have $\dots = \varepsilon * \text{card } R$
 by simp
 finally show ?A
 by simp

have $|\text{integral}^L \text{?}p1 Y - \text{integral}^L \text{?}p2 Y| \leq 2 * \text{real} (\text{card } R) * \gamma$
 using Y -exp-diff-1 by simp
 also have $\dots \leq 2 * \text{real} (\text{card } B) * \gamma$
 by (intro mult-mono of-nat-mono $\text{assms } \gamma$ -nonneg) auto
 finally have Y -exp-diff-2:
 $|\text{integral}^L \text{?}p1 Y - \text{integral}^L \text{?}p2 Y| \leq 2 * \gamma * \text{real} (\text{card } B)$
 by (simp add: algebra-simps)

have int-Y : integrable (measure-pmf (p c)) Y for c
 using fin-R card-image-le unfolding Y -def
 by (intro integrable-pmf-iff-bounded[where $C = \text{card } R$]) auto

have int-Y-sq : integrable (measure-pmf (p c)) ($\lambda \omega. Y \omega \wedge 2$) for c
 using fin-R card-image-le unfolding Y -def
 by (intro integrable-pmf-iff-bounded[where $C = \text{real} (\text{card } R) \wedge 2$]) auto

have $|(\int \omega. (\sum i \in \text{?}Bd. \text{?}\varphi 2 (Z (\text{fst } i) \omega) (Z (\text{snd } i) \omega)) \partial \text{?}p1) -$
 $(\int \omega. (\sum i \in \text{?}Bd. \text{?}\varphi 2 (Z (\text{fst } i) \omega) (Z (\text{snd } i) \omega)) \partial \text{?}p2)|$
 $\leq |(\sum i \in \text{?}Bd.$
 $(\int \omega. \varphi (Z (\text{fst } i) \omega) \partial \text{?}p1) + (\int \omega. \varphi (Z (\text{snd } i) \omega) \partial \text{?}p1) -$
 $(\int \omega. \varphi (Z (\text{fst } i) \omega + Z (\text{snd } i) \omega) \partial \text{?}p1) - ((\int \omega. \varphi (Z (\text{fst } i) \omega) \partial \text{?}p2) +$
 $(\int \omega. \varphi (Z (\text{snd } i) \omega) \partial \text{?}p2) - (\int \omega. \varphi (Z (\text{fst } i) \omega + Z (\text{snd } i) \omega) \partial \text{?}p2)))|$ (is ?R3 $\leq -$)
 using Z -integrable[OF $\text{assms}(2)$] Z -any-integrable-2[OF $\text{assms}(2)$]
 by (simp add: integral-sum sum-subtractf)
 also have $\dots = |(\sum i \in \text{?}Bd.$
 $((\int \omega. \varphi (Z (\text{fst } i) \omega) \partial \text{?}p1) - (\int \omega. \varphi (Z (\text{fst } i) \omega) \partial \text{?}p2)) +$
 $((\int \omega. \varphi (Z (\text{snd } i) \omega) \partial \text{?}p1) - (\int \omega. \varphi (Z (\text{snd } i) \omega) \partial \text{?}p2)) +$
 $((\int \omega. \varphi (Z (\text{fst } i) \omega + Z (\text{snd } i) \omega) \partial \text{?}p2) - (\int \omega. \varphi (Z (\text{fst } i) \omega + Z (\text{snd } i) \omega) \partial \text{?}p1)))|$
 by (intro arg-cong[where $f = \text{abs}$] sum.cong) auto
 also have $\dots \leq (\sum i \in \text{?}Bd. |$
 $((\int \omega. \varphi (Z (\text{fst } i) \omega) \partial \text{?}p1) - (\int \omega. \varphi (Z (\text{fst } i) \omega) \partial \text{?}p2)) +$
 $((\int \omega. \varphi (Z (\text{snd } i) \omega) \partial \text{?}p1) - (\int \omega. \varphi (Z (\text{snd } i) \omega) \partial \text{?}p2)) +$
 $((\int \omega. \varphi (Z (\text{fst } i) \omega + Z (\text{snd } i) \omega) \partial \text{?}p2) - (\int \omega. \varphi (Z (\text{fst } i) \omega + Z (\text{snd } i) \omega) \partial \text{?}p1)))|$
 by (intro sum-abs)
 also have $\dots \leq (\sum i \in \text{?}Bd.$
 $|(\int \omega. \varphi (Z (\text{fst } i) \omega) \partial \text{?}p1) - (\int \omega. \varphi (Z (\text{fst } i) \omega) \partial \text{?}p2)| +$
 $|(\int \omega. \varphi (Z (\text{snd } i) \omega) \partial \text{?}p1) - (\int \omega. \varphi (Z (\text{snd } i) \omega) \partial \text{?}p2)| +$
 $|(\int \omega. \varphi (Z (\text{fst } i) \omega + Z (\text{snd } i) \omega) \partial \text{?}p2) - (\int \omega. \varphi (Z (\text{fst } i) \omega + Z (\text{snd } i) \omega) \partial \text{?}p1)|)$
 by (intro sum-mono) auto
 also have $\dots \leq (\sum i \in \text{?}Bd. 2 * \gamma + 2 * \gamma + 2 \wedge (k+2) * \gamma)$

by (intro sum-mono add-mono Z-poly-diff-2 Z-poly-diff-3) auto
also have ... = $(2^{(k+2)+4}) * \gamma * \text{real}(\text{card } ?Bd)$
by (simp add: algebra-simps)
finally have $Y\text{-sq-exp-diff-1}: ?R3 \leq (2^{(k+2)+4}) * \gamma * \text{real}(\text{card } ?Bd)$
by simp

have $|(\int \omega. Y \omega \hat{2} \partial ?p1) - (\int \omega. Y \omega \hat{2} \partial ?p2)| =$
 $|(\int \omega. (\sum i \in ?Bd. ?\varphi 2 (Z (fst i) \omega) (Z (snd i) \omega)) + Y \omega \partial ?p1) -$
 $(\int \omega. (\sum i \in ?Bd. ?\varphi 2 (Z (fst i) \omega) (Z (snd i) \omega)) + Y \omega \partial ?p2)|$
by (intro-cong $[\sigma_2 (-), \sigma_1 \text{abs}]$ more: integral-cong-AE AE-pmfI Y-sq-eq') auto
also have ... $\leq |(\int \omega. Y \omega \partial ?p1) - (\int \omega. Y \omega \partial ?p2)| +$
 $|(\int \omega. (\sum i \in ?Bd. ?\varphi 2 (Z (fst i) \omega) (Z (snd i) \omega)) \partial ?p1) -$
 $(\int \omega. (\sum i \in ?Bd. ?\varphi 2 (Z (fst i) \omega) (Z (snd i) \omega)) \partial ?p2)|$
using Z-integrable[OF assms(2)] Z-any-integrable-2[OF assms(2)] int-Y by simp
also have ... $\leq 2 * \gamma * \text{real}(\text{card } B) + ?R3$
by (intro add-mono Y-exp-diff-2, simp)
also have ... $\leq (2^{(k+2)+4}) * \gamma * \text{real}(\text{card } B) + (2^{(k+2)+4}) * \gamma * \text{real}(\text{card } ?Bd)$
using γ -nonneg by (intro add-mono Y-sq-exp-diff-1 mult-right-mono) auto
also have ... = $(2^{(k+2)+4}) * \gamma * (\text{real}(\text{card } B) + \text{real}(\text{card } ?Bd))$
by (simp add: algebra-simps)
also have ... = $(2^{(k+2)+4}) * \gamma * \text{real}(\text{card } B) \hat{2}$
using power2-nat-le-imp-le
by (simp add: card-distinct-pairs of-nat-diff)
finally have Y-sq-exp-diff:
 $|(\int \omega. Y \omega \hat{2} \partial ?p1) - (\int \omega. Y \omega \hat{2} \partial ?p2)| \leq (2^{(k+2)+4}) * \gamma * \text{real}(\text{card } B) \hat{2}$ by simp

have Y-exp-rough-bound: $|\text{integral}^L (p \ c) \ Y| \leq \text{card } B$ (is $?L \leq ?R$) for c
proof -

have $?L \leq (\int \omega. |Y \omega| \partial (p \ c))$
by (intro integral-abs-bound)
also have ... $\leq (\int \omega. \text{real}(\text{card } R) \partial (p \ c))$
unfolding Y-def using card-image-le[OF fin-R]
by (intro integral-mono integrable-pmf-iff-bounded[where $C = \text{card } R$])
auto
also have ... = $\text{card } R$ by simp
also have ... $\leq \text{card } B$ using assms by simp
finally show ?thesis by simp

qed

have $|\text{measure-pmf.variance } ?p1 \ Y - \text{measure-pmf.variance } ?p2 \ Y| =$
 $|(\int \omega. Y \omega \hat{2} \partial ?p1) - (\int \omega. Y \omega \partial ?p1) \hat{2} - ((\int \omega. Y \omega \hat{2} \partial ?p2) - (\int \omega. Y \omega \partial ?p2) \hat{2})|$
by (intro-cong $[\sigma_2 (-), \sigma_1 \text{abs}]$ more: measure-pmf.variance-eq int-Y int-Y-sq)
also have ... $\leq |(\int \omega. Y \omega \hat{2} \partial ?p1) - (\int \omega. Y \omega \hat{2} \partial ?p2)| + |(\int \omega. Y \omega \partial ?p1)^2 - (\int \omega. Y \omega$
 $\partial ?p2)^2|$
by simp
also have ... = $|(\int \omega. Y \omega \hat{2} \partial ?p1) - (\int \omega. Y \omega \hat{2} \partial ?p2)| +$
 $|(\int \omega. Y \omega \partial ?p1) - (\int \omega. Y \omega \partial ?p2)| * |(\int \omega. Y \omega \partial ?p1) + (\int \omega. Y \omega \partial ?p2)|$
by (simp add: power2-eq-square algebra-simps abs-mult[symmetric])
also have ... $\leq (2^{(k+2)+4}) * \gamma * \text{real}(\text{card } B) \hat{2} + (2 * \gamma * \text{real}(\text{card } B)) *$
 $(|\int \omega. Y \omega \partial ?p1| + |\int \omega. Y \omega \partial ?p2|)$
using γ -nonneg
by (intro add-mono mult-mono divide-left-mono Y-sq-exp-diff Y-exp-diff-2) auto
also have ... $\leq (2^{(k+2)+4}) * \gamma * \text{real}(\text{card } B) \hat{2} + (2 * \gamma * \text{real}(\text{card } B)) *$
 $(\text{real}(\text{card } B) + \text{real}(\text{card } B))$
using γ -nonneg by (intro add-mono mult-left-mono Y-exp-rough-bound) auto
also have ... = $(2^{(k+2)+2} \hat{3}) * \gamma * \text{real}(\text{card } B) \hat{2}$
by (simp add: algebra-simps power2-eq-square)
also have ... $\leq (2^{(k+2)+2} \hat{(k+2)}) * \gamma * \text{real}(\text{card } B) \hat{2}$

```

using k-ge-2 γ-nonneg
by (intro mult-right-mono add-mono power-increasing, simp-all)
also have ... =  $(2^{k+3} / \text{fact } k) * \text{card } B^2$ 
by (simp add:power-add γ-def)
also have ...  $\leq (1 / (\text{real } (\text{card } B) / \varepsilon))^2 * \text{card } B^2$ 
using k-condition(2) k-condition-h-2
by (intro mult-right-mono) auto
also have ... =  $\varepsilon^2$ 
using card-B-gt-0 by (simp add:divide-simps)
finally show ?B
by simp
qed

```

lemma

```

assumes card R  $\leq$  card B
assumes lim-balls-and-bins (k+1) p
assumes k  $\geq 7.5 * (\ln (\text{card } B) + 2)$ 
shows exp-approx-2:  $|\text{measure-pmf.expectation } p \ Y - \mu| \leq \text{card } R / \text{sqrt } (\text{card } B)$ 
  (is ?AL  $\leq$  ?AR)
  and var-approx-2:  $\text{measure-pmf.variance } p \ Y \leq \text{real } (\text{card } R)^2 / \text{card } B$ 
  (is ?BL  $\leq$  ?BR)

```

proof –

```

define q where  $q = (\lambda c. \text{if } c \text{ then } \Omega \text{ else } p)$ 

```

```

have q-altdef:  $q \ \text{True} = \Omega$   $q \ \text{False} = p$ 
unfolding q-def by auto

```

```

have a:lim-balls-and-bins (k+1) (q c) for c
unfolding q-def using assms lim-balls-and-bins-from-ind-balls-and-bins by auto

```

```

define  $\varepsilon :: \text{real}$  where  $\varepsilon = \min (\text{sqrt } (1 / \text{card } B)) (1 / \text{exp } 2)$ 

```

```

have c:  $\varepsilon \in \{0 < .. 1 / \text{exp } 2\}$ 
using card-B-gt-0 unfolding ε-def by auto

```

```

have b:  $5 * \ln (\text{card } B / \varepsilon) / \ln (\ln (\text{card } B / \varepsilon)) \leq \text{real } k$ 

```

```

proof (cases card B  $\geq$  exp 4)

```

```

case True

```

```

hence  $\text{sqrt}(1 / \text{card } B) \leq \text{sqrt}(1 / \text{exp } 4)$ 

```

```

using card-B-gt-0 by (intro real-sqrt-le-mono divide-left-mono) auto

```

```

also have ... =  $(1 / \text{exp } 2)$ 

```

```

by (subst powr-half-sqrt[symmetric]) (auto simp add:powr-divide exp-powr)

```

```

finally have  $\text{sqrt}(1 / \text{card } B) \leq (1 / \text{exp } 2)$  by simp

```

```

hence ε-eq:  $\varepsilon = \text{sqrt}(1 / \text{card } B)$ 

```

```

unfolding ε-def by simp

```

```

have exp (6::real) =  $(\text{exp } 4) \ \text{powr } (3/2)$ 

```

```

by (simp add:exp-powr)

```

```

also have ...  $\leq \text{card } B \ \text{powr } (3/2)$ 

```

```

by (intro powr-mono2 True) auto

```

```

finally have q4:exp 6  $\leq \text{card } B \ \text{powr } (3/2)$  by simp

```

```

have (2::real)  $\leq \text{exp } 6$ 

```

```

by (approximation 5)

```

```

hence q1:  $2 \leq \text{real } (\text{card } B) \ \text{powr } (3 / 2)$ 

```

```

using q4 by argo

```

```

have (1::real)  $< \ln(\text{exp } 6)$ 

```

```

by (approximation 5)

```

also have $\dots \leq \ln (\text{card } B \text{ powr } (3 / 2))$
using *card-B-gt-0* **by** (*intro iffD2[OF ln-le-cancel-iff] q4*) *auto*
finally have $q2: 1 < \ln (\text{card } B \text{ powr } (3 / 2))$ **by** *simp*
have $\exp (\exp (1::\text{real})) \leq \exp 6$
by (*approximation 5*)
also have $\dots \leq \text{card } B \text{ powr } (3/2)$ **using** $q4$ **by** *simp*
finally have $\exp (\exp 1) \leq \text{card } B \text{ powr } (3/2)$
by *simp*
hence $q3: 1 \leq \ln (\ln (\text{card } B \text{ powr } (3/2)))$
using *card-B-gt-0 q1* **by** (*intro iffD2[OF ln-ge-iff] ln-gt-zero, auto*)

have $5 * \ln (\text{card } B / \varepsilon) / \ln (\ln (\text{card } B / \varepsilon)) =$
 $5 * \ln (\text{card } B \text{ powr } (1+1/2)) / \ln (\ln (\text{card } B \text{ powr } (1+1/2)))$
unfolding *powr-add* **by** (*simp add:real-sqrt-divide powr-half-sqrt[symmetric] ε-eq*)
also have $\dots \leq 5 * \ln (\text{card } B \text{ powr } (1+1/2)) / 1$
using *True q1 q2 q3* **by** (*intro divide-left-mono mult-nonneg-nonneg mult-pos-pos*
ln-ge-zero ln-gt-zero) *auto*
also have $\dots = 5 * (1+1/2) * \ln(\text{card } B)$
using *card-B-gt-0* **by** (*subst ln-powr*) *auto*
also have $\dots = 7.5 * \ln(\text{card } B)$ **by** *simp*
also have $\dots \leq k$ **using** *assms(3)* **by** *simp*
finally show *?thesis* **by** *simp*

next
case *False*
have $(1::\text{real}) / \exp 2 \leq \sqrt{1 / \exp 4}$
by (*simp add:real-sqrt-divide powr-half-sqrt[symmetric] exp-powr*)
also have $\dots \leq \sqrt{1 / \text{card } B}$
using *False card-B-gt-0*
by (*intro real-sqrt-le-mono divide-left-mono mult-pos-pos*) *auto*
finally have $1 / \exp 2 \leq \sqrt{1 / \text{card } B}$
by *simp*
hence $\varepsilon\text{-eq}: \varepsilon = 1 / \exp 2$
unfolding $\varepsilon\text{-def}$ **by** *simp*

have $q2: 5 * (\ln x + 2) / \ln (\ln x + 2) \leq 7.5 * (\ln x + 2)$
if $x \in \{1.. \exp 4\}$ **for** $x::\text{real}$
using *that* **by** (*approximation 10 splitting: x=10*)

have $5 * \ln (\text{card } B / \varepsilon) / \ln (\ln (\text{card } B / \varepsilon)) =$
 $5 * (\ln (\text{card } B) + 2) / \ln (\ln (\text{card } B) + 2)$
using *card-B-gt-0* **unfolding** $\varepsilon\text{-eq}$ **by** (*simp add:ln-mult*)
also have $\dots \leq 7.5 * (\ln (\text{card } B) + 2)$
using *False card-B-gt-0* **by** (*intro q2*) *auto*
also have $\dots \leq k$ **using** *assms(3)* **by** *simp*
finally show *?thesis* **by** *simp*

qed

have $?AL = |(\int \omega. Y \omega \partial(q \text{ True})) - (\int \omega. Y \omega \partial(q \text{ False}))|$
using *exp-balls-and-bins* **unfolding** $q\text{-def}$ **by** *simp*
also have $\dots \leq \varepsilon * \text{card } R$
by (*intro exp-approx[OF assms(1) a c b]*)
also have $\dots \leq \sqrt{1 / \text{card } B} * \text{card } R$
unfolding $\varepsilon\text{-def}$ **by** (*intro mult-right-mono*) *auto*
also have $\dots = ?AR$ **using** *real-sqrt-divide* **by** *simp*
finally show $?AL \leq ?AR$ **by** *simp*

show $?BL \leq ?BR$
proof (*cases R = {}*)

```

case True
then show ?thesis unfolding Y-def by simp
next
case False
hence card R > 0 using fin-R by auto
hence card-R-ge-1: real (card R) ≥ 1 by simp

have ?BL ≤ measure-pmf.variance (q True) Y +
  |measure-pmf.variance (q True) Y - measure-pmf.variance (q False) Y|
  unfolding q-def by auto
also have ... ≤ measure-pmf.variance (q True) Y + ε2
  by (intro add-mono var-approx[OF assms(1) a c b]) auto
also have ... ≤ measure-pmf.variance (q True) Y + sqrt(1 / card B)2
  unfolding ε-def by (intro add-mono power-mono) auto
also have ... ≤ card R * (real (card R) - 1) / card B + sqrt(1 / card B)2
  unfolding q-altdef by (intro add-mono var-balls-and-bins) auto
also have ... = card R * (real (card R) - 1) / card B + 1 / card B
  by (auto simp add:power-divide real-sqrt-divide)
also have ... ≤ card R * (real (card R) - 1) / card B + card R / card B
  by (intro add-mono divide-right-mono card-R-ge-1) auto
also have ... = (card R * (real (card R) - 1) + card R) / card B
  by argo
also have ... = ?BR
  by (simp add:algebra-simps power2-eq-square)
finally show ?BL ≤ ?BR by simp
qed
qed

```

lemma devitation-bound:

```

assumes card R ≤ card B
assumes lim-balls-and-bins k p
assumes real k ≥ C2 * ln (real (card B)) + C3
shows measure p {ω. |Y ω - μ| > 9 * real (card R) / sqrt (real (card B))} ≤ 1 / 26
(is ?L ≤ ?R)

```

proof (cases card R > 0)

case True

```

define k' :: nat where k' = k - 1
have (1::real) ≤ 7.5 * 0 + 16 by simp
also have ... ≤ 7.5 * ln (real (card B)) + 16
  using card-B-ge-1 by (intro add-mono mult-left-mono ln-ge-zero) auto
also have ... ≤ k using assms(3) unfolding C2-def C3-def by simp
finally have k-ge-1: k ≥ 1 by simp
have lim: lim-balls-and-bins (k'+1) p
  using k-ge-1 assms(2) unfolding k'-def by simp

```

```

have k'-min: real k' ≥ 7.5 * (ln (real (card B)) + 2)
  using k-ge-1 assms(3) unfolding C2-def C3-def k'-def by simp

```

```

let ?r = real (card R)
let ?b = real (card B)
have a: integrable p (λω. (Y ω)2)
  unfolding Y-def
  by (intro integrable-pmf-iff-bounded[where C=real (card R)2])
  (auto intro!: card-image-le[OF fin-R])

```

```

have ?L ≤ P(ω in measure-pmf p. |Y ω - (∫ ω. Y ω ∂p)| ≥ 8*?r / sqrt ?b)
proof (rule pmf-mono)

```



```

fix  $\omega$  assume  $\omega \in \text{set-pmf } p$ 
assume  $a:\omega \in \{\omega. 9 * \text{real } (\text{card } R) / \text{sqrt } (\text{real } (\text{card } B)) < |Y \omega - \mu|\}$ 
have  $8 * ?r / \text{sqrt } ?b = 9 * ?r / \text{sqrt } ?b - ?r / \text{sqrt } ?b$ 
  by simp
also have  $\dots \leq |Y \omega - \mu| - |(\int \omega. Y \omega \partial p) - \mu|$ 
  using  $a$  by (intro diff-mono exp-approx-2[OF assms(1) lim k'-min]) simp
also have  $\dots \leq |Y \omega - (\int \omega. Y \omega \partial p)|$ 
  by simp
finally have  $8 * ?r / \text{sqrt } ?b \leq |Y \omega - (\int \omega. Y \omega \partial p)|$  by simp
thus  $\omega \in \{\omega \in \text{space } (\text{measure-pmf } p). 8 * ?r / \text{sqrt } ?b \leq |Y \omega - (\int \omega. Y \omega \partial p)|\}$ 
  by simp
qed
also have  $\dots \leq \text{measure-pmf.variance } p Y / (8 * ?r / \text{sqrt } ?b)^2$ 
  using True card-B-gt-0 a
  by (intro measure-pmf.Chebyshev-inequality) auto
also have  $\dots \leq (?r^2 / ?b) / (8 * ?r / \text{sqrt } ?b)^2$ 
  by (intro divide-right-mono var-approx-2[OF assms(1) lim k'-min]) simp
also have  $\dots = 1/2^6$ 
  using card-B-gt-0 True
  by (simp add:power2-eq-square)
finally show ?thesis by simp
next
case False
hence  $R = \{\}$  card R = 0 using fin-R by auto
thus ?thesis
  unfolding Y-def  $\mu$ -def by simp
qed
end

unbundle no-intro-cong-syntax

end

```

5 Tail Bounds for Expander Walks

```

theory Distributed-Distinct-Elements-Tail-Bounds
imports
  Distributed-Distinct-Elements-Preliminary
  Expander-Graphs.Pseudorandom-Objects-Expander-Walks
  HOL-Decision-Procs.Approximation
begin

```

This section introduces tail estimates for random walks in expander graphs, specific to the verification of this algorithm (in particular to two-stage expander graph sampling and obtained tail bounds for subgaussian random variables). They follow from the more fundamental results *regular-graph.kl-chernoff-property* and *regular-graph.uniform-property* which are verified in the AFP entry for expander graphs [10].

```
hide-fact Henstock-Kurzweil-Integration.integral-sum
```

```
unbundle intro-cong-syntax
```

```
lemma x-ln-x-min:
```

```
  assumes  $x \geq (0::\text{real})$ 
```

```
  shows  $x * \ln x \geq -\exp(-1)$ 
```

```
proof -
```

```
  define  $f$  where  $f x = x * \ln x$  for  $x :: \text{real}$ 
```

define f' where $f' x = \ln x + 1$ for $x :: \text{real}$

have $0:(f \text{ has-real-derivative } (f' x)) (at x)$ if $x > 0$ for x
unfolding $f\text{-def}$ $f'\text{-def}$ using that
by (auto intro!: derivative-eq-intros)

have $f' x \geq 0$ if $\exp(-1) \leq x$ for $x :: \text{real}$

proof –

have $\ln x \geq -1$
using that order-less-le-trans[OF exp-gt-zero]
by (intro iffD2[OF ln-ge-iff]) auto
thus ?thesis
unfolding $f'\text{-def}$ by (simp)

qed

hence $\exists y. (f \text{ has-real-derivative } y) (at x) \wedge 0 \leq y$ if $x \geq \exp(-1)$ for $x :: \text{real}$

using that order-less-le-trans[OF exp-gt-zero]
by (intro exI[where $x=f' x$] conjI 0) auto

hence $f(\exp(-1)) \leq f x$ if $\exp(-1) \leq x$

by (intro DERIV-nonneg-imp-nondecreasing[OF that]) auto

hence 2: ?thesis if $\exp(-1) \leq x$

unfolding $f\text{-def}$ using that by simp

have $f' x \leq 0$ if $x > 0$ $x \leq \exp(-1)$ for $x :: \text{real}$

proof –

have $\ln x \leq \ln(\exp(-1))$
by (intro iffD2[OF ln-le-cancel-iff] that exp-gt-zero)
also have $\dots = -1$
by simp
finally have $\ln x \leq -1$ by simp
thus ?thesis unfolding $f'\text{-def}$ by simp

qed

hence $\exists y. (f \text{ has-real-derivative } y) (at x) \wedge y \leq 0$ if $x > 0$ $x \leq \exp(-1)$ for $x :: \text{real}$

using that by (intro exI[where $x=f' x$] conjI 0) auto

hence $f(\exp(-1)) \leq f x$ if $x > 0$ $x \leq \exp(-1)$

using that(1) by (intro DERIV-nonpos-imp-nonincreasing[OF that(2)]) auto

hence 3: ?thesis if $x > 0$ $x \leq \exp(-1)$

unfolding $f\text{-def}$ using that by simp

have ?thesis if $x = 0$

using that by simp

thus ?thesis

using 2 3 assms by fastforce

qed

theorem (in regular-graph) walk-tail-bound:

assumes $l > 0$

assumes $S \subseteq \text{verts } G$

defines $\mu \equiv \text{real } (\text{card } S) / \text{card } (\text{verts } G)$

assumes $\gamma < 1$ $\mu + \Lambda_a \leq \gamma$

shows $\text{measure } (\text{pmf-of-multiset } (\text{walks } G l)) \{w. \text{real } (\text{card } \{i \in \{..<l\}. w ! i \in S\}) \geq \gamma * l\}$
 $\leq \exp(-\text{real } l * (\gamma * \ln(1/(\mu + \Lambda_a)) - 2 * \exp(-1)))$ (is ?L ≤ ?R)

proof (cases $\mu > 0$)

case True

have $0 < \mu + \Lambda_a$

by (intro add-pos-nonneg $\Lambda\text{-ge-0}$ True)

also have $\dots \leq \gamma$
using *assms(5)* **by** *simp*
finally have $\gamma\text{-gt-0}$: $0 < \gamma$ **by** *simp*

hence $\gamma\text{-ge-0}$: $0 \leq \gamma$
by *simp*

have $\text{card } S \leq \text{card } (\text{verts } G)$
by (*intro card-mono assms(2)*) *auto*
hence $\mu\text{-le-1}$: $\mu \leq 1$
unfolding $\mu\text{-def}$ **by** (*simp add:divide-simps*)

have 2 : $0 < \mu + \Lambda_a * (1 - \mu)$
using $\mu\text{-le-1}$ **by** (*intro add-pos-nonneg True mult-nonneg-nonneg $\Lambda\text{-ge-0}$*) *auto*

have $\mu + \Lambda_a * (1 - \mu) \leq \mu + \Lambda_a * 1$
using $\Lambda\text{-ge-0 True}$ **by** (*intro add-mono mult-left-mono*) *auto*
also have $\dots \leq \gamma$
using *assms(5)* **by** *simp*
also have $\dots < 1$
using *assms(4)* **by** *simp*
finally have 4 : $\mu + \Lambda_a * (1 - \mu) < 1$ **by** *simp*
hence 3 : $1 \leq 1 / (\mu + \Lambda_a * (1 - \mu))$
using 2 **by** (*subst pos-le-divide-eq simp-all*)

have $\text{card } S \leq n$
unfolding $n\text{-def}$ **by** (*intro card-mono assms(2)*) *auto*
hence 0 : $\mu \leq 1$
unfolding $\mu\text{-def}$ $n\text{-def}$ [*symmetric*] **using** $n\text{-gt-0}$ **by** *simp*

have $\gamma * \ln (1 / (\mu + \Lambda_a)) - 2 * \exp (- 1) = \gamma * \ln (1 / (\mu + \Lambda_a * 1)) + 0 - 2 * \exp (- 1)$
by *simp*
also have $\dots \leq \gamma * \ln (1 / (\mu + \Lambda_a * (1 - \mu))) + 0 - 2 * \exp (- 1)$
using *True $\gamma\text{-ge-0}$ $\Lambda\text{-ge-0}$ 0 2*
by (*intro diff-right-mono mult-left-mono iffD2[OF ln-le-cancel-iff] divide-pos-pos divide-left-mono add-mono*) *auto*
also have $\dots \leq \gamma * \ln (1 / (\mu + \Lambda_a * (1 - \mu))) + (1 - \gamma) * \ln (1 / (1 - (\mu + \Lambda_a * (1 - \mu)))) - 2 * \exp (- 1)$
using *assms(4) 3* **by** (*intro add-mono diff-mono mult-nonneg-nonneg ln-ge-zero*) *auto*
also have $\dots = (- \exp (- 1)) + \gamma * \ln (1 / (\mu + \Lambda_a * (1 - \mu))) + (- \exp (- 1)) + (1 - \gamma) * \ln (1 / (1 - (\mu + \Lambda_a * (1 - \mu))))$
by *simp*
also have $\dots \leq \gamma * \ln \gamma + \gamma * \ln (1 / (\mu + \Lambda_a * (1 - \mu))) + (1 - \gamma) * \ln (1 - \gamma) + (1 - \gamma) * \ln (1 / (1 - (\mu + \Lambda_a * (1 - \mu))))$
using *assms(4) $\gamma\text{-ge-0}$* **by** (*intro add-mono x-ln-x-min*) *auto*
also have $\dots = \gamma * (\ln \gamma + \ln (1 / (\mu + \Lambda_a * (1 - \mu)))) + (1 - \gamma) * (\ln (1 - \gamma) + \ln (1 / (1 - (\mu + \Lambda_a * (1 - \mu)))))$
by (*simp add:algebra-simps*)
also have $\dots = \gamma * \ln (\gamma * (1 / (\mu + \Lambda_a * (1 - \mu)))) + (1 - \gamma) * \ln ((1 - \gamma) * (1 / (1 - (\mu + \Lambda_a * (1 - \mu)))))$
using 2 4 *assms(4) $\gamma\text{-gt-0}$*
by (*intro-cong [$\sigma_2(+)$, $\sigma_2(*)$] more:ln-mult[symmetric] divide-pos-pos*) *auto*
also have $\dots = \text{KL-div } \gamma (\mu + \Lambda_a * (1 - \mu))$
unfolding *KL-div-def* **by** *simp*
finally have 1 : $\gamma * \ln (1 / (\mu + \Lambda_a)) - 2 * \exp (- 1) \leq \text{KL-div } \gamma (\mu + \Lambda_a * (1 - \mu))$
by *simp*

have $\mu + \Lambda_a * (1 - \mu) \leq \mu + \Lambda_a * 1$
using *True*
by (*intro add-mono mult-left-mono $\Lambda\text{-ge-0}$*) *auto*
also have $\dots \leq \gamma$
using *assms(5)* **by** *simp*
finally have $\mu + \Lambda_a * (1 - \mu) \leq \gamma$ **by** *simp*

moreover have $\mu + \Lambda_a * (1 - \mu) > 0$
using 0 **by** (*intro add-pos-nonneg True mult-nonneg-nonneg Λ -ge-0*) *auto*
ultimately have $\mu + \Lambda_a * (1 - \mu) \in \{0 < .. \gamma\}$ **by** *simp*
hence $?L \leq \exp(-\text{real } l * \text{KL-div } \gamma (\mu + \Lambda_a * (1 - \mu)))$
using *assms(4)* **unfolding** μ -def **by** (*intro kl-bernoulli-property assms(1,2)*) *auto*
also have $\dots \leq ?R$
using *assms(1)* 1 **by** *simp*
finally show *?thesis* **by** *simp*
next
case *False*
hence $\mu \leq 0$ **by** *simp*
hence $\text{card } S = 0$
unfolding μ -def n -def [*symmetric*] **using** n -gt-0 **by** (*simp add:divide-simps*)
moreover have *finite S*
using *finite-subset[OF assms(2) finite-verts]* **by** *auto*
ultimately have $0:S = \{\}$ **by** *auto*
have $\mu = 0$
unfolding μ -def 0 **by** *simp*
hence $\mu + \Lambda_a \geq 0$
using Λ -ge-0 **by** *simp*
hence $\gamma \geq 0$
using *assms(5)* **by** *simp*
hence $\gamma * \text{real } l \geq 0$
by (*intro mult-nonneg-nonneg*) *auto*
thus *?thesis* **using** 0 **by** *simp*
qed

theorem (in regular-graph) walk-tail-bound-2:
assumes $l > 0$ $\Lambda_a \leq \Lambda$ $\Lambda > 0$
assumes $S \subseteq \text{verts } G$
defines $\mu \equiv \text{real } (\text{card } S) / \text{card } (\text{verts } G)$
assumes $\gamma < 1$ $\mu + \Lambda \leq \gamma$
shows $\text{measure } (\text{pmf-of-multiset } (\text{walks } G \ l)) \{w. \text{real } (\text{card } \{i \in \{..<l\}. w \mid i \in S\}) \geq \gamma * l\}$
 $\leq \exp(-\text{real } l * (\gamma * \ln(1/(\mu + \Lambda)) - 2 * \exp(-1)))$ **(is ?L ≤ ?R)**
proof (cases $\mu > 0$)
case *True*

have $0: 0 < \mu + \Lambda_a$
by (*intro add-pos-nonneg Λ -ge-0 True*)
hence $0 < \mu + \Lambda$
using *assms(2)* **by** *simp*
hence $1: 0 < (\mu + \Lambda) * (\mu + \Lambda_a)$
using 0 **by** *simp*

have $3: \mu + \Lambda_a \leq \gamma$
using *assms(2,7)* **by** *simp*
have $2: 0 \leq \gamma$
using 3 *True* Λ -ge-0 **by** *simp*

have $?L \leq \exp(-\text{real } l * (\gamma * \ln(1/(\mu + \Lambda_a)) - 2 * \exp(-1)))$
using 3 **unfolding** μ -def **by** (*intro walk-tail-bound assms(1,4,6)*)
also have $\dots = \exp(-(\text{real } l * (\gamma * \ln(1/(\mu + \Lambda_a)) - 2 * \exp(-1))))$
by *simp*
also have $\dots \leq \exp(-(\text{real } l * (\gamma * \ln(1/(\mu + \Lambda)) - 2 * \exp(-1))))$
using *True* *assms(2,3)* **using** 0 1 2
by (*intro iffD2[OF exp-le-cancel-iff] mult-left-mono diff-mono iffD2[OF ln-le-cancel-iff]*
divide-left-mono le-imp-neg-le) *simp-all*
also have $\dots = ?R$

by *simp*
 finally show *?thesis* by *simp*
 next
 case *False*
 hence $\mu \leq 0$ by *simp*
 hence $\text{card } S = 0$
 unfolding $\mu\text{-def } n\text{-def}[\textit{symmetric}]$ using *n-gt-0* by (*simp add: divide-simps*)
 moreover have *finite S*
 using *finite-subset[OF assms(4) finite-verts]* by *auto*
 ultimately have $0:S = \{\}$ by *auto*
 have $\mu = 0$
 unfolding $\mu\text{-def } 0$ by *simp*
 hence $\mu + \Lambda_a \geq 0$
 using $\Lambda\text{-ge-0}$ by *simp*
 hence $\gamma \geq 0$
 using *assms* by *simp*
 hence $\gamma * \text{real } l \geq 0$
 by (*intro mult-nonneg-nonneg*) *auto*
 thus *?thesis* using 0 by *simp*
 qed

lemma *disjI-safe*: $(\neg x \implies y) \implies x \vee y$ by *auto*

lemma *walk-tail-bound*:

fixes *T*
 assumes $l > 0 \ \Lambda > 0$
 assumes *measure (sample-pro S) {w. T w} $\leq \mu$*
 assumes $\gamma \leq 1 \ \mu + \Lambda \leq \gamma \ \mu \leq 1$
 shows *measure (sample-pro (E l Λ S)) {w. real (card {i \in {.. l }. T (w i))} $\geq \gamma * l$*
 $\leq \text{exp}(-\text{real } l * (\gamma * \ln(1/(\mu + \Lambda)) - 2 * \text{exp}(-1)))$ (*is ?L \leq ?R*)

proof –

have $\mu\text{-ge-0}$: $\mu \geq 0$ using *assms(3) measure-nonneg order.trans* by *metis*
 hence $\gamma\text{-gt-0}$: $\gamma > 0$ using *assms(2,5)* by *auto*
 hence $\gamma\text{-ge-0}$: $\gamma \geq 0$ by *simp*

have $\mu + \Lambda * (1 - \mu) \leq \mu + \Lambda * 1$ using *assms(2,6) $\mu\text{-ge-0}$* by *auto*
 also have $\dots \leq \gamma$ using *assms(5)* by *simp*
 finally have $1:\mu + \Lambda * (1 - \mu) \leq \gamma$ by *simp*

have $2: 0 < \mu + \Lambda * (1 - \mu)$

proof (*cases $\mu = 1$*)

case *True* then show *?thesis* by *simp*

next

case *False*

then show *?thesis* using *assms(2,6)*

by (*intro add-nonneg-pos $\mu\text{-ge-0}$ linordered-semiring-strict-class.mult-pos-pos*) *auto*

qed

have $3: 0 < \mu + \Lambda$ using $\mu\text{-ge-0}$ *assms(2)* by *simp*

have $\gamma * \ln(1 / (\mu + \Lambda)) - 2 * \text{exp}(-1) = \gamma * \ln(1 / (\mu + \Lambda * 1)) + 0 - 2 * \text{exp}(-1)$ by *simp*

also have $\dots \leq \gamma * \ln(1 / (\mu + \Lambda * (1 - \mu))) + 0 - 2 * \text{exp}(-1)$

using $2 \ 3 \ \gamma\text{-ge-0} \ \mu\text{-ge-0} \ \textit{assms}(2)$ by (*intro diff-right-mono add-mono mult-left-mono iffD2[OF ln-le-cancel-iff] divide-left-mono divide-pos-pos*) *simp-all*

also have $\dots \leq \gamma * \ln(1 / (\mu + \Lambda * (1 - \mu))) + (1 - \gamma) * \ln(1 / (1 - (\mu + \Lambda * (1 - \mu)))) - 2 * \text{exp}(-1)$

proof (*cases $\gamma < 1$*)

case *True*

hence $\mu + \Lambda * (1 - \mu) < 1$ using 1 by *simp*

thus *?thesis* **using** *assms(4)* 2
by (*intro diff-right-mono add-mono mult-nonneg-nonneg order.refl ln-ge-zero*) *auto*
next
case *False*
hence $\gamma=1$ **using** *assms(4)* **by** *simp*
thus *?thesis* **by** *simp*
qed
also have ... = $(-exp(-1)) + \gamma * ln(1 / (\mu + \Lambda * (1 - \mu))) + (-exp(-1)) + (1 - \gamma) * ln(1 / (1 - (\mu + \Lambda * (1 - \mu))))$
by *simp*
also have ... $\leq \gamma * ln \gamma + \gamma * ln(1 / (\mu + \Lambda * (1 - \mu))) + (1 - \gamma) * ln(1 - \gamma) + (1 - \gamma) * ln(1 / (1 - (\mu + \Lambda * (1 - \mu))))$
using *assms(4)* γ -ge-0 **by** (*intro add-mono x-ln-x-min*) *auto*
also have ... = $\gamma * (ln \gamma + ln(1 / (\mu + \Lambda * (1 - \mu)))) + (1 - \gamma) * (ln(1 - \gamma) + ln(1 / (1 - (\mu + \Lambda * (1 - \mu))))$
by (*simp add:algebra-simps*)
also have ... = $\gamma * ln(\gamma * (1 / (\mu + \Lambda * (1 - \mu)))) + (1 - \gamma) * ln((1 - \gamma) * (1 / (1 - (\mu + \Lambda * (1 - \mu))))$
using 2 1 *assms(4)* γ -gt-0 **by** (*intro arg-cong2[where f=(+)] iffD2[OF mult-cancel-left]*
disjI-safe ln-mult[symmetric] divide-pos-pos) *auto*
also have ... = $KL-div \gamma (\mu + \Lambda * (1 - \mu))$ **unfolding** *KL-div-def* **by** *simp*
finally have 4: $\gamma * ln(1 / (\mu + \Lambda)) - 2 * exp(-1) \leq KL-div \gamma (\mu + \Lambda * (1 - \mu))$
by *simp*

have $?L \leq exp(-real\ l * KL-div \gamma (\mu + \Lambda * (1 - \mu)))$
using 1 **by** (*intro expander-kl-chernoff-bound assms*)
also have ... $\leq exp(-real\ l * (\gamma * ln(1 / (\mu + \Lambda)) - 2 * exp(-1)))$
by (*intro iffD2[OF exp-le-cancel-iff] mult-left-mono-neg 4*) *auto*
finally show *?thesis* **by** *simp*

qed

definition $C_1 :: real$ **where** $C_1 = exp\ 2 + exp\ 3 + (exp\ 1 - 1)$

lemma *deviation-bound*:

fixes $f :: 'a \Rightarrow real$
assumes $l > 0$
assumes $\Lambda \in \{0 <.. exp(-real\ l * ln(real\ l)^3)\}$
assumes $\bigwedge x. x \geq 20 \implies measure(sample-pro\ S) \{v. f\ v \geq x\} \leq exp(-x * ln\ x^3)$
shows $measure(sample-pro(\mathcal{E}\ l\ \Lambda\ S)) \{\omega. (\sum_{i < l} f(\omega\ i)) \geq C_1 * l\} \leq exp(-real\ l)$ (**is** $?L \leq ?R$)

proof -

let $?w = sample-pro(\mathcal{E}\ l\ \Lambda\ S)$
let $?p = sample-pro\ S$
let $?a = real\ l * (exp\ 2 + exp\ 3)$

define $b :: real$ **where** $b = exp\ 1 - 1$
have b -gt-0: $b > 0$ **unfolding** *b-def* **by** (*approximation 5*)

define L **where**

$L\ k = measure\ ?w \{\omega. exp(real\ k) * card\ \{i \in \{.. < l\}. f(\omega\ i) \geq exp(real\ k)\} \geq real\ l / real\ k^2\}$ **for** k

define k -max **where** k -max = $max\ 4 (MAX\ v \in pro-set\ S. nat\ \lfloor ln(f\ v) \rfloor + 1)$

have k -max-ge-4: k -max ≥ 4 **unfolding** *k-max-def* **by** *simp*
have k -max-ge-3: k -max ≥ 3 **unfolding** *k-max-def* **by** *simp*

have 1: $of_bool(\lfloor ln(max\ x (exp\ 1)) \rfloor + 1 = int\ k) = (of_bool(x \geq exp(real\ k - 1)) - of_bool(x \geq exp\ k)) :: real$

(**is** $?L1 = ?R1$) **if** $k \geq 3$ **for** $k\ x$

proof -

have $a1$: $real\ k - 1 \leq k$ **by** *simp*
have $?L1 = of_bool(\lfloor ln(max\ x (exp\ 1)) \rfloor = int\ k - 1)$ **by** *simp*

also have ... = *of-bool*($\ln(\max x (exp\ 1)) \in \{real\ k-1..<real\ k\}$) **unfolding** *floor-eq-iff* **by** *simp*
also have ... = *of-bool*($\exp(\ln(\max x (exp\ 1))) \in \{exp\ (real\ k-1)..<exp\ (real\ k)\}$) **by** *simp*
also have ... = *of-bool*($\max x (exp\ 1) \in \{exp\ (real\ k-1)..<exp\ (real\ k)\}$)
by (*subst exp-ln*) (*auto intro!:max.strict-coboundedI2*)
also have ... = *of-bool*($x \in \{exp\ (real\ k-1)..<exp\ (real\ k)\}$)
proof (*cases x ≥ exp 1*)
case *True*
then show *?thesis* **by** *simp*
next
case *False*
have $\{exp\ (real\ k - 1)..<exp\ (real\ k)\} \subseteq \{exp\ (real\ k - 1)..\}$ **by** *auto*
also have ... $\subseteq \{exp\ 1..\}$ **using** *that* **by** *simp*
finally have $\{exp\ (real\ k - 1)..<exp\ (real\ k)\} \subseteq \{exp\ 1..\}$ **by** *simp*
moreover have $x \notin \{exp\ 1..\}$ **using** *False* **by** *simp*
ultimately have $x \notin \{exp\ (real\ k - 1)..<exp\ (real\ k)\}$ **by** *blast*
hence *of-bool*($x \in \{exp\ (real\ k-1)..<exp\ (real\ k)\}$) = 0 **by** *simp*
also have ... = *of-bool*($\max x (exp\ 1) \in \{exp\ (real\ k-1)..<exp\ (real\ k)\}$)
using *False that* **by** *simp*
finally show *?thesis* **by** *metis*
qed
also have ... = *?R1* **using** *order-trans[OF iffD2[OF exp-le-cancel-iff a1]]* **by** *auto*
finally show *?thesis* **by** *simp*
qed

have 0: $\{nat\ \lfloor \ln(\max (f\ x) (exp\ 1)) \rfloor + 1\} \subseteq \{2..k-max\}$ (**is** $\{?L1\} \subseteq ?R2$)
if $x \in$ *pro-set S* **for** x
proof (*cases f x ≥ exp 1*)
case *True*
hence $?L1 = nat\ \lfloor \ln(f\ x) \rfloor + 1$ **by** *simp*
also have ... $\leq (MAX\ v \in$ *pro-set S*. $nat\ \lfloor \ln(f\ v) \rfloor + 1)$
by (*intro Max-ge finite-imageI imageI that finite-pro-set*)
also have ... $\leq k-max$ **unfolding** *k-max-def* **by** *simp*
finally have *le-0*: $?L1 \leq k-max$ **by** *simp*
have ($1::nat$) $\leq nat\ \lfloor \ln(exp\ (1::real)) \rfloor$ **by** *simp*
also have ... $\leq nat\ \lfloor \ln(f\ x) \rfloor$
using *True order-less-le-trans[OF exp-gt-zero]*
by (*intro nat-mono floor-mono iffD2[OF ln-le-cancel-iff]*) *auto*
finally have $1 \leq nat\ \lfloor \ln(f\ x) \rfloor$ **by** *simp*
hence $?L1 \geq 2$ **using** *True* **by** *simp*
hence $?L1 \in ?R2$ **using** *le-0* **by** *simp*
then show *?thesis* **by** *simp*
next
case *False*
hence $\{?L1\} = \{2\}$ **by** *simp*
also have ... $\subseteq ?R2$ **using** *k-max-ge-3* **by** *simp*
finally show *?thesis* **by** *simp*
qed

have 2: $(\sum i < l. f\ (w\ i)) \leq ?a + b * (\sum k = 3..<k-max. exp\ k * card\ \{i \in \{..<l\}. f\ (w\ i) \geq exp\ k\})$
(is $?L1 \leq ?R1$) **if** $w \in$ *pro-set (E l Λ S)* **for** w
proof –
have *s-w*: $w\ i \in$ *pro-set S* **for** i
using *that expander-pro-range[OF assms(1)] assms(2)*
unfolding *set-sample-pro[where S=E l Λ S]* **by** *auto*

have $?L1 \leq (\sum i < l. exp(\ln(\max(f\ (w\ i)) (exp\ 1))))$
by (*intro sum-mono*) (*simp add:less-max-iff-disj*)
also have ... $\leq (\sum i < l. exp(of-nat(nat\ \lfloor \ln(\max(f\ (w\ i)) (exp\ 1)) \rfloor + 1)))$

by (*intro sum-mono iffD2[OF exp-le-cancel-iff]*) *linarith*
also have ... = $(\sum i < l. (\sum k = 2..k\text{-max}. \text{exp } k * \text{of-bool } (k = \text{nat } \lfloor \ln (\text{max } (f (w i)) (\text{exp } 1)) \rfloor + 1)))$
using *Int-absorb1[OF 0] s-w by (intro sum.cong map-cong refl)*
(simp add:of-bool-def if-distrib if-distribR sum.If-cases)
also have ... =
 $(\sum i < l. (\sum k \in (\text{insert } 2 \{3..k\text{-max}\}). \text{exp } k * \text{of-bool } (k = \text{nat } \lfloor \ln (\text{max } (f (w i)) (\text{exp } 1)) \rfloor + 1)))$
using *k-max-ge-3 by (intro-cong [\sigma_1 sum-list] more:map-cong sum.cong) auto*
also have ... = $(\sum i < l. \text{exp } 2 * \text{of-bool } (2 = \text{nat } \lfloor \ln (\text{max } (f (w i)) (\text{exp } 1)) \rfloor + 1) +$
 $(\sum k = 3..k\text{-max}. \text{exp } k * \text{of-bool } (k = \text{nat } \lfloor \ln (\text{max } (f (w i)) (\text{exp } 1)) \rfloor + 1)))$
by (*subst sum.insert*) *auto*
also have ... $\leq (\sum i < l. \text{exp } 2 * 1 + (\sum k = 3..k\text{-max}. \text{exp } k * \text{of-bool } (k = \text{nat } \lfloor \ln (\text{max } (f (w i)) (\text{exp } 1)) \rfloor + 1)))$
by (*intro sum-mono add-mono mult-left-mono*) *auto*
also have ... = $(\sum i < l. \text{exp } 2 + (\sum k = 3..k\text{-max}. \text{exp } k * \text{of-bool } (\lfloor \ln (\text{max } (f (w i)) (\text{exp } 1)) \rfloor + 1 = \text{int } k)))$
by (*intro-cong [\sigma_1 sum-list, \sigma_1 of-bool, \sigma_2(+), \sigma_2(*)] more:map-cong sum.cong*) *auto*
also have ... =
 $(\sum i < l. \text{exp } 2 + (\sum k = 3..k\text{-max}. \text{exp } k * (\text{of-bool } (f (w i) \geq \text{exp } (\text{real } k - 1)) - \text{of-bool } (f (w i) \geq \text{exp } k))))$
by (*intro-cong [\sigma_1 sum-list, \sigma_1 of-bool, \sigma_2(+), \sigma_2(*)] more:map-cong sum.cong 1*) *auto*
also have ... = $(\sum i < l. \text{exp } 2 + (\sum k = 2 + 1..<k\text{-max} + 1. \text{exp } k * (\text{of-bool } (f (w i) \geq \text{exp } (\text{real } k - 1)) - \text{of-bool } (f (w i) \geq \text{exp } k))))$
by (*intro-cong [\sigma_2(+)] more:map-cong sum.cong*) *auto*
also have ... = $(\sum i < l. \text{exp } 2 + (\sum k = 2..<k\text{-max}. \text{exp } (k + 1) * (\text{of-bool } (f (w i) \geq \text{exp } k) - \text{of-bool } (f (w i) \geq \text{exp } (\text{Suc } k))))$
by (*subst sum.shift-bounds-nat-ivl*) *simp*
also have ... = $(\sum i < l. \text{exp } 2 + (\sum k = 2..<k\text{-max}. \text{exp } (k + 1) * \text{of-bool } (f (w i) \geq \text{exp } k) -$
 $(\sum k = 2..<k\text{-max}. \text{exp } (k + 1) * \text{of-bool } (f (w i) \geq \text{exp } (k + 1))))$
by (*simp add:sum-subtractf algebra-simps*)
also have ... = $(\sum i < l. \text{exp } 2 + (\sum k = 2..<k\text{-max}. \text{exp } (k + 1) * \text{of-bool } (f (w i) \geq \text{exp } k) -$
 $(\sum k = 3..<k\text{-max} + 1. \text{exp } k * \text{of-bool } (f (w i) \geq \text{exp } k))))$
by (*subst sum.shift-bounds-nat-ivl[symmetric]*) (*simp cong:sum.cong*)
also have ... = $(\sum i < l. \text{exp } 2 + (\sum k \in \text{insert } 2 \{3..<k\text{-max}\}. \text{exp } (k + 1) * \text{of-bool } (f (w i) \geq \text{exp } k) -$
 $(\sum k = 3..<k\text{-max} + 1. \text{exp } k * \text{of-bool } (f (w i) \geq \text{exp } k))))$
using *k-max-ge-3 by (intro-cong [\sigma_2 (+), \sigma_2 (-)] more:map-cong sum.cong) auto*
also have ... = $(\sum i < l. \text{exp } 2 + \text{exp } 3 * \text{of-bool } (f (w i) \geq \text{exp } 2) +$
 $(\sum k = 3..<k\text{-max}. \text{exp } (k + 1) * \text{of-bool } (f (w i) \geq \text{exp } k)) -$
 $(\sum k = 3..<k\text{-max} + 1. \text{exp } k * \text{of-bool } (f (w i) \geq \text{exp } k)))$
by (*subst sum.insert*) (*simp-all add:algebra-simps*)
also have ... $\leq (\sum i < l. \text{exp } 2 + \text{exp } 3 + (\sum k = 3..<k\text{-max}. \text{exp } (k + 1) * \text{of-bool } (f (w i) \geq \text{exp } k) -$
 $(\sum k = 3..<k\text{-max} + 1. \text{exp } k * \text{of-bool } (f (w i) \geq \text{exp } k))))$
by (*intro sum-mono add-mono diff-mono*) *auto*
also have ... = $(\sum i < l. \text{exp } 2 + \text{exp } 3 + (\sum k = 3..<k\text{-max}. \text{exp } (k + 1) * \text{of-bool } (f (w i) \geq \text{exp } k) -$
 $(\sum k \in \text{insert } k\text{-max } \{3..<k\text{-max}\}. \text{exp } k * \text{of-bool } (f (w i) \geq \text{exp } k))))$
using *k-max-ge-3 by (intro-cong [\sigma_2 (+), \sigma_2 (-)] more:map-cong sum.cong) auto*
also have ... = $(\sum i < l. \text{exp } 2 + \text{exp } 3 + (\sum k = 3..<k\text{-max}. (\text{exp } (k + 1) - \text{exp } k) * \text{of-bool } (f (w i) \geq \text{exp } k) -$
 $(\text{exp } k\text{-max} * \text{of-bool } (f (w i) \geq \text{exp } k\text{-max}))))$
by (*subst sum.insert*) (*auto simp add:sum-subtractf algebra-simps*)
also have ... $\leq (\sum i < l. \text{exp } 2 + \text{exp } 3 + (\sum k = 3..<k\text{-max}. (\text{exp } (k + 1) - \text{exp } k) * \text{of-bool } (f (w i) \geq \text{exp } k) - 0)$
by (*intro sum-mono add-mono diff-mono*) *auto*
also have ... $\leq (\sum i < l. \text{exp } 2 + \text{exp } 3 + (\sum k = 3..<k\text{-max}. (\text{exp } (k + 1) - \text{exp } k) * \text{of-bool } (f (w i) \geq \text{exp } k)))$
by *auto*

also have ... = $(\sum i < l. \exp 2 + \exp 3 + (\sum k = 3 .. < k\text{-max}. (\exp 1 - 1) * (\exp k * \text{of-bool}(f (w i) \geq \exp k))))$

by (simp add: exp-add algebra-simps)

also have ... = $(\sum i < l. \exp 2 + \exp 3 + b * (\sum k = 3 .. < k\text{-max}. \exp k * \text{of-bool}(f (w i) \geq \exp k)))$

unfolding b-def by (subst sum-distrib-left) simp

also have ... = $?a + b * (\sum i < l. (\sum k = 3 .. < k\text{-max}. \exp k * \text{of-bool}(f (w i) \geq \exp k)))$

by (simp add: sum-distrib-left[symmetric])

also have ... = ?R1

by (subst sum.swap) (simp add: ac-simps Int-def)

finally show ?thesis by simp

qed

have 3: $\exists k \in \{3 .. < k\text{-max}\}. g k \geq l / \text{real } k^2$ if $(\sum k = 3 .. < k\text{-max}. g k) \geq \text{real } l$ for g
proof (rule ccontr)

assume a3: $\neg(\exists k \in \{3 .. < k\text{-max}\}. g k \geq l / \text{real } k^2)$

hence $g k < l / \text{real } k^2$ if $k \in \{3 .. < k\text{-max}\}$ for k using that by force

hence $(\sum k = 3 .. < k\text{-max}. g k) < (\sum k = 3 .. < k\text{-max}. l / \text{real } k^2)$

using k-max-ge-4 by (intro sum-strict-mono) auto

also have ... $\leq (\sum k = 3 .. < k\text{-max}. l / (\text{real } k * (\text{real } k - 1)))$

by (intro sum-mono divide-left-mono) (auto simp: power2-eq-square)

also have ... = $l * (\sum k = 3 .. < k\text{-max}. 1 / (\text{real } k - 1) - 1/k)$

by (simp add: sum-distrib-left field-simps)

also have ... = $l * (\sum k = 2 + 1 .. < (k\text{-max} - 1) + 1. (-1)/k - (-1) / (\text{real } k - 1))$

by (intro sum.cong arg-cong2[where f=(*)]) auto

also have ... = $l * (\sum k = 2 .. < (k\text{-max} - 1). (-1)/(Suc k) - (-1) / k)$

by (subst sum.shift-bounds-nat-ivl) auto

also have ... = $l * (1/2 - 1 / \text{real } (k\text{-max} - 1))$

using k-max-ge-3 by (subst sum-Suc-diff') auto

also have ... $\leq \text{real } l * (1 - 0)$ by (intro mult-left-mono diff-mono) auto

also have ... = l by simp

finally have $(\sum k = 3 .. < k\text{-max}. g k) < l$ by simp

thus False using that by simp

qed

have 4: $L k \leq \exp(-\text{real } l - k + 2)$ if $k \geq 3$ for k

proof (cases $k \leq \ln l$)

case True

define γ where $\gamma = 1 / (\text{real } k)^2 / \exp(\text{real } k)$

define μ where $\mu = \exp(-\exp(\text{real } k) * \text{real } k^3)$

have exp-k-ubound: $\exp(\text{real } k) \leq \text{real } l$ using True assms(1) by (simp add: ln-ge-iff)

have $20 \leq \exp(3::\text{real})$ by (approximation 10)

also have ... $\leq \exp(\text{real } k)$ using that by simp

finally have exp-k-lbound: $20 \leq \exp(\text{real } k)$ by simp

have measure (sample-pro S) $\{v. f v \geq \exp(\text{real } k)\} \leq \exp(-\exp(\text{real } k) * \ln(\exp(\text{real } k)) ^ 3)$

by (intro assms(3) exp-k-lbound)

also have ... = $\exp(-(\exp(\text{real } k) * \text{real } k^3))$ by simp

finally have μ -bound: measure (sample-pro S) $\{v. f v \geq \exp(\text{real } k)\} \leq \mu$ by (simp add: μ -def)

have $\mu + \Lambda \leq \exp(-\exp(\text{real } k) * \text{real } k^3) + \exp(-\text{real } l * \ln(\text{real } l) ^ 3)$

unfolding μ -def using assms by (intro add-mono) auto

also have ... = $\exp(-(\exp(\text{real } k) * \text{real } k^3)) + \exp(-(\text{real } l * \ln(\text{real } l) ^ 3))$ by simp

also have ... $\leq \exp(-(\exp(\text{real } k) * \text{real } k^3)) + \exp(-(\exp(\text{real } k) * \ln(\exp(\text{real } k)) ^ 3))$

using assms(1) exp-k-ubound by (intro add-mono iffD2[OF exp-le-cancel-iff] le-imp-neg-le mult-mono power-mono iffD2[OF ln-le-cancel-iff]) simp-all

also have ... = $2 * \exp(-\exp(\text{real } k) * \text{real } k^3)$ by simp

finally have $\mu+\Lambda$ -bound: $\mu+\Lambda \leq 2 * \exp(-\exp(\text{real } k) * \text{real } k^3)$ by *simp*

have $\mu+\Lambda \leq 2 * \exp(-\exp(\text{real } k) * \text{real } k^3)$ by (intro $\mu+\Lambda$ -bound)

also have $\dots = \exp(-\exp(\text{real } k) * \text{real } k^3 + \ln 2)$ **unfolding** *exp-add* by *simp*

also have $\dots = \exp(-(\exp(\text{real } k) * \text{real } k^3 - \ln 2))$ by *simp*

also have $\dots \leq \exp(-((1 + \text{real } k) * \text{real } k^3 - \ln 2))$

using that by (intro *iffD2[OF exp-le-cancel-iff]* *le-imp-neg-le* *diff-mono* *mult-right-mono* *exp-ge-add-one-self-aux*) *auto*

also have $\dots = \exp(-(\text{real } k^4 + (\text{real } k^3 - \ln 2)))$

by (*simp add:power4-eq-xxxx power3-eq-cube algebra-simps*)

also have $\dots \leq \exp(-(\text{real } k^4 + (2^3 - \ln 2)))$ **using that**

by (intro *iffD2[OF exp-le-cancel-iff]* *le-imp-neg-le* *add-mono* *diff-mono* *power-mono*) *auto*

also have $\dots \leq \exp(-(\text{real } k^4 + 0))$

by (intro *iffD2[OF exp-le-cancel-iff]* *le-imp-neg-le* *add-mono* *order.refl*) (*approximation 5*)

also have $\dots \leq \exp(-(\text{real } k^3 * \text{real } k))$

by (*simp add:power4-eq-xxxx power3-eq-cube algebra-simps*)

also have $\dots \leq \exp(-2^3 * \text{real } k)$ **using that**

by (intro *iffD2[OF exp-le-cancel-iff]* *le-imp-neg-le* *mult-right-mono* *power-mono*) *auto*

also have $\dots \leq \exp(-3 * \text{real } k)$ by (intro *iffD2[OF exp-le-cancel-iff]*) *auto*

also have $\dots = \exp(-(\text{real } k + 2 * \text{real } k))$ by *simp*

also have $\dots \leq \exp(-(\text{real } k + 2 * \ln k))$

using that

by (intro *iffD2[OF exp-le-cancel-iff]* *le-imp-neg-le* *add-mono* *mult-left-mono* *ln-bound*) *auto*

also have $\dots = \exp(-(\text{real } k + \ln(k^2)))$ **using that** by (*subst ln-pow[symmetric]*) *auto*

also have $\dots = \gamma$

using that **unfolding** γ -def *exp-minus* *exp-add* *inverse-eq-divide* by (*simp add:algebra-simps*)

finally have $\mu+\Lambda$ -le- γ : $\mu+\Lambda \leq \gamma$ by *simp*

have $\mu \geq 0$ **unfolding** μ -def by *simp*

hence $\mu+\Lambda$ -gt-0: $\mu+\Lambda > 0$ **using** *assms(2)* by *auto*

have $\gamma = 1 / ((\text{real } k)^2 * \exp(\text{real } k))$ **unfolding** γ -def by *simp*

also have $\dots \leq 1 / (2^2 * \exp 2)$

using that by (intro *divide-left-mono* *mult-mono* *power-mono*) (*auto*)

finally have γ -ubound: $\gamma \leq 1 / (4 * \exp 2)$ by *simp*

have $\gamma \leq 1 / (4 * \exp 2)$ by (intro γ -ubound)

also have $\dots < 1$ by (*approximation 5*)

finally have γ -lt-1: $\gamma < 1$ by *simp*

have γ -ge-0: $\gamma \geq 0$ **using that** **unfolding** γ -def by (intro *divide-nonneg-pos*) *auto*

have μ -le-1: $\mu \leq 1$ **unfolding** μ -def by *simp*

have L $k = \text{measure } ?w \{w. \gamma * l \leq \text{real } (\text{card } \{i \in \{..<l\}. \exp(\text{real } k) \leq f(w\ i)\})\}$

unfolding L -def γ -def **using that**

by (intro-*cong* [σ_2 *measure*] *more:Collect-cong*) (*simp add:field-simps*)

also have $\dots \leq \exp(-\text{real } l * (\gamma * \ln(1/(\mu+\Lambda)) - 2 * \exp(-1)))$

using γ -lt-1 *assms(2)* by (intro *walk-tail-bound* μ -bound *assms(1)* $\mu+\Lambda$ -le- γ μ -le-1) *auto*

also have $\dots = \exp(\text{real } l * (\gamma * \ln(\mu+\Lambda) + 2 * \exp(-1)))$

using $\mu+\Lambda$ -gt-0 by (*simp-all add:ln-div algebra-simps*)

also have $\dots \leq \exp(\text{real } l * (\gamma * \ln(2 * \exp(-\exp(\text{real } k) * \text{real } k^3)) + 2 * \exp(-1)))$

using $\mu+\Lambda$ -gt-0 $\mu+\Lambda$ -bound γ -ge-0

by (intro *iffD2[OF exp-le-cancel-iff]* *mult-left-mono* *add-mono* *iffD2[OF ln-le-cancel-iff]*) *simp-all*

also have $\dots = \exp(\text{real } l * (\gamma * (\ln 2 - \exp(\text{real } k) * \text{real } k^3) + 2 * \exp(-1)))$

by (*simp add:ln-mult*)

also have $\dots = \exp(\text{real } l * (\gamma * \ln 2 - \text{real } k + 2 * \exp(-1)))$

using that **unfolding** γ -def by (*simp add:field-simps power2-eq-square power3-eq-cube*)

also have ... $\leq \exp(\text{real } l * (\ln 2 / (4 * \exp 2) - \text{real } k + 2 * \exp(-1)))$
using γ -ubound **by** (intro iffD2[OF exp-le-cancel-iff] mult-left-mono add-mono diff-mono)
(auto simp: divide-simps)
also have ... $= \exp(\text{real } l * (\ln 2 / (4 * \exp 2) + 2 * \exp(-1) - \text{real } k))$
by simp
also have ... $\leq \exp(\text{real } l * (1 - \text{real } k))$
by (intro iffD2[OF exp-le-cancel-iff] mult-left-mono diff-mono order.refl of-nat-0-le-iff)
(approximation 12)
also have ... $\leq \exp(-\text{real } l - \text{real } k + 2)$
proof (intro iffD2[OF exp-le-cancel-iff])
have $1 * (\text{real } k - 2) \leq \text{real } l * (\text{real } k - 2)$
using assms(1) **that by** (intro mult-right-mono) auto
thus $\text{real } l * (1 - \text{real } k) \leq -\text{real } l - \text{real } k + 2$ **by** argo
qed
finally show ?thesis **by** simp
next
case False
hence k-gt-l: $k \geq \ln l$ **by** simp
define γ **where** $\gamma = 1 / (\text{real } k)^2 / \exp(\text{real } k)$

have $20 \leq \exp(3::\text{real})$ **by** (approximation 10)
also have ... $\leq \exp(\text{real } k)$ **using** that **by** simp
finally have exp-k-lbound: $20 \leq \exp(\text{real } k)$ **by** simp

have γ -gt-0: $0 < \gamma$ **using** that **unfolding** γ -def **by** (intro divide-pos-pos) auto

hence γ -l-gt-0: $0 < \gamma * \text{real } l$ **using** assms(1) **by** auto

have $L k = \text{measure } ?w \{w. \gamma * l \leq \text{real } (\text{card } \{i \in \{..<l\}. \exp(\text{real } k) \leq f(w i)\})\}$
unfolding L-def γ -def **using** that
by (intro-cong [σ_2 measure] more:Collect-cong) (simp add:field-simps)
also have ... $\leq (\int w. \text{real } (\text{card } \{i \in \{..<l\}. \exp(\text{real } k) \leq f(w i)\}) \partial ?w) / (\gamma * l)$
by (intro pmf-markov γ -l-gt-0) simp-all
also have ... $= (\int w. (\sum i < l. \text{of-bool } (\exp(\text{real } k) \leq f(w i))) \partial ?w) / (\gamma * l)$
by (intro-cong [σ_2 (/)] more:integral-cong-AE AE-pmfI) (auto simp add:Int-def)
also have ... $= (\sum i < l. (\int w. \text{of-bool } (\exp(\text{real } k) \leq f(w i)) \partial ?w)) / (\gamma * l)$
by (intro-cong [σ_2 (/)] more:integral-sum integrable-measure-pmf-finite finite-pro-set)
also have ... $= (\sum i < l. (\int v. \text{of-bool } (\exp(\text{real } k) \leq f v) \partial (\text{map-pmf } (\lambda w. w i) ?w))) / (\gamma * l)$
by simp
also have ... $= (\sum i < l. (\int v. \text{of-bool } (\exp(\text{real } k) \leq f v) \partial ?p)) / (\gamma * l)$ **using** assms(1,2)
by (intro-cong [σ_2 (/), σ_2 (integral^L), σ_1 measure-pmf] more:sum.cong expander-uniform-property)
simp-all
also have ... $= (\sum i < l. (\int v. \text{indicat-real } \{v. (\exp(\text{real } k) \leq f v)\} v \partial ?p)) / (\gamma * l)$
by (intro-cong [σ_2 (/), σ_2 (integral^L)] more:sum.cong) auto
also have ... $= (\sum i < l. (\text{measure } ?p \{v. f v \geq \exp(\text{real } k)\})) / (\gamma * l)$ **by** simp
also have ... $\leq (\sum i < l. \exp(-\exp(\text{real } k) * \ln(\exp(\text{real } k))^{\wedge} 3)) / (\gamma * l)$
using γ -l-gt-0 **by** (intro divide-right-mono sum-mono assms(3) exp-k-lbound) auto
also have ... $= \exp(-\exp(\text{real } k) * \text{real } k^{\wedge} 3) / \gamma$ **using** assms(1) **by** simp
also have ... $= \exp(\text{real } k + \ln(k^{\wedge} 2) - \exp(\text{real } k) * \text{real } k^{\wedge} 3)$
using that **unfolding** γ -def
by (simp add:exp-add exp-diff exp-minus algebra-simps inverse-eq-divide)
also have ... $= \exp(\text{real } k + 2 * \ln k - \exp(\text{real } k) * \text{real } k^{\wedge} 3)$
using that **by** (subst ln-powr[symmetric]) auto
also have ... $\leq \exp(\text{real } k + 2 * \text{real } k - \exp(\ln l) * \text{real } k^{\wedge} 3)$
using that k-gt-l ln-bound
by (intro iffD2[OF exp-le-cancel-iff] add-mono diff-mono mult-left-mono mult-right-mono)
auto
also have ... $= \exp(3 * \text{real } k - l * (\text{real } k^{\wedge} 3 - 1) - l)$

using *assms(1)* **by** (*subst exp-ln*) (*auto simp add:algebra-simps*)
also have $\dots \leq \exp(3 * \text{real } k - 1 * (\text{real } k^3 - 1) - l)$
using *assms(1)* **that by** (*intro iffD2[OF exp-le-cancel-iff]*) *diff-mono mult-right-mono*) *auto*
also have $\dots = \exp(3 * \text{real } k - \text{real } k * \text{real } k^2 - 1 - l + 2)$
by (*simp add:power2-eq-square power3-eq-cube*)
also have $\dots \leq \exp(3 * \text{real } k - \text{real } k * 2^2 - 0 - l + 2)$
using *assms(1)* **that**
by (*intro iffD2[OF exp-le-cancel-iff]*) *add-mono diff-mono mult-left-mono power-mono*) *auto*
also have $\dots = \exp(-\text{real } l - \text{real } k + 2)$ **by** *simp*
finally show *?thesis* **by** *simp*
qed

have $?L \leq \text{measure } ?w$
 $\{w. ?a + b * (\sum k=3..<k\text{-max. } \exp(\text{real } k) * \text{card } \{i \in \{..<l\}. f(w\ i) \geq \exp(\text{real } k)\}) \geq C_1 * l\}$
using *order-trans[OF - 2]* **by** (*intro pmf-mono*) *simp*
also have $\dots = \text{measure } ?w$
 $\{w. (\sum k=3..<k\text{-max. } \exp(\text{real } k) * \text{card } \{i \in \{..<l\}. f(w\ i) \geq \exp(\text{real } k)\}) \geq l\}$
unfolding *C1-def b-def[symmetric]* **using** *b-gt-0*
by (*intro-cong [\sigma_2 measure]*) *more:Collect-cong*) (*simp add:algebra-simps*)
also have $\dots \leq \text{measure } ?w$
 $\{w. (\exists k \in \{3..<k\text{-max}\}. \exp(\text{real } k) * \text{card } \{i \in \{..<l\}. f(w\ i) \geq \exp(\text{real } k)\}) \geq \text{real } l / \text{real } k^2\}$
using *3* **by** (*intro pmf-mono*) *simp*
also have $\dots = \text{measure } ?w$
 $(\bigcup k \in \{3..<k\text{-max}\}. \{w. \exp(\text{real } k) * \text{card } \{i \in \{..<l\}. f(w\ i) \geq \exp(\text{real } k)\} \geq \text{real } l / \text{real } k^2\})$
by (*intro-cong [\sigma_2 measure]*) *auto*
also have $\dots \leq (\sum k=3..<k\text{-max. } L\ k)$
unfolding *L-def* **by** (*intro finite-measure.finite-measure-subadditive-finite*) *auto*
also have $\dots \leq (\sum k=3..<k\text{-max. } \exp(-\text{real } l - \text{real } k + 2))$ **by** (*intro sum-mono 4*) *auto*
also have $\dots = (\sum k=0+3..<(k\text{-max}-3)+3. \exp(-\text{real } l - \text{real } k + 2))$
using *k-max-ge-3* **by** (*intro sum.cong*) *auto*
also have $\dots = (\sum k=0..<k\text{-max}-3. \exp(-1 - \text{real } l - \text{real } k))$
by (*subst sum.shift-bounds-nat-ivl*) (*simp add:algebra-simps*)
also have $\dots = \exp(-1 - \text{real } l) * (\sum k < k\text{-max}-3. \exp(\text{real } k * (-1)))$
using *atLeast0LessThan*
by (*simp add:exp-diff exp-add sum-distrib-left exp-minus inverse-eq-divide*)
also have $\dots = \exp(-1 - \text{real } l) * ((\exp(-1) ^ (k\text{-max} - 3) - 1) / (\exp(-1) - 1))$
unfolding *exp-of-nat-mult* **by** (*subst geometric-sum*) *auto*
also have $\dots = \exp(-1 - \text{real } l) * (1 - \exp(-1) ^ (k\text{-max} - 3)) / (1 - \exp(-1))$
by (*simp add:field-simps*)
also have $\dots \leq \exp(-1 - \text{real } l) * (1 - 0) / (1 - \exp(-1))$
using *k-max-ge-3* **by** (*intro mult-left-mono divide-right-mono diff-mono*) *auto*
also have $\dots = \exp(-\text{real } l) * (\exp(-1) / (1 - \exp(-1)))$
by (*simp add:exp-diff exp-minus inverse-eq-divide*)
also have $\dots \leq \exp(-\text{real } l) * 1$
by (*intro mult-left-mono exp-ge-zero*) (*approximation 10*)
finally show *?thesis* **by** *simp*
qed

unbundle *no-intro-cong-syntax*

end

6 Inner Algorithm

This section introduces the inner algorithm (as mentioned it is already a solution to the cardinality estimation with the caveat that, if ε is too small it requires too much space. The outer algorithm in Section 10 resolved this problem.

The algorithm makes use of the balls and bins model, more precisely, the fact that the number of hit bins can be used to estimate the number of balls thrown (even if there are collisions). I.e. it assigns each universe element to a bin using a k -wise independent hash function. Then it counts the number of bins hit.

This strategy however would only work if the number of balls is roughly equal to the number of bins, to remedy that the algorithm performs an adaptive sub-sampling strategy. This works by assigning each universe element a level (using a second hash function) with a geometric distribution. The algorithm then selects a level that is appropriate based on a rough estimate obtained using the maximum level in the bins.

To save space the algorithm drops information about small levels, whenever the space usage would be too high otherwise. This level will be called the cutoff-level. This is okay as long as the cutoff level is not larger than the sub-sampling threshold. A lot of the complexity in the proof is devoted to verifying that the cutoff-level will not cross it, it works by defining a third value s_M that is both an upper bound for the cutoff level and a lower bound for the subsampling threshold simultaneously with high probability.

theory *Distributed-Distinct-Elements-Inner-Algorithm*

imports

Universal-Hash-Families.Pseudorandom-Objects-Hash-Families
Distributed-Distinct-Elements-Preliminary
Distributed-Distinct-Elements-Balls-and-Bins
Distributed-Distinct-Elements-Tail-Bounds
Prefix-Free-Code-Combinators.Prefix-Free-Code-Combinators

begin

unbundle *intro-cong-syntax*

hide-const *Abstract-Rewriting.restrict*

definition $C_4 :: \text{real}$ **where** $C_4 = 3^2 * 2^3$

definition $C_5 :: \text{int}$ **where** $C_5 = 3^3$

definition $C_6 :: \text{real}$ **where** $C_6 = 4$

definition $C_7 :: \text{nat}$ **where** $C_7 = 2^5$

locale *inner-algorithm* =

fixes $n :: \text{nat}$

fixes $\delta :: \text{real}$

fixes $\varepsilon :: \text{real}$

assumes *n-gt-0*: $n > 0$

assumes *delta-gt-0*: $\delta > 0$ **and** *delta-lt-1*: $\delta < 1$

assumes *epsilon-gt-0*: $\varepsilon > 0$ **and** *epsilon-lt-1*: $\varepsilon < 1$

begin

definition *b-exp* **where** $b\text{-exp} = \text{nat } \lceil \log 2 (C_4 / \varepsilon^2) \rceil$

definition $b :: \text{nat}$ **where** $b = 2^{b\text{-exp}}$

definition l **where** $l = \text{nat } \lceil C_6 * \ln (2 / \delta) \rceil$

definition k **where** $k = \text{nat } \lceil C_2 * \ln b + C_3 \rceil$

definition $\Lambda :: \text{real}$ **where** $\Lambda = \min (1/16) (\exp (-l * \ln l^3))$

definition $\rho :: \text{real} \Rightarrow \text{real}$ **where** $\rho x = b * (1 - (1-1/b)^x)$

definition $\rho\text{-inv} :: \text{real} \Rightarrow \text{real}$ **where** $\rho\text{-inv } x = \ln (1-x/b) / \ln (1-1/b)$

lemma *l-bound*: $C_6 * \ln (2 / \delta) \leq l$

unfolding *l-def* **by** *linarith*

lemma *k-min*: $C_2 * \ln (\text{real } b) + C_3 \leq \text{real } k$

unfolding *k-def* **by** *linarith*

lemma Λ -gt-0: $\Lambda > 0$
 unfolding Λ -def min-less-iff-conj by auto

lemma Λ -le-1: $\Lambda \leq 1$
 unfolding Λ -def by auto

lemma l-gt-0: $l > 0$

proof –
 have $0 < C_6 * \ln (2 / \delta)$
 unfolding C_6 -def using δ -gt-0 δ -lt-1
 by (intro Rings.mult-pos-pos ln-gt-zero) auto
 also have $\dots \leq l$
 by (intro l-lbound)
 finally show ?thesis
 by simp
 qed

lemma l-ubound: $l \leq C_6 * \ln(1 / \delta) + C_6 * \ln 2 + 1$

proof –
 have $l = \text{of-int } \lceil C_6 * \ln (2 / \delta) \rceil$
 using l-gt-0 unfolding l-def
 by (intro of-nat-nat) simp
 also have $\dots \leq C_6 * \ln (1 / \delta * 2) + 1$
 by simp
 also have $\dots = C_6 * \ln (1 / \delta) + C_6 * \ln 2 + 1$
 using δ -gt-0 δ -lt-1
 by (subst ln-mult) (auto simp add: algebra-simps)
 finally show ?thesis by simp
 qed

lemma b-exp-ge-26: $b\text{-exp} \geq 26$

proof –
 have $2 \text{ powr } 25 < C_4 / 1$ unfolding C_4 -def by simp
 also have $\dots \leq C_4 / \varepsilon^2$
 using ε -gt-0 ε -lt-1 unfolding C_4 -def
 by (intro divide-left-mono power-le-one) auto
 finally have $2 \text{ powr } 25 < C_4 / \varepsilon^2$ by simp
 hence $\log 2 (C_4 / \varepsilon^2) > 25$
 using ε -gt-0 unfolding C_4 -def
 by (intro iffD2[OF less-log-iff] divide-pos-pos zero-less-power) auto
 hence $\lceil \log 2 (C_4 / \varepsilon^2) \rceil \geq 26$ by simp
 thus ?thesis
 unfolding b-exp-def by linarith
 qed

lemma b-min: $b \geq 2^{26}$

unfolding b-def
 by (meson b-exp-ge-26 nat-power-less-imp-less not-less power-eq-0-iff power-zero-numeral)

lemma k-gt-0: $k > 0$

proof –
 have $(0::\text{real}) < 7.5 * 0 + 16$ by simp
 also have $\dots \leq 7.5 * \ln(\text{real } b) + 16$
 using b-min
 by (intro add-mono mult-left-mono ln-ge-zero) auto
 finally have $0 < \text{real } k$
 using k-min unfolding C_2 -def C_3 -def by simp
 thus ?thesis by simp

qed

lemma *b-ne*: $\{..<b\} \neq \{\}$

proof –

have $0 \in \{0..<b\}$

using *b-min* **by** *simp*

thus *?thesis*

by *auto*

qed

lemma *b-lower-bound*: $C_4 / \varepsilon^2 \leq \text{real } b$

proof –

have $C_4 / \varepsilon^2 = 2 \text{ powr } (\log 2 (C_4 / \varepsilon^2))$

using *ε-gt-0* **unfolding** *C₄-def* **by** (*intro powr-log-cancel[symmetric] divide-pos-pos*) *auto*

also have $\dots \leq 2 \text{ powr } (\text{nat } \lceil \log 2 (C_4 / \varepsilon^2) \rceil)$

by (*intro powr-mono of-nat-ceiling*) *simp*

also have $\dots = \text{real } b$

unfolding *b-def b-exp-def* **by** (*simp add:powr-realpow*)

finally show *?thesis* **by** *simp*

qed

definition *n-exp* **where** $n\text{-exp} = \max (\text{nat } \lceil \log 2 n \rceil) 1$

lemma *n-exp-gt-0*: $n\text{-exp} > 0$

unfolding *n-exp-def* **by** *simp*

abbreviation Ψ_1 **where** $\Psi_1 \equiv \mathcal{H} \ 2 \ n \ (\mathcal{G} \ n\text{-exp})$

abbreviation Ψ_2 **where** $\Psi_2 \equiv \mathcal{H} \ 2 \ n \ (\mathcal{N} \ (C_7 * b^2))$

abbreviation Ψ_3 **where** $\Psi_3 \equiv \mathcal{H} \ k \ (C_7 * b^2) \ (\mathcal{N} \ b)$

definition Ψ **where** $\Psi = \Psi_1 \times_P \Psi_2 \times_P \Psi_3$

abbreviation Ω **where** $\Omega \equiv \mathcal{E} \ l \ \wedge \ \Psi$

type-synonym *state* = $(\text{nat} \Rightarrow \text{nat} \Rightarrow \text{int}) \times (\text{nat})$

fun *is-too-large* :: $(\text{nat} \Rightarrow \text{nat} \Rightarrow \text{int}) \Rightarrow \text{bool}$ **where**

is-too-large $B = ((\sum (i,j) \in \{..<l\} \times \{..<b\}. \lfloor \log 2 (\max (B \ i \ j) (-1) + 2) \rfloor) > C_5 * b * l)$

fun *compress-step* :: *state* \Rightarrow *state* **where**

compress-step $(B,q) = (\lambda \ i \ j. \max (B \ i \ j - 1) (-1), q+1)$

function *compress* :: *state* \Rightarrow *state* **where**

compress $(B,q) = ($

if is-too-large B

then (*compress* (*compress-step* (B,q)))

else (B,q))

by *auto*

fun *compress-termination* :: *state* \Rightarrow *nat* **where**

compress-termination $(B,q) = (\sum (i,j) \in \{..<l\} \times \{..<b\}. \text{nat } (B \ i \ j + 1))$

lemma *compress-termination*:

assumes *is-too-large* B

shows *compress-termination* (*compress-step* (B,q)) $<$ *compress-termination* (B,q)

proof (*rule ccontr*)

let $?I = \{..<l\} \times \{..<b\}$

have $a: \text{nat } (\max (B \ i \ j - 1) (-1) + 1) \leq \text{nat } (B \ i \ j + 1)$ **for** $i \ j$

by simp
assume \neg *compress-termination* (*compress-step* (B, q)) < *compress-termination* (B, q)
hence $(\sum (i,j) \in ?I. \text{nat } (B \ i \ j + 1)) \leq (\sum (i,j) \in ?I. \text{nat } (\max (B \ i \ j - 1) (-1) + 1))$
by simp
moreover have $(\sum (i,j) \in ?I. \text{nat } (B \ i \ j + 1)) \geq (\sum (i,j) \in ?I. \text{nat } (\max (B \ i \ j - 1) (-1) + 1))$
by (*intro sum-mono*) *auto*
ultimately have b :
 $(\sum (i,j) \in ?I. \text{nat } (\max (B \ i \ j - 1) (-1) + 1)) = (\sum (i,j) \in ?I. \text{nat } (B \ i \ j + 1))$
using *order-antisym* **by** *simp*
have $\text{nat } (B \ i \ j + 1) = \text{nat } (\max (B \ i \ j - 1) (-1) + 1)$ **if** $(i,j) \in ?I$ **for** $i \ j$
using *sum-mono-inv[OF b]* **that a** **by** *auto*
hence $\max (B \ i \ j) (-1) = -1$ **if** $(i,j) \in ?I$ **for** $i \ j$
using *that* **by** *fastforce*
hence $(\sum (i,j) \in ?I. \lfloor \log 2 (\max (B \ i \ j) (-1) + 2) \rfloor) = (\sum (i,j) \in ?I. 0)$
by (*intro sum.cong, auto*)
also have $\dots = 0$ **by** *simp*
also have $\dots \leq C_5 * b * l$ **unfolding** *C₅-def* **by** *simp*
finally have \neg *is-too-large B* **by** *simp*
thus *False* **using** *assms* **by** *simp*
qed

termination *compress*
using *measure-def compress-termination*
by (*relation Wellfounded.measure (compress-termination), auto*)

fun *merge1* :: *state* \Rightarrow *state* \Rightarrow *state* **where**
merge1 ($B1, q_1$) ($B2, q_2$) = (
let $q = \max q_1 \ q_2$ *in* $(\lambda \ i \ j. \max (B1 \ i \ j + q_1 - q) (B2 \ i \ j + q_2 - q), q)$)

fun *merge* :: *state* \Rightarrow *state* \Rightarrow *state* **where**
merge $x \ y = \text{compress } (\text{merge1 } x \ y)$

type-synonym *seed* = *nat* \Rightarrow (*nat* \Rightarrow *nat*) \times (*nat* \Rightarrow *nat*) \times (*nat* \Rightarrow *nat*)

fun *single1* :: *seed* \Rightarrow *nat* \Rightarrow *state* **where**
single1 $\omega \ x = (\lambda \ i \ j.$
let $(f,g,h) = \omega \ i$ *in* (
if $h (g \ x) = j \wedge i < l$ *then* $\text{int } (f \ x)$ *else* (-1)), 0)

fun *single* :: *seed* \Rightarrow *nat* \Rightarrow *state* **where**
single $\omega \ x = \text{compress } (\text{single1 } \omega \ x)$

fun *estimate1* :: *state* \Rightarrow *nat* \Rightarrow *real* **where**
estimate1 (B, q) $i =$ (
let $s = \max 0 (\text{Max } (\{B \ i\} \text{ '}\{..\<b\}\} + q - \lfloor \log 2 \ b \rfloor + 9);$
 $p = \text{card } \{ j. j \in \{..\<b\} \wedge B \ i \ j + q \geq s \}$ *in*
 $2 \ \text{powl } s * \ln (1-p/b) / \ln(1-1/b)$)

fun *estimate* :: *state* \Rightarrow *real* **where**
estimate $x = \text{median } l (\text{estimate1 } x)$

6.1 History Independence

fun τ_0 :: (*nat* \Rightarrow *nat*) \times (*nat* \Rightarrow *nat*) \times (*nat* \Rightarrow *nat*) \Rightarrow *nat set* \Rightarrow *nat* \Rightarrow *int*
where $\tau_0 (f,g,h) \ A \ j = \text{Max } (\{ \text{int } (f \ a) \mid a . a \in A \wedge h (g \ a) = j \} \cup \{-1\})$

definition τ_1 :: (*nat* \Rightarrow *nat*) \times (*nat* \Rightarrow *nat*) \times (*nat* \Rightarrow *nat*) \Rightarrow *nat set* \Rightarrow *nat* \Rightarrow *nat* \Rightarrow *int*

where $\tau_1 \psi A q j = \max (\tau_0 \psi A j - q) (-1)$

definition $\tau_2 :: \text{seed} \Rightarrow \text{nat set} \Rightarrow \text{nat} \Rightarrow \text{nat} \Rightarrow \text{nat} \Rightarrow \text{int}$
where $\tau_2 \omega A q i j = (\text{if } i < l \text{ then } \tau_1 (\omega i) A q j \text{ else } (-1))$

definition $\tau_3 :: \text{seed} \Rightarrow \text{nat set} \Rightarrow \text{nat} \Rightarrow \text{state}$
where $\tau_3 \omega A q = (\tau_2 \omega A q, q)$

definition $q :: \text{seed} \Rightarrow \text{nat set} \Rightarrow \text{nat}$
where $q \omega A = (\text{LEAST } q . \neg(\text{is-too-large } (\tau_2 \omega A q)))$

definition $\tau :: \text{seed} \Rightarrow \text{nat set} \Rightarrow \text{state}$
where $\tau \omega A = \tau_3 \omega A (q \omega A)$

lemma $\tau_2\text{-step}$: $\tau_2 \omega A (x+y) = (\lambda i j. \max (\tau_2 \omega A x i j - y) (-1))$
by (intro ext) (auto simp add: $\tau_2\text{-def}$ $\tau_1\text{-def}$)

lemma $\tau_3\text{-step}$: $\text{compress-step } (\tau_3 \omega A x) = \tau_3 \omega A (x+1)$
unfolding $\tau_3\text{-def}$ using $\tau_2\text{-step}$ [where $y=1$] by simp

lemma Ψ_1 : $\text{is-prime-power (pro-size } (\mathcal{G} \ n\text{-exp}))$
unfolding geom-pro-size by (intro $\text{is-prime-powerI } n\text{-exp-gt-0}$) auto

lemma Ψ_2 : $\text{is-prime-power (pro-size } (\mathcal{N} (C_7 * b^{\wedge} 2)))$
proof –
have $0:\text{pro-size } (\mathcal{N} (C_7 * b^{\wedge} 2)) = 2^{\wedge} (5 + 2 * b\text{-exp})$
unfolding $C_7\text{-def}$ $b\text{-def}$ by (subst nat-pro-size) (auto simp add: power-add power-even-eq)
thus ?thesis unfolding 0 by (intro is-prime-powerI) auto
qed

lemma Ψ_3 : $\text{is-prime-power (pro-size } (\mathcal{N} \ b))$
proof –
have $0:\text{pro-size } (\mathcal{N} \ b) = 2^{\wedge} b\text{-exp}$ unfolding $b\text{-def}$ by (subst nat-pro-size) auto
thus ?thesis using $b\text{-exp-ge-26}$ unfolding 0 by (intro is-prime-powerI) auto
qed

lemma $\text{sample-pro-}\Psi$:
 $\text{sample-pro } \Psi = \text{pair-pmf (sample-pro } \Psi_1) (\text{pair-pmf (sample-pro } \Psi_2) (\text{sample-pro } \Psi_3))$
unfolding $\Psi\text{-def}$ by (simp add: prod-pro)

lemma $\text{sample-set-}\Psi$: $\text{pro-set } \Psi = \text{pro-set } \Psi_1 \times \text{pro-set } \Psi_2 \times \text{pro-set } \Psi_3$
unfolding $\Psi\text{-def}$ by (simp add: prod-pro-set)

lemma $f\text{-range}$:
assumes $(f,g,h) \in \text{pro-set } \Psi$
shows $f \ x \leq n\text{-exp}$
proof –
have $f \in \text{pro-set } \Psi_1$ using $\text{sample-set-}\Psi$ assms by auto
hence $f \in \text{pro-select } \Psi_1 \ \{..\text{<pro-size } \Psi_1\}$ unfolding set-sample-pro by auto
hence $f \ x \in \text{pro-set } (\mathcal{G} \ n\text{-exp})$ using $\text{hash-pro-range}[OF \ \Psi_1]$ by auto
thus ?thesis using geom-pro-range by auto
qed

lemma $g\text{-range-1}$:
assumes $g \in \text{pro-set } \Psi_2$
shows $g \ x < C_7 * b^{\wedge} 2$
proof –
have $g \in \text{pro-select } \Psi_2 \ \{..\text{<pro-size } \Psi_2\}$ using assms unfolding set-sample-pro by auto

hence $g x \in \text{pro-set } (\mathcal{N} (C_7 * b^{\wedge 2}))$ **using** $\text{hash-pro-range}[OF \Psi_2]$ **by** *auto*
moreover have $C_7 * b^{\wedge 2} > 0$ **unfolding** $C_7\text{-def } b\text{-def}$ **by** *simp*
ultimately show *?thesis* **using** nat-pro-set **by** *auto*
qed

lemma *h-range-1*:

assumes $h \in \text{pro-set } \Psi_3$
shows $h x < b$

proof –

have $h \in \text{pro-select } \Psi_3 \text{ ' } \{..<\text{pro-size } \Psi_3\}$ **using** *assms* **unfolding** *set-sample-pro* **by** *auto*
hence $h x \in \text{pro-set } (\mathcal{N} b)$ **using** $\text{hash-pro-range}[OF \Psi_3]$ **by** *auto*
moreover have $b > 0$ **unfolding** *b-def* **by** *simp*
ultimately show *?thesis* **using** nat-pro-set **by** *auto*

qed

lemma *g-range*:

assumes $(f,g,h) \in \text{pro-set } \Psi$
shows $g x < C_7 * b^{\wedge 2}$
using *g-range-1 sample-set-Ψ assms* **by** *simp*

lemma *h-range*:

assumes $(f,g,h) \in \text{pro-set } \Psi$
shows $h x < b$
using *h-range-1 sample-set-Ψ assms* **by** *simp*

lemma *fin-f*:

assumes $(f,g,h) \in \text{pro-set } \Psi$
shows $\text{finite } \{ \text{int } (f a) \mid a. P a \}$ **(is finite ?M)**

proof –

have $\text{finite } (\text{range } f)$
using $f\text{-range}[OF \text{assms}]$ *finite-nat-set-iff-bounded-le* **by** *auto*
hence $\text{finite } (\text{range } (\text{int } \circ f))$
by (*simp add:image-image[symmetric]*)
moreover have $?M \subseteq (\text{range } (\text{int } \circ f))$
using *image-mono* **by** (*auto simp add: setcompr-eq-image*)
ultimately show *?thesis*
using *finite-subset* **by** *auto*

qed

lemma *Max-int-range*: $x \leq (y::\text{int}) \implies \text{Max } \{x..y\} = y$
by *auto*

lemma Ω : $l > 0 \ \Lambda > 0$ **using** *l-gt-0 Λ-gt-0* **by** *auto*

lemma ω -range:

assumes $\omega \in \text{pro-set } \Omega$
shows $\omega i \in \text{pro-set } \Psi$

proof –

have $\omega \in \text{pro-select } \Omega \text{ ' } \{..<\text{pro-size } \Omega\}$ **using** *assms* **unfolding** *set-sample-pro* **by** *auto*
thus $\omega i \in \text{pro-set } \Psi$ **using** $\text{expander-pro-range}[OF \Omega]$ *assms* **by** *auto*

qed

lemma *max-q-1*:

assumes $\omega \in \text{pro-set } \Omega$
shows $\tau_2 \ \omega \ A \ (\text{nat } [\log 2 \ n]+2) \ i \ j = (-1)$

proof (*cases i < l*)

case *True*

obtain $f \ g \ h$ **where** $w\text{-}i: \omega i = (f,g,h)$ **by** (*metis prod-cases3*)

```

let ?max-q = max [log 2 (real n)] 1

have c: (f,g,h) ∈ pro-set Ψ using ω-range[OF assms] w-i[symmetric] by auto
have a:int (f x) ≤ ?max-q for x
proof -
  have int (f x) ≤ int n-exp
    using f-range[OF c] by auto
  also have ... = ?max-q unfolding n-exp-def by simp
  finally show ?thesis by simp
qed
have τ0 (ω i) A j ≤ Max {(-1)..?max-q}
  unfolding w-i τ0.simps using a by (intro Max-mono) auto
also have ... = ?max-q
  by (intro Max-int-range) auto
finally have τ0 (ω i) A j ≤ ?max-q by simp
hence max (τ0 (ω i) A j - int (nat [log 2 (real n)] + 2)) (- 1) = (-1)
  by (intro max-absorb2) linarith
thus ?thesis
  unfolding τ2-def τ1-def using True by auto
next
case False
thus ?thesis unfolding τ2-def τ1-def by simp
qed

lemma max-q-2:
  assumes ω ∈ pro-set Ω
  shows ¬ (is-too-large (τ2 ω A (nat [log 2 n]+2)))
  using max-q-1[OF assms] by (simp add:C5-def case-prod-beta mult-less-0-iff)

lemma max-s-3:
  assumes ω ∈ pro-set Ω
  shows q ω A ≤ (nat [log 2 n]+2)
  unfolding q-def by (intro wellorder-Least-lemma(2) max-q-2 assms)

lemma max-mono: x ≤ (y::'a::linorder) ⇒ max x z ≤ max y z
  using max.coboundedI1 by auto

lemma max-mono-2: y ≤ (z::'a::linorder) ⇒ max x y ≤ max x z
  using max.coboundedI2 by auto

lemma τ0-mono:
  assumes ψ ∈ pro-set Ψ
  assumes A ⊆ B
  shows τ0 ψ A j ≤ τ0 ψ B j
proof -
  obtain f g h where w-i: ψ = (f,g,h)
    by (metis prod-cases3)
  show ?thesis
    using assms fin-f unfolding τ0.simps w-i
    by (intro Max-mono) auto
qed

lemma τ2-mono:
  assumes ω ∈ pro-set Ω
  assumes A ⊆ B
  shows τ2 ω A x i j ≤ τ2 ω B x i j
proof -

```

have $\max (\tau_0 (\omega i) A j - \text{int } x) (-1) \leq \max (\tau_0 (\omega i) B j - \text{int } x) (-1)$ **if** $i < l$
using *that* $\omega\text{-range}[OF \text{ assms}(1)]$ **by** (*intro max-mono diff-mono τ_0 -mono assms(2) order.refl*)
thus *?thesis* **by** (*cases i < l*) (*auto simp add: τ_2 -def τ_1 -def*)
qed

lemma *is-too-large-antimono:*

assumes $\omega \in \text{pro-set } \Omega$
assumes $A \subseteq B$
assumes *is-too-large* ($\tau_2 \omega A x$)
shows *is-too-large* ($\tau_2 \omega B x$)

proof –

have $C_5 * b * l < (\sum (i,j) \in \{..<l\} \times \{..<b\}. \lfloor \log 2 (\max (\tau_2 \omega A x i j) (-1) + 2) \rfloor)$
using *assms(3)* **by** *simp*
also have $\dots = (\sum y \in \{..<l\} \times \{..<b\}. \lfloor \log 2 (\max (\tau_2 \omega A x (\text{fst } y) (\text{snd } y)) (-1) + 2) \rfloor)$
by (*simp add:case-prod-beta*)
also have $\dots \leq (\sum y \in \{..<l\} \times \{..<b\}. \lfloor \log 2 (\max (\tau_2 \omega B x (\text{fst } y) (\text{snd } y)) (-1) + 2) \rfloor)$
by (*intro sum-mono floor-mono iffD2[OF log-le-cancel-iff] iffD2[OF of-int-le-iff]*
add-mono max-mono τ_2 -mono[OF assms(1,2)]) *auto*
also have $\dots = (\sum (i,j) \in \{..<l\} \times \{..<b\}. \lfloor \log 2 (\max (\tau_2 \omega B x i j) (-1) + 2) \rfloor)$
by (*simp add:case-prod-beta*)
finally have $(\sum (i,j) \in \{..<l\} \times \{..<b\}. \lfloor \log 2 (\max (\tau_2 \omega B x i j) (-1) + 2) \rfloor) > C_5 * b * l$
by *simp*
thus *?thesis* **by** *simp*

qed

lemma *q-compact:*

assumes $\omega \in \text{pro-set } \Omega$
shows $\neg (\text{is-too-large } (\tau_2 \omega A (q \omega A)))$
unfolding *q-def* **using** *max-q-2[OF assms]*
by (*intro wellorder-Least-lemma(1)*) *blast*

lemma *q-mono:*

assumes $\omega \in \text{pro-set } \Omega$
assumes $A \subseteq B$
shows $q \omega A \leq q \omega B$

proof –

have $\neg (\text{is-too-large } (\tau_2 \omega A (q \omega B)))$
using *is-too-large-antimono[OF assms]* *q-compact[OF assms(1)]* **by** *blast*
hence (*LEAST* $q . \neg (\text{is-too-large } (\tau_2 \omega A q))$) $\leq q \omega B$
by (*intro Least-le*) *blast*
thus *?thesis*
by (*simp add:q-def*)

qed

lemma *lt-s-too-large:* $x < q \omega A \implies \text{is-too-large } (\tau_2 \omega A x)$

using *not-less-Least* **unfolding** *q-def* **by** *auto*

lemma *compress-result-1:*

assumes $\omega \in \text{pro-set } \Omega$
shows $\text{compress } (\tau_3 \omega A (q \omega A - i)) = \tau \omega A$

proof (*induction i*)

case *0*

then show *?case* **using** *q-compact[OF assms]* **by** (*simp add: τ_3 -def τ -def*)

next

case (*Suc i*)

show *?case*

proof (*cases i < q ωA*)

case *True*

have *is-too-large* $(\tau_2 \omega A (q \omega A - \text{Suc } i))$
using *True* **by** *(intro lt-s-too-large)* *simp*
hence *compress* $(\tau_3 \omega A (q \omega A - \text{Suc } i)) = \text{compress} (\text{compress-step} (\tau_3 \omega A (q \omega A - \text{Suc } i)))$
unfolding $\tau_3\text{-def}$ *compress.simps*
by *(simp del: compress.simps compress-step.simps)*
also have $\dots = \text{compress} (\tau_3 \omega A ((q \omega A - \text{Suc } i) + 1))$
by *(subst $\tau_3\text{-step}$)* *blast*
also have $\dots = \text{compress} (\tau_3 \omega A (q \omega A - i))$
using *True* **by** *(metis Suc-diff-Suc Suc-eq-plus1)*
also have $\dots = \tau \omega A$ **using** *Suc* **by** *auto*
finally show *?thesis* **by** *simp*
next
case *False*
then show *?thesis* **using** *Suc* **by** *simp*
qed
qed

lemma *compress-result:*

assumes $\omega \in \text{pro-set } \Omega$
assumes $x \leq q \omega A$
shows *compress* $(\tau_3 \omega A x) = \tau \omega A$

proof –

obtain *i* **where** *i-def*: $x = q \omega A - i$ **using** *assms* **by** *(metis diff-diff-cancel)*
have *compress* $(\tau_3 \omega A x) = \text{compress} (\tau_3 \omega A (q \omega A - i))$
by *(subst i-def)* *blast*
also have $\dots = \tau \omega A$
using *compress-result-1[OF assms(1)]* **by** *blast*
finally show *?thesis* **by** *simp*

qed

lemma $\tau_0\text{-merge}$:

assumes $(f, g, h) \in \text{pro-set } \Psi$
shows $\tau_0 (f, g, h) (A \cup B) j = \max (\tau_0 (f, g, h) A j) (\tau_0 (f, g, h) B j)$ **(is ?L = ?R)**

proof –

let $?f = \lambda a. \text{int } (f a)$
have $?L = \text{Max} (\{\text{int } (f a) \mid a . a \in A \wedge h (g a) = j\} \cup \{-1\}) \cup$
 $\{\text{int } (f a) \mid a . a \in B \wedge h (g a) = j\} \cup \{-1\})$
unfolding $\tau_0.\text{simps}$
by *(intro arg-cong[where f=Max])* *auto*
also have $\dots = \max (\text{Max} (\{\text{int } (f a) \mid a . a \in A \wedge h (g a) = j\} \cup \{-1\}))$
 $(\text{Max} (\{\text{int } (f a) \mid a . a \in B \wedge h (g a) = j\} \cup \{-1\}))$
by *(intro Max-Un finite-UnI fin-f[OF assms])* *auto*
also have $\dots = ?R$
by *(simp)*
finally show *?thesis* **by** *simp*

qed

lemma $\tau_2\text{-merge}$:

assumes $\omega \in \text{pro-set } \Omega$
shows $\tau_2 \omega (A \cup B) x i j = \max (\tau_2 \omega A x i j) (\tau_2 \omega B x i j)$

proof *(cases i < l)*

case *True*

obtain *f g h* **where** *w-i*: $\omega i = (f, g, h)$ **by** *(metis prod-cases3)*

have $a: (f, g, h) \in \text{pro-set } \Psi$ **using** *w-i[symmetric]* *$\omega\text{-range}$ [OF assms(1)]* **by** *auto*
show *?thesis*

unfolding τ_2 -def τ_1 -def
using *True* **by** (*simp add:w-i τ_0 -merge[OF a] del: τ_0 .simps*)
next
case *False*
thus *?thesis* **by** (*simp add: τ_2 -def*)
qed

lemma *merge1-result*:

assumes $\omega \in \text{pro-set } \Omega$
shows $\text{merge1 } (\tau \omega A) (\tau \omega B) = \tau_3 \omega (A \cup B) (\text{max } (q \omega A) (q \omega B))$

proof –

let $?qmax = \text{max } (q \omega A) (q \omega B)$
obtain u **where** $u\text{-def}: q \omega A + u = ?qmax$
by (*metis add.commute max.commute nat-minus-add-max*)
obtain v **where** $v\text{-def}: q \omega B + v = ?qmax$
by (*metis add.commute nat-minus-add-max*)

have $u = 0 \vee v = 0$ **using** $u\text{-def } v\text{-def}$ **by** *linarith*

moreover have $\tau_2 \omega A (q \omega A) i j - u \geq (-1)$ **if** $u = 0$ **for** $i j$
using *that* **by** (*simp add: τ_2 -def τ_1 -def*)

moreover have $\tau_2 \omega B (q \omega B) i j - v \geq (-1)$ **if** $v = 0$ **for** $i j$
using *that* **by** (*simp add: τ_2 -def τ_1 -def*)

ultimately have $a:\text{max } (\tau_2 \omega A (q \omega A) i j - u) (\tau_2 \omega B (q \omega B) i j - v) \geq (-1)$ **for** $i j$
unfolding *le-max-iff-disj* **by** *blast*

have $\tau_2 \omega (A \cup B) ?qmax = (\lambda i j. \text{max } (\tau_2 \omega A ?qmax i j) (\tau_2 \omega B ?qmax i j))$
using τ_2 -merge[OF *assms*] **by** *blast*

also have $\dots = (\lambda i j. \text{max } (\tau_2 \omega A (q \omega A + u) i j) (\tau_2 \omega B (q \omega B + v) i j))$
unfolding $u\text{-def } v\text{-def}$ **by** *blast*

also have $\dots = (\lambda i j. \text{max } (\text{max } (\tau_2 \omega A (q \omega A) i j - u) (-1)) (\text{max } (\tau_2 \omega B (q \omega B) i j - v) (-1)))$
by (*simp only: τ_2 -step*)

also have $\dots = (\lambda i j. \text{max } (\text{max } (\tau_2 \omega A (q \omega A) i j - u) (\tau_2 \omega B (q \omega B) i j - v)) (-1))$
by (*metis (no-types, opaque-lifting) max.commute max.left-commute max.left-idem*)

also have $\dots = (\lambda i j. \text{max } (\tau_2 \omega A (q \omega A) i j - u) (\tau_2 \omega B (q \omega B) i j - v))$
using a **by** *simp*

also have $\dots = (\lambda i j. \text{max } (\tau_2 \omega A (q \omega A) i j + \text{int } (q \omega A) - ?qmax) (\tau_2 \omega B (q \omega B) i j + \text{int } (q \omega B) - ?qmax))$

by (*subst u-def[symmetric], subst v-def[symmetric]*) *simp*

finally have $\tau_2 \omega (A \cup B) (\text{max } (q \omega A) (q \omega B)) =$

$(\lambda i j. \text{max } (\tau_2 \omega A (q \omega A) i j + \text{int } (q \omega A) - \text{int } (?qmax))$

$(\tau_2 \omega B (q \omega B) i j + \text{int } (q \omega B) - \text{int } (?qmax)))$ **by** *simp*

thus *?thesis*

by (*simp add:Let-def τ -def τ_3 -def*)

qed

lemma *merge-result*:

assumes $\omega \in \text{pro-set } \Omega$

shows $\text{merge } (\tau \omega A) (\tau \omega B) = \tau \omega (A \cup B)$ (**is** $?L = ?R$)

proof –

have $a:\text{max } (q \omega A) (q \omega B) \leq q \omega (A \cup B)$

using $q\text{-mono}$ [OF *assms*] **by** *simp*

have $?L = \text{compress } (\text{merge1 } (\tau \omega A) (\tau \omega B))$

by *simp*

also have $\dots = \text{compress } (\tau_3 \omega (A \cup B) (\text{max } (q \omega A) (q \omega B)))$

by (*subst merge1-result[OF *assms*] blast*)

also have $\dots = ?R$

by (intro compress-result[OF assms] a Un-least)
 finally show ?thesis by blast
 qed

lemma single1-result: single1 ω $x = \tau_3 \omega \{x\} 0$

proof –
 have (case ω i of (f, g, h) \Rightarrow if h (g x) = j \wedge i < l then int (f x) else - 1) = $\tau_2 \omega \{x\} 0$ i j
 for i j
 proof –
 obtain f g h where w-i: ω i = (f, g,h) by (metis prod-cases3)
 show ?thesis
 by (simp add:w-i τ_2 -def τ_1 -def)
 qed
 thus ?thesis
 unfolding τ_3 -def by fastforce
 qed

lemma single-result:

assumes $\omega \in$ pro-set Ω
 shows single ω $x = \tau \omega \{x\}$ (is ?L = ?R)
 proof –
 have ?L = compress (single1 ω x)
 by (simp)
 also have ... = compress ($\tau_3 \omega \{x\} 0$)
 by (subst single1-result) blast
 also have ... = ?R
 by (intro compress-result[OF assms]) auto
 finally show ?thesis by blast
 qed

6.2 Encoding states of the inner algorithm

definition is-state-table :: (nat \times nat \Rightarrow int) \Rightarrow bool where
 is-state-table g = (range g \subseteq $\{-1..\}$ \wedge g ' ($-\{\dots<l\} \times \{\dots<b\}$)) \subseteq $\{-1\}$)

Encoding for state table values:

definition V_e :: int encoding
 where V_e x = (if $x \geq -1$ then N_e (nat (x+1)) else None)

Encoding for state table:

definition T_e' :: (nat \times nat \Rightarrow int) encoding where
 T_e' g = (
 if is-state-table g
 then (List.product [0.. l] [0.. b] \rightarrow_e V_e) (restrict g ($\{\dots<l\} \times \{\dots<b\}$))
 else None)

definition T_e :: (nat \Rightarrow nat \Rightarrow int) encoding
 where T_e f = T_e' (case-prod f)

definition encode-state :: state encoding
 where encode-state = $T_e \times_e N_b$ (nat $\lceil \log 2 n \rceil + 3$)

lemma inj-on-restrict:

assumes $B \subseteq \{f. f ' (- A) \subseteq \{c\}\}$
 shows inj-on ($\lambda x. \text{restrict } x A$) B
 proof (rule inj-onI)
 fix f g assume a:f \in B g \in B restrict f A = restrict g A

have $f x = g x$ **if** $x \in A$ **for** x
 by (intro restrict-eq-imp[OF a(3) that])
moreover have $f x = g x$ **if** $x \notin A$ **for** x
proof –
 have $f x = c$ $g x = c$
 using that a(1,2) assms(1) by auto
 thus ?thesis by simp
qed
ultimately show $f = g$
 by (intro ext) auto
qed

lemma encode-state: is-encoding encode-state

proof –
have is-encoding V_e
 unfolding V_e -def
 by (intro encoding-compose[OF exp-golomb-encoding] inj-onI) auto
hence 0:is-encoding (List.product [0.. l] [0.. b] $\rightarrow_e V_e$)
 by (intro fun-encoding)
have is-encoding T_e'
 unfolding T_e' -def is-state-table-def
 by (intro encoding-compose[OF 0] inj-on-restrict[where $c=-1$]) auto
moreover have inj case-prod
 by (intro injI) (metis curry-case-prod)
ultimately have is-encoding T_e
 unfolding T_e -def by (rule encoding-compose-2)

thus ?thesis
 unfolding encode-state-def
 by (intro dependent-encoding bounded-nat-encoding)
qed

lemma state-bit-count:

assumes $\omega \in \text{pro-set } \Omega$
shows bit-count (encode-state ($\tau \omega A$)) $\leq 2^{36} * (\ln(1/\delta)+1) / \varepsilon^2 + \log 2 (\log 2 n + 3)$
 (is ?L \leq ?R)
proof –
define t **where** $t = \tau_2 \omega A (q \omega A)$

have $\log 2 (\text{real } n) \geq 0$
 using n-gt-0 by simp
hence 0: $-1 < \log 2 (\text{real } n)$
 by simp

have $t x y = -1$ **if** $x < l$ $y \geq b$ **for** $x y$

proof –
obtain $f g h$ **where** ω -def: $\omega x = (f, g, h)$
 by (metis prod-cases3)
have $(f, g, h) \in \text{pro-set } \Psi$
 using ω -range[OF assms] unfolding Pi-def ω -def[symmetric] by auto
hence $h (g a) < b$ **for** a
 using h-range by auto
hence $y \neq h (g a)$ **for** a
 using that(2) not-less by blast
hence aux-4: $\{ \text{int } (f a) \mid a. a \in A \wedge h (g a) = y \} = \{ \}$
 by auto
hence $\max (\text{Max } (\text{insert } (-1) \{ \text{int } (f a) \mid a. a \in A \wedge h (g a) = y \}) - \text{int } (q \omega A)) (-1) =$
 -1

unfolding *aux-4* **by** *simp*
thus *?thesis*
unfolding *t-def* τ_2 -*def* τ_1 -*def* **by** (*simp add:omega-def*)
qed
moreover **have** $t x y = -1$ **if** $x \geq l$ **for** $x y$
using *that unfolding t-def tau2-def tau1-def by simp*
ultimately **have** $1: t x y = -1$ **if** $x \geq l \vee y \geq b$ **for** $x y$
using *that by (meson not-less)*

have $2: t x y \geq -1$ **for** $x y$
unfolding *t-def tau2-def tau1-def by simp*
hence $3: t x y + 1 \geq 0$ **for** $x y$
by (*metis add.commute le-add-same-cancel1 minus-add-cancel*)

have $4: is_state_table$ (*case-prod t*)
using $2\ 1$ **unfolding** *is-state-table-def* **by** *auto*

have $bit_count(T_e (\tau_2 \omega A (q \omega A))) = bit_count(T_e t)$
unfolding *t-def* **by** *simp*
also **have** $\dots = bit_count ((List.product [0..<l] [0..<b] \rightarrow_e V_e) (\lambda(x, y) \in \{..<l\} \times \{..<b\}. t x y))$
using 4 **unfolding** *T_e-def T_e'-def* **by** *simp*
also **have** $\dots =$
 $(\sum x \leftarrow List.product [0..<l] [0..<b]. bit_count (V_e ((\lambda(x, y) \in \{..<l\} \times \{..<b\}. t x y) x)))$
using *restrict-extensional atLeast0LessThan* **by** (*simp add:fun-bit-count*)
also **have** $\dots = (\sum (x, y) \leftarrow List.product [0..<l] [0..<b]. bit_count (V_e (t x y)))$
by (*intro arg-cong[where f=sum-list] map-cong refl*)
(simp add:atLeast0LessThan case-prod-beta)
also **have** $\dots = (\sum x \in \{0..<l\} \times \{0..<b\}. bit_count (V_e (t (fst x) (snd x))))$
by (*subst sum-list-distinct-conv-sum-set*)
(auto intro:distinct-product simp add:case-prod-beta)
also **have** $\dots = (\sum x \in \{..<l\} \times \{..<b\}. bit_count (N_e (nat (t (fst x) (snd x) + 1))))$
using 2 **unfolding** *V_e-def not-less[symmetric]*
by (*intro sum.cong refl arg-cong[where f=bit-count] auto*)
also **have** $\dots = (\sum x \in \{..<l\} \times \{..<b\}. 1 + 2 * of_int \lfloor \log 2 (1 + real (nat (t (fst x) (snd x) + 1))) \rfloor)$
unfolding *exp-golomb-bit-count-exact is-too-large.simps not-less* **by** *simp*
also **have** $\dots = (\sum x \in \{..<l\} \times \{..<b\}. 1 + 2 * of_int \lfloor \log 2 (2 + of_int (t (fst x) (snd x))) \rfloor)$
using 3 **by** (*subst of-nat-nat*) (*auto simp add:ac-simps*)
also **have** $\dots = b * l + 2 * of_int (\sum (i, j) \in \{..<l\} \times \{..<b\}. \lfloor \log 2 (2 + of_int (max (t i j) (-1))) \rfloor)$
using 2 **by** (*subst max-absorb1*) (*auto simp add:case-prod-beta sum.distrib sum-distrib-left*)
also **have** $\dots \leq b * l + 2 * of_int (C_5 * int b * int l)$
using *q-compact[OF assms, where A=A]* **unfolding** *is-too-large.simps not-less t-def[symmetric]*
by (*intro add-mono ereal-mono iffD2[OF of-int-le-iff] mult-left-mono order.refl*)
(simp-all add:ac-simps)
also **have** $\dots = (2 * C_5 + 1) * b * l$
by (*simp add:algebra-simps*)
finally **have** $5: bit_count (T_e (\tau_2 \omega A (q \omega A))) \leq (2 * C_5 + 1) * b * l$
by *simp*

have $C_4 \geq 1$
unfolding *C_4-def* **by** *simp*
moreover **have** $\varepsilon^2 \leq 1$
using *epsilon-lt-1 epsilon-gt-0*
by (*intro power-le-one*) *auto*
ultimately **have** $0 \leq \log 2 (C_4 / \varepsilon^2)$
using *epsilon-gt-0 epsilon-lt-1*
by (*intro iffD2[OF zero-le-log-cancel-iff] divide-pos-pos*) *auto*
hence $6: -1 < \log 2 (C_4 / \varepsilon^2)$
by *simp*

have $b = 2 \text{ powr } (\text{real } (\text{nat } \lceil \log 2 (C_4 / \varepsilon^2) \rceil))$
unfolding $b\text{-def } b\text{-exp-def}$ **by** $(\text{simp add:powr-realpow})$
also have $\dots = 2 \text{ powr } (\lceil \log 2 (C_4 / \varepsilon^2) \rceil)$
using 6 **by** $(\text{intro arg-cong2}[\text{where } f=(\text{powr})] \text{ of-nat-nat refl}) \text{ simp}$
also have $\dots \leq 2 \text{ powr } (\log 2 (C_4 / \varepsilon^2) + 1)$
by $(\text{intro powr-mono}) \text{ auto}$
also have $\dots = 2 * C_4 / \varepsilon^2$
using $\varepsilon\text{-gt-0}$ **unfolding** $\text{powr-add } C_4\text{-def}$
by $(\text{subst powr-log-cancel}) (\text{auto intro:divide-pos-pos})$
finally have $7:b \leq 2 * C_4 / \varepsilon^2$ **by** simp

have $l \leq C_6 * \ln (1 / \delta) + C_6 * \ln 2 + 1$
by (intro l-ubound)
also have $\dots \leq 4 * \ln(1/\delta) + 3+1$
unfolding $C_6\text{-def}$ **by** $(\text{intro add-mono order.refl}) (\text{approximation } 5)$
also have $\dots = 4 * (\ln(1/\delta)+1)$
by simp
finally have $8:l \leq 4 * (\ln(1/\delta)+1)$
by simp

have $\varepsilon^2 = 0 + \varepsilon^2$
by simp
also have $\dots \leq \ln (1 / \delta) + 1$
using $\delta\text{-gt-0 } \delta\text{-lt-1 } \varepsilon\text{-gt-0 } \varepsilon\text{-lt-1}$
by $(\text{intro add-mono power-le-one}) \text{ auto}$
finally have $9: \varepsilon^2 \leq \ln (1 / \delta) + 1$
by simp

have $10: 0 \leq \ln (1 / \delta) + 1$
using $\delta\text{-gt-0 } \delta\text{-lt-1}$ **by** $(\text{intro add-nonneg-nonneg}) \text{ auto}$

have $?L = \text{bit-count } (T_e (\tau_2 \omega A (q \omega A))) + \text{bit-count } (N_{b_e} (\text{nat } \lceil \log 2 (\text{real } n) \rceil + 3) (q \omega A))$
unfolding $\text{encode-state-def } \tau\text{-def } \tau_3\text{-def}$ **by** $(\text{simp add:dependent-bit-count})$
also have $\dots = \text{bit-count}(T_e(\tau_2 \omega A (q \omega A))) + \text{ereal } (1 + \text{of-int}[\log 2 (2 + \text{real } (\text{nat } \lceil \log 2 n \rceil))])$
using $\text{max-s-3}[OF \text{ assms}]$ **by** $(\text{subst bounded-nat-bit-count-2})$
 $(\text{simp-all add:numeral-eq-Suc le-imp-less-Suc floorlog-def})$
also have $\dots = \text{bit-count}(T_e(\tau_2 \omega A (q \omega A))) + \text{ereal } (1 + \text{of-int}[\log 2 (2 + \text{of-int } \lceil \log 2 n \rceil)])$
using 0 **by** simp
also have $\dots \leq \text{bit-count}(T_e(\tau_2 \omega A (q \omega A))) + \text{ereal } (1 + \log 2 (2 + \text{of-int } \lceil \log 2 n \rceil))$
by $(\text{intro add-mono ereal-mono}) \text{ simp-all}$
also have $\dots \leq \text{bit-count}(T_e(\tau_2 \omega A (q \omega A))) + \text{ereal } (1 + \log 2 (2 + (\log 2 n + 1)))$
using 0 n-gt-0 **by** $(\text{intro add-mono ereal-mono iffD2}[OF \text{ log-le-cancel-iff}] \text{ add-pos-nonneg}) \text{ auto}$
also have $\dots = \text{bit-count}(T_e(\tau_2 \omega A (q \omega A))) + \text{ereal } (1 + \log 2 (\log 2 n + 3))$
by $(\text{simp add:ac-simps})$
also have $\dots \leq \text{ereal } ((2 * C_5 + 1) * b * l) + \text{ereal } (1 + \log 2 (\log 2 n + 3))$
by $(\text{intro add-mono } 5) \text{ auto}$
also have $\dots = (2 * C_5 + 1) * \text{real } b * \text{real } l + \log 2 (\log 2 n + 3) + 1$
by simp
also have $\dots \leq (2 * C_5 + 1) * (2 * C_4 / \varepsilon^2) * \text{real } l + \log 2 (\log 2 n + 3) + 1$
unfolding $C_5\text{-def}$
by $(\text{intro ereal-mono mult-right-mono mult-left-mono add-mono } 7) \text{ auto}$
also have $\dots = (4 * \text{of-int } C_5 + 2) * C_4 * \text{real } l / \varepsilon^2 + \log 2 (\log 2 n + 3) + 1$
by simp
also have $\dots \leq (4 * \text{of-int } C_5 + 2) * C_4 * (4 * (\ln(1/\delta) + 1)) / \varepsilon^2 + \log 2 (\log 2 n + 3) + 1$
using $\varepsilon\text{-gt-0}$ **unfolding** $C_5\text{-def } C_4\text{-def}$
by $(\text{intro ereal-mono add-mono order.refl divide-right-mono mult-left-mono } 8) \text{ auto}$
also have $\dots = ((2 * 33 + 1) * 9 * 2^{26}) * (\ln(1/\delta) + 1) / \varepsilon^2 + \log 2 (\log 2 n + 3) + 1$

unfolding C_5 -def C_4 -def **by** *simp*
also have $\dots \leq (2^{36}-1) * (\ln(1/\delta)+1) / \varepsilon^2 + \log 2 (\log 2 n + 3) + (\ln(1/\delta)+1) / \varepsilon^2$
using ε -gt-0 δ -gt-0 ε -lt-1 9 10
by (*intro add-mono ereal-mono divide-right-mono mult-right-mono mult-left-mono*) *simp-all*
also have $\dots = 2^{36} * (\ln(1/\delta)+1) / \varepsilon^2 + \log 2 (\log 2 n + 3)$
by (*simp add:divide-simps*)
finally show *?thesis*
by *simp*
qed

lemma *random-bit-count*:

pro-size $\Omega \leq 2 \text{ powr } (4 * \log 2 n + 48 * (\log 2 (1 / \varepsilon) + 16))^2 + (55 + 60 * \ln (1 / \delta))^3$
(is ?L ≤ ?R)

proof –

have $1: \log 2 (\text{real } n) \geq 0$
using n -gt-0 **by** *simp*
hence $0: -1 < \log 2 (\text{real } n)$
by *simp*

have $10: \log 2 C_4 \leq 27$
unfolding C_4 -def **by** (*approximation 10*)
have $\varepsilon^2 \leq 1$
using ε -gt-0 ε -lt-1 **by** (*intro power-le-one*) *auto*

also have $\dots \leq C_4$
unfolding C_4 -def **by** *simp*
finally have $\varepsilon^2 \leq C_4$ **by** *simp*
hence $9: 0 \leq \log 2 (C_4 / \varepsilon^2)$
using ε -gt-0 **unfolding** C_4 -def
by (*intro iffD2[OF zero-le-log-cancel-iff]*) *simp-all*
hence $2: -1 < \log 2 (C_4 / \varepsilon^2)$
by *simp*

have $3: 0 < C_7 * b^2$ **unfolding** C_7 -def **using** *b-min* **by** (*intro Rings.mult-pos-pos*) *auto*

have $0 \leq \log 2 (\text{real } C_7) + \text{real } (b\text{-exp} * 2)$
unfolding C_7 -def **by** (*intro add-nonneg-nonneg*) *auto*
hence $4: -1 < \log 2 (\text{real } C_7) + \text{real } (b\text{-exp} * 2)$ **by** *simp*
have $(2, n\text{-exp}) = \text{split-power} (\text{pro-size } (\mathcal{G} \ n\text{-exp}))$
unfolding *geom-pro-size* **by** (*intro split-power-prime[symmetric] n-exp-gt-0*) *auto*
hence $\text{real } (\text{pro-size } \Psi_1) = \text{real } (2 \wedge (2 * \max n\text{-exp} (\text{nat } \lceil \log (\text{real } 2) (\text{real } n) \rceil)))$
by (*intro arg-cong[where f=real] hash-pro-size[OF Ψ_1 n-gt-0]*)
also have $\dots = 2 \wedge (2 * \max n\text{-exp} (\text{nat } \lceil \log 2 (\text{real } n) \rceil))$ **by** *simp*
also have $\dots = 2 \wedge (2 * \max 1 (\text{nat } \lceil \log 2 (\text{real } n) \rceil))$ **unfolding** $n\text{-exp-def}$ **by** *simp*
also have $\dots \leq 2 \text{ powr } (2 * \max (\text{nat } \lceil \log 2 (\text{real } n) \rceil) 1)$
by (*subst powr-realpow*) *auto*
also have $\dots = 2 \text{ powr } (2 * \max (\text{real } (\text{nat } \lceil \log 2 (\text{real } n) \rceil)) 1)$
using n -gt-0 **unfolding** *of-nat-mult of-nat-max* **by** *simp*
also have $\dots = 2 \text{ powr } (2 * \max (\text{of-int } \lceil \log 2 (\text{real } n) \rceil) 1)$
using 0 **by** (*subst of-nat-nat*) *simp-all*
also have $\dots \leq 2 \text{ powr } (2 * \max (\log 2 (\text{real } n) + 1) 1)$
by (*intro powr-mono mult-left-mono max-mono*) *auto*
also have $\dots = 2 \text{ powr } (2 * (\log 2 (\text{real } n) + 1))$
using 1 **by** (*subst max-absorb1*) *auto*
finally have $5: \text{real } (\text{pro-size } \Psi_1) \leq 2 \text{ powr } (2 * \log 2 n + 2)$
by *simp*

have $(2, 5 + b\text{-exp} * 2) = \text{split-power} (2 \wedge (5 + b\text{-exp} * 2))$
by (*intro split-power-prime[symmetric]*) *auto*

also have ... = *split-power* ($C_7 * b^2$)
 unfolding C_7 -def b -def *power-mult*[*symmetric*] *power-add* by *simp*
 also have ... = *split-power* (*pro-size* (\mathcal{N} ($C_7 * b^2$)))
 unfolding C_7 -def b -def by (*subst nat-pro-size*) *auto*
 finally have ($2, 5 + b\text{-exp} * 2$) = *split-power* (*pro-size* (\mathcal{N} ($C_7 * b^2$))) by *simp*
 hence *real* (*pro-size* Ψ_2) = *real* ($2 \wedge (2 * \max (5 + b\text{-exp} * 2) (\text{nat } \lceil \log (\text{real } 2) (\text{real } n) \rceil))$)
 by (*intro arg-cong*[*where f=real*] *hash-pro-size'*[*OF* Ψ_2 $n\text{-gt-0}$])
 also have ... = $2 \wedge (\max (5 + b\text{-exp} * 2) (\text{nat } \lceil \log 2 (\text{real } n) \rceil) * 2)$ by *simp*
 also have ... $\leq 2 \wedge (((5 + b\text{-exp} * 2) + (\text{nat } \lceil \log 2 (\text{real } n) \rceil)) * 2)$
 by (*intro power-increasing mult-right-mono*) *auto*
 also have ... = $2 \text{ powr } ((5 + b\text{-exp} * 2 + \text{real } (\text{nat } \lceil \log 2 (\text{real } n) \rceil)) * 2)$
 by (*subst powr-realpow*[*symmetric*]) *auto*
 also have ... = $2 \text{ powr } ((5 + \text{of-int } b\text{-exp} * 2 + \text{of-int } \lceil \log 2 (\text{real } n) \rceil) * 2)$
 using 0 by (*subst of-nat-nat*) *auto*
 also have ... $\leq 2 \text{ powr } ((5 + \text{of-int } b\text{-exp} * 2 + (\log 2 (\text{real } n) + 1)) * 2)$
 by (*intro powr-mono mult-right-mono add-mono*) *simp-all*
 also have ... = $2 \text{ powr } (12 + 4 * \text{real} (\text{nat } \lceil \log 2 (C_4 / \varepsilon^2) \rceil) + \log 2 (\text{real } n) * 2)$
 unfolding $b\text{-exp-def}$ by (*simp add:ac-simps*)
 also have ... = $2 \text{ powr } (12 + 4 * \text{real-of-int } \lceil \log 2 (C_4 / \varepsilon^2) \rceil + \log 2 (\text{real } n) * 2)$
 using 2 by (*subst of-nat-nat*) *simp-all*
 also have ... $\leq 2 \text{ powr } (12 + 4 * (\log 2 (C_4 / \varepsilon^2) + 1) + \log 2 (\text{real } n) * 2)$
 by (*intro powr-mono add-mono order.refl mult-left-mono*) *simp-all*
 also have ... = $2 \text{ powr } (2 * \log 2 n + 4 * \log 2 (C_4 / \varepsilon^2) + 16)$
 by (*simp add:ac-simps*)
 finally have $6:\text{real} (\text{pro-size } \Psi_2) \leq 2 \text{ powr } (2 * \log 2 n + 4 * \log 2 (C_4 / \varepsilon^2) + 16)$
 by *simp*
 have ($2, b\text{-exp}$) = *split-power* ($2 \wedge b\text{-exp}$)
 using $b\text{-exp-ge-26}$ by (*intro split-power-prime*[*symmetric*]) *auto*
 also have ... = *split-power* (*pro-size* (\mathcal{N} b))
 unfolding b -def by (*subst nat-pro-size*) *auto*
 finally have ($2, b\text{-exp}$) = *split-power* (*pro-size* (\mathcal{N} b)) by *simp*
 hence *real* (*pro-size* Ψ_3) = *real* ($2 \wedge (k * \max b\text{-exp} (\text{nat } \lceil \log (\text{real } 2) (\text{real } (C_7 * b^2)) \rceil))$)
 by (*intro arg-cong*[*where f=real*] *hash-pro-size'*[*OF* Ψ_3]) (*simp-all add:C7-def b-def*)
 also have ... = $2 \wedge (k * \max b\text{-exp} (\text{nat } \lceil \log 2 (\text{real } C_7 * (2 \wedge (b\text{-exp} * 2)) \rceil))$)
 unfolding b -def *power-mult* by *simp*
 also have ... = $2 \wedge (\max b\text{-exp} (\text{nat } \lceil \log 2 C_7 + \log 2 (2 \wedge (b\text{-exp} * 2)) \rceil) * k)$
 unfolding C_7 -def by (*subst log-mult*) *simp-all*
 also have ... = $2 \wedge (\max b\text{-exp} (\text{nat } \lceil \log 2 C_7 + (b\text{-exp} * 2) \rceil) * k)$
 by (*subst log-nat-power*) *simp-all*
 also have ... = $2 \text{ powr } (\max (\text{real } b\text{-exp}) (\text{real } (\text{nat } \lceil \log 2 C_7 + (b\text{-exp} * 2) \rceil)) * \text{real } k)$
 by (*subst powr-realpow*[*symmetric*]) *simp-all*
 also have ... = $2 \text{ powr } (\max (\text{real } b\text{-exp}) (\text{of-int } \lceil \log 2 C_7 + (b\text{-exp} * 2) \rceil) * \text{real } k)$
 using 4 by (*subst of-nat-nat*) *simp-all*
 also have ... $\leq 2 \text{ powr } (\max (\text{real } b\text{-exp}) (\log 2 C_7 + \text{real } b\text{-exp} * 2 + 1) * \text{real } k)$
 by (*intro powr-mono mult-right-mono max-mono-2*) *simp-all*
 also have ... = $2 \text{ powr } ((\log 2 (2^5) + \text{real } b\text{-exp} * 2 + 1) * \text{real } k)$
 unfolding C_7 -def by (*subst max-absorb2*) *simp-all*
 also have ... = $2 \text{ powr } ((\text{real } b\text{-exp} * 2 + 6) * \text{real } k)$
 unfolding C_7 -def by (*subst log-nat-power*) (*simp-all add:ac-simps*)
 also have ... = $2 \text{ powr } ((\text{of-int } \lceil \log 2 (C_4 / \varepsilon^2) \rceil * 2 + 6) * \text{real } k)$
 using 2 unfolding $b\text{-exp-def}$ by (*subst of-nat-nat*) *simp-all*
 also have ... $\leq 2 \text{ powr } (((\log 2 (C_4 / \varepsilon^2) + 1) * 2 + 6) * \text{real } k)$
 by (*intro powr-mono mult-right-mono add-mono*) *simp-all*
 also have ... = $2 \text{ powr } ((\log 2 (C_4 / \varepsilon^2) * 2 + 8) * \text{real } k)$
 by (*simp add:ac-simps*)
 finally have $7:\text{real} (\text{pro-size } \Psi_3) \leq 2 \text{ powr } ((\log 2 (C_4 / \varepsilon^2) * 2 + 8) * \text{real } k)$
 by *simp*

have $\ln(\text{real } b) \geq 0$
using *b-min* **by** *simp*
hence $\text{real } k = \text{of-int } \lceil 7.5 * \ln(\text{real } b) + 16 \rceil$
unfolding *k-def C₂-def C₃-def* **by** (*subst of-nat-nat*) *simp-all*
also have $\dots \leq (7.5 * \ln(\text{real } b) + 16) + 1$
unfolding *b-def* **by** (*intro of-int-ceiling-le-add-one*)
also have $\dots = 7.5 * \ln(2 \text{ powr } b\text{-exp}) + 17$
unfolding *b-def* **using** *powr-realpow* **by** *simp*
also have $\dots = \text{real } b\text{-exp} * (7.5 * \ln 2) + 17$
unfolding *powr-def* **by** *simp*
also have $\dots \leq \text{real } b\text{-exp} * 6 + 17$
by (*intro add-mono mult-left-mono order.refl of-nat-0-le-iff*) (*approximation 5*)
also have $\dots = \text{of-int } \lceil \log 2 (C_4 / \varepsilon^2) \rceil * 6 + 17$
using 2 **unfolding** *b-exp-def* **by** (*subst of-nat-nat*) *simp-all*
also have $\dots \leq (\log 2 (C_4 / \varepsilon^2) + 1) * 6 + 17$
by (*intro add-mono mult-right-mono*) *simp-all*
also have $\dots = 6 * \log 2 (C_4 / \varepsilon^2) + 23$
by *simp*
finally have $8:\text{real } k \leq 6 * \log 2 (C_4 / \varepsilon^2) + 23$
by *simp*

have $\text{real } (\text{pro-size } \Psi) = \text{real } (\text{pro-size } \Psi_1) * \text{real } (\text{pro-size } \Psi_2) * \text{real } (\text{pro-size } \Psi_3)$
unfolding *Ψ-def prod-pro-size* **by** *simp*
also have $\dots \leq$
 $2 \text{ powr}(2 * \log 2 n + 2) * 2 \text{ powr}(2 * \log 2 n + 4 * \log 2 (C_4 / \varepsilon^2) + 16) * 2 \text{ powr}((\log 2 (C_4 / \varepsilon^2) * 2 + 8) * \text{real } k)$
by (*intro mult-mono 5 6 7 mult-nonneg-nonneg*) *simp-all*
also have $\dots = 2 \text{ powr}(2 * \log 2 n + 2 + 2 * \log 2 n + 4 * \log 2 (C_4 / \varepsilon^2) + 16 + (\log 2 (C_4 / \varepsilon^2) * 2 + 8) * \text{real } k)$
unfolding *powr-add* **by** *simp*
also have $\dots = 2 \text{ powr}(4 * \log 2 n + 4 * \log 2 (C_4 / \varepsilon^2) + 18 + (2 * \log 2 (C_4 / \varepsilon^2) + 8) * \text{real } k)$
by (*simp add:ac-simps*)
also have $\dots \leq$
 $2 \text{ powr}(4 * \log 2 n + 4 * \log 2 (C_4 / \varepsilon^2) + 18 + (2 * \log 2 (C_4 / \varepsilon^2) + 8) * (6 * \log 2 (C_4 / \varepsilon^2) + 23))$
using 9 **by** (*intro powr-mono add-mono order.refl mult-left-mono 8 add-nonneg-nonneg*) *simp-all*
also have $\dots = 2 \text{ powr}(4 * \log 2 n + 12 * \log 2 (C_4 / \varepsilon^2)^2 + 98 * \log 2 (C_4 / \varepsilon^2) + 202)$
by (*simp add:algebra-simps power2-eq-square*)
also have $\dots \leq 2 \text{ powr}(4 * \log 2 n + 12 * \log 2 (C_4 / \varepsilon^2)^2 + 120 * \log 2 (C_4 / \varepsilon^2) + 300)$
using 9 **by** (*intro powr-mono add-mono order.refl mult-right-mono*) *simp-all*
also have $\dots = 2 \text{ powr}(4 * \log 2 n + 12 * (\log 2 (C_4 * (1 / \varepsilon)^2) + 5)^2)$
by (*simp add:power2-eq-square algebra-simps*)
also have $\dots = 2 \text{ powr}(4 * \log 2 n + 12 * (\log 2 C_4 + \log 2 ((1 / \varepsilon)^2) + 5)^2)$
unfolding *C₄-def* **using** *ε-gt-0* **by** (*subst log-mult*) *auto*
also have $\dots \leq 2 \text{ powr}(4 * \log 2 n + 12 * (27 + \log 2 ((1 / \varepsilon)^2) + 5)^2)$
using *ε-gt-0 ε-lt-1*
by (*intro powr-mono add-mono order.refl mult-left-mono power-mono add-nonneg-nonneg 10*) *(simp-all add:C₄-def)*
also have $\dots = 2 \text{ powr}(4 * \log 2 n + 12 * (2 * (\log 2 (1 / \varepsilon) + 16))^2)$
using *ε-gt-0* **by** (*subst log-nat-power*) *(simp-all add:ac-simps)*
also have $\dots = 2 \text{ powr}(4 * \log 2 n + 48 * (\log 2 (1 / \varepsilon) + 16)^2)$
unfolding *power-mult-distrib* **by** *simp*
finally have $19:\text{real } (\text{pro-size } \Psi) \leq 2 \text{ powr}(4 * \log 2 n + 48 * (\log 2 (1 / \varepsilon) + 16)^2)$
by *simp*

have $0 \leq \ln \Lambda / \ln (19 / 20)$
using *Λ-gt-0 Λ-le-1* **by** (*intro divide-nonpos-neg*) *simp-all*

hence 11: $-1 < \ln \Lambda / \ln (19 / 20)$ **by** *simp*
have 12: $\ln (19 / 20) \leq -(0.05::\text{real}) - \ln (1 / 16) \leq (2.8::\text{real})$ **by** (*approximation 10*)
have 13: $\ln l \geq 0$ **using** *l-gt-0* **by** *auto*
have $\ln l^3 = 27 * (0 + \ln l/3)^3$ **by** (*simp add:power3-eq-cube*)
also have $\dots \leq 27 * (1 + \ln l/\text{real } 3)^3$
using *l-gt-0* **by** (*intro mult-left-mono add-mono power-mono*) *auto*
also have $\dots \leq 27 * (\exp (\ln l))$
using *l-gt-0 13* **by** (*intro mult-left-mono exp-ge-one-plus-x-over-n-power-n*) *linarith*
also have $\dots = 27 * \text{real } l$ **using** *l-gt-0* **by** (*subst exp-ln*) *auto*
finally have 14: $\ln l^3 \leq 27 * \text{real } l$ **by** *simp*

have 15: $C_6 * \ln (2 / \delta) > 0$
using *delta-lt-1 delta-gt-0 unfolding C6-def*
by (*intro Rings.mult-pos-pos ln-gt-zero*) *auto*
hence $1 \leq \text{real-of-int } \lceil C_6 * \ln (2 / \delta) \rceil$ **by** *simp*
hence 16: $1 \leq 3 * \text{real-of-int } \lceil C_6 * \ln (2 / \delta) \rceil$ **by** *argo*

have 17: $12 * \ln 2 \leq (9::\text{real})$ **by** (*approximation 5*)

have 16 $\wedge ((l - 1) * \text{nat} \lceil \ln \Lambda / \ln 0.95 \rceil) = 16 \text{ powr } (\text{real } (l-1) * \text{real} (\text{nat} \lceil \ln \Lambda / \ln (19 / 20) \rceil))$
by (*subst powr-realpow[symmetric]*) *auto*
also have $\dots = 16 \text{ powr } (\text{real } (l-1) * \text{of-int } \lceil \ln \Lambda / \ln (19 / 20) \rceil)$
using 11 **by** (*subst of-nat-nat simp-all*)
also have $\dots \leq 16 \text{ powr } (\text{real } (l-1) * (\ln \Lambda / \ln (19/20)+1))$
by (*intro powr-mono mult-left-mono*) *auto*
also have $\dots = 16 \text{ powr } ((\text{real } l - 1) * (\ln \Lambda / \ln (19/20)+1))$
using *l-gt-0* **by** (*subst of-nat-diff*) *auto*
also have $\dots \leq 16 \text{ powr } ((\text{real } l - 1) * (\ln \Lambda / (-0.05)+1))$
using *l-gt-0* *Lambda-gt-0* *Lambda-le-1*
by (*intro powr-mono mult-left-mono add-mono divide-left-mono-neg 12*) *auto*
also have $\dots = 16 \text{ powr } ((\text{real } l - 1) * (20 * (-\ln \Lambda)+1))$
by (*simp add:algebra-simps*)
also have $\dots = 16 \text{ powr } ((\text{real } l - 1) * (20 * -(\min (\ln (1/16)) (-l * \ln l^3))+1))$
unfolding *Lambda-def* **by** (*subst ln-min-swap*) *auto*
also have $\dots = 16 \text{ powr } ((\text{real } l - 1) * (20 * \max (-\ln (1/16)) (l * \ln l^3)+1))$
by (*intro-cong* $[\sigma_2 (\text{powr}), \sigma_2 (+), \sigma_2 (*)]$) *simp*
also have $\dots \leq 16 \text{ powr } ((\text{real } l - 1) * (20 * \max (2.8) (l * \ln l^3)+1))$
using *l-gt-0* **by** (*intro powr-mono mult-left-mono add-mono max-mono 12*) *auto*
also have $\dots \leq 16 \text{ powr } ((\text{real } l - 1) * (20 * (2.8 + l * \ln l^3)+1))$
using *l-gt-0* **by** (*intro powr-mono mult-left-mono add-mono*) *auto*
also have $\dots = 16 \text{ powr } ((\text{real } l - 1) * (20 * (l * \ln l^3)+57))$
by (*simp add:algebra-simps*)
also have $\dots \leq 16 \text{ powr } ((\text{real } l - 1) * (20 * (\text{real } l * (27 * \text{real } l))+57))$
using *l-gt-0* **by** (*intro powr-mono mult-left-mono add-mono 14*) *auto*
also have $\dots = 16 \text{ powr } (540 * \text{real } l^3 - 540 * \text{real } l^2 + 57 * \text{real } l - 57)$
by (*simp add:algebra-simps numeral-eq-Suc*)
also have $\dots \leq 16 \text{ powr } (540 * \text{real } l^3 - 540 * \text{real } l^2 + 180 * \text{real } l - 20)$
by (*intro powr-mono add-mono diff-mono order.refl mult-right-mono*) *auto*
also have $\dots = 16 \text{ powr } (20 * (3 * \text{real } l - 1)^3)$
by (*simp add: algebra-simps power3-eq-cube power2-eq-square*)
also have $\dots = 16 \text{ powr } (20 * (3 * \text{of-int } \lceil C_6 * \ln (2 / \delta) \rceil - 1)^3)$
using 15 **unfolding** *l-def* **by** (*subst of-nat-nat*) *auto*
also have $\dots \leq 16 \text{ powr } (20 * (3 * (C_6 * \ln (2 / \delta) + 1) - 1)^3)$
using 16 **by** (*intro powr-mono mult-left-mono power-mono diff-mono*) *auto*

```

also have ... = 16 powr (20 * (2 + 12 * ln (2 * (1 / δ))) ^ 3)
  by (simp add: algebra-simps C6-def)
also have ... = (2 powr 4) powr (20 * (2 + 12 * (ln 2 + ln(1/ δ)))) ^ 3)
  using δ-gt-0 by (subst ln-mult) auto
also have ... = 2 powr (80 * (2 + 12 * ln 2 + 12 * ln (1 / δ))) ^ 3)
  unfolding powr-powr by (simp add: ac-simps)
also have ... ≤ 2 powr (80 * (2 + 9 + 12 * ln (1 / δ))) ^ 3)
  using δ-gt-0 δ-lt-1
  by (intro powr-mono mult-left-mono power-mono add-mono 17 add-nonneg-nonneg) auto
also have ... = 2 powr (80 * (11 + 12 * ln (1 / δ))) ^ 3) by simp
also have ... ≤ 2 powr (53 * (11 + 12 * ln (1 / δ))) ^ 3)
  using δ-gt-0 δ-lt-1 by (intro powr-mono mult-right-mono) (auto intro!: add-nonneg-nonneg)
also have ... = 2 powr ((55 + 60 * ln (1 / δ)) ^ 3)
  unfolding power-mult-distrib[symmetric] by simp
finally have 18: 16(l-1) * nat[ln Λ / ln (19/20)] ≤ 2 powr ((55 + 60 * ln (1 / δ)) ^ 3)
  by simp

```

```

have ?L = real (pro-size Ψ) * 16(l-1) * nat [ln Λ / ln (19 / 20)]
  unfolding expander-pro-size[OF Ω] by simp
also have ... ≤ 2 powr (4 * log 2 n + 48 * (log 2 (1/ε) + 16) ^ 2) * 2 powr ((55 + 60 * ln (1 / δ)) ^ 3)
  by (intro mult-mono 18 19) simp-all
also have ... = 2 powr (4 * log 2 n + 48 * (log 2 (1 / ε) + 16) ^ 2 + (55 + 60 * ln (1 / δ)) ^ 3)
  unfolding powr-add[symmetric] by simp
finally show ?thesis by simp
qed

```

end

unbundle no-intro-cong-syntax

end

7 Accuracy without cutoff

This section verifies that each of the l estimate have the required accuracy with high probability assuming that there was no cut-off, i.e., that $s = 0$. Section 9 will then show that this remains true as long as the cut-off is below $t f$ the subsampling threshold.

theory *Distributed-Distinct-Elements-Accuracy-Without-Cutoff*

imports

Concentration-Inequalities.Bienaymes-Identity

Distributed-Distinct-Elements-Inner-Algorithm

Distributed-Distinct-Elements-Balls-and-Bins

begin

hide-fact (open) *Discrete.log-mono*

no-notation *Polynomials.var* (X₁)

locale *inner-algorithm-fix-A* = *inner-algorithm* +

fixes A

assumes *A-range*: $A \subseteq \{..<n\}$

assumes *A-nonempty*: $\{\} \neq A$

begin

definition $X :: nat$ **where** $X = card A$

definition $q-max$ **where** $q-max = nat ([log 2 X] - b-exp)$

definition $t :: (nat \Rightarrow nat) \Rightarrow int$
where $t f = int (Max (f ' A)) - b-exp + 9$

definition $s :: (nat \Rightarrow nat) \Rightarrow nat$
where $s f = nat (t f)$

definition $R :: (nat \Rightarrow nat) \Rightarrow nat\ set$
where $R f = \{a. a \in A \wedge f a \geq s f\}$

definition $r :: nat \Rightarrow (nat \Rightarrow nat) \Rightarrow nat$
where $r x f = card \{a. a \in A \wedge f a \geq x\}$

definition p **where** $p = (\lambda(f,g,h). card \{j \in \{..<b\}. \tau_1 (f,g,h) A \ 0 j \geq s f\})$

definition Y **where** $Y = (\lambda(f,g,h). 2 \wedge s f * \varrho-inv (p (f,g,h)))$

lemma $fin-A$: *finite A*
using $A-range\ finite-nat-iff-bounded$ **by** $auto$

lemma $X-le-n$: $X \leq n$
proof –
have $card\ A \leq card \{..<n\}$
by ($intro\ card-mono\ A-range$) $simp$
thus $?thesis$
unfolding $X-def$ **by** $simp$
qed

lemma $X-ge-1$: $X \geq 1$
unfolding $X-def$
using $fin-A\ A-nonempty$ **by** ($simp\ add: leI$)

lemma $of-bool-square$: $(of-bool\ x)^2 = ((of-bool\ x)::real)$
by ($cases\ x, auto$)

lemma $r-eq$: $r\ x\ f = (\sum\ a \in A. (of-bool (x \leq f a) :: real))$
unfolding $r-def\ of-bool-def\ sum.If-cases[OF\ fin-A]$
by ($simp\ add: Collect-conj-eq$)

lemma
shows
 $r-exp: (\int \omega. real (r\ x\ \omega) \ \partial \Psi_1) = real\ X * (of-bool (x \leq max (nat \lceil \log 2 n \rceil) 1) / 2^x)$ **and**
 $r-var: measure-pmf.variance\ \Psi_1 (\lambda \omega. real (r\ x\ \omega)) \leq (\int \omega. real (r\ x\ \omega) \ \partial \Psi_1)$

proof –
define $V :: nat \Rightarrow (nat \Rightarrow nat) \Rightarrow real$ **where** $V = (\lambda a\ f. of-bool (x \leq f a))$

have $V-exp: (\int \omega. V\ a\ \omega \ \partial \Psi_1) = of-bool (x \leq max (nat \lceil \log 2 n \rceil) 1) / 2^x$
(is $?L = ?R$) **if** $a \in A$ **for** a

proof –
have $a-le-n: a < n$
using $that\ A-range$ **by** $auto$

have $?L = (\int \omega. indicator \{f. x \leq f a\} \ \omega \ \partial \Psi_1)$
unfolding $V-def$ **by** ($intro\ integral-cong-AE$) $auto$
also have $... = measure (map-pmf (\lambda \omega. \omega\ a) (sample-pro\ \Psi_1)) \{f. x \leq f\}$
by $simp$
also have $... = measure (\mathcal{G}\ n-exp) \{f. x \leq f\}$
by ($subst\ hash-pro-component[OF\ \Psi_1\ a-le-n]$) $auto$
also have $... = of-bool (x \leq max (nat \lceil \log 2 n \rceil) 1) / 2^x$

unfolding *geom-prob n-exp-def* **by** *simp*
finally show *?thesis* **by** *simp*
qed

have $b: (\int \omega. \text{real } (r \ x \ \omega) \ \partial \ \Psi_1) = (\sum a \in A. (\int \omega. V \ a \ \omega \ \partial \ \Psi_1))$
unfolding *r-eg V-def* **by** (*intro Bochner-Integration.integral-sum*) *auto*
also have $\dots = (\sum a \in A. \text{of-bool } (x \leq \max (\text{nat } \lceil \log 2 \ n \rceil) \ 1) / 2^{\wedge} x)$
using *V-exp* **by** (*intro sum.cong*) *auto*
also have $\dots = X * (\text{of-bool } (x \leq \max (\text{nat } \lceil \log 2 \ n \rceil) \ 1) / 2^{\wedge} x)$
using *X-def* **by** *simp*
finally show $(\int \omega. \text{real } (r \ x \ \omega) \ \partial \ \Psi_1) = \text{real } X * (\text{of-bool } (x \leq \max (\text{nat } \lceil \log 2 \ n \rceil) \ 1) / 2^{\wedge} x)$
by *simp*

have $(\int \omega. (V \ a \ \omega)^2 \ \partial \ \Psi_1) = (\int \omega. V \ a \ \omega \ \partial \ \Psi_1)$ **for** a
unfolding *V-def of-bool-square* **by** *simp*

hence $a: \text{measure-pmf.variance } \Psi_1 \ (V \ a) \leq \text{measure-pmf.expectation } \Psi_1 \ (V \ a)$ **for** a
by (*subst measure-pmf.variance-eg*) *auto*

have $J \subseteq A \implies \text{card } J = 2 \implies \text{prob-space.indep-vars } \Psi_1 \ (\lambda-. \text{borel}) \ V \ J$ **for** J
unfolding *V-def* **using** *A-range finite-subset[OF - fin-A]*
by (*intro prob-space.indep-vars-compose2* [**where** $Y = \lambda i \ y. \text{of-bool}(x \leq y)$ **and** $M' = \lambda-. \text{discrete}$] *hash-pro-indep[OF \Psi_1]*) (*auto simp: prob-space-measure-pmf*)
hence $\text{measure-pmf.variance } \Psi_1 \ (\lambda \omega. \text{real } (r \ x \ \omega)) = (\sum a \in A. \text{measure-pmf.variance } \Psi_1 \ (V \ a))$
unfolding *r-eg V-def* **by** (*intro measure-pmf.bienaymes-identity-pairwise-indep-2 fin-A*) *simp-all*
also have $\dots \leq (\sum a \in A. (\int \omega. V \ a \ \omega \ \partial \ \Psi_1))$
by (*intro sum-mono a*)
also have $\dots = (\int \omega. \text{real } (r \ x \ \omega) \ \partial \ \Psi_1)$
unfolding b **by** *simp*
finally show $\text{measure-pmf.variance } \Psi_1 \ (\lambda \omega. \text{real } (r \ x \ \omega)) \leq (\int \omega. \text{real } (r \ x \ \omega) \ \partial \ \Psi_1)$ **by** *simp*
qed

definition E_1 **where** $E_1 = (\lambda(f,g,h). 2 \ \text{powr } (-t \ f) * X \in \{b/2^{\wedge} 16 .. b/2\})$

lemma *t-low*:
 $\text{measure } \Psi_1 \ \{f. \text{of-int } (t \ f) < \log 2 \ (\text{real } X) + 1 - b\text{-exp}\} \leq 1/2^{\wedge} 7$ (**is** $?L \leq ?R$)
proof (*cases* $\log 2 \ (\text{real } X) \geq 8$)
case *True*
define $Z :: (\text{nat} \Rightarrow \text{nat}) \Rightarrow \text{real}$ **where** $Z = r \ (\text{nat } \lceil \log 2 \ (\text{real } X) - 8 \rceil)$

have $\log 2 \ (\text{real } X) \leq \log 2 \ (\text{real } n)$
using *X-le-n X-ge-1* **by** (*intro log-mono*) *auto*
hence $\text{nat } \lceil \log 2 \ (\text{real } X) - 8 \rceil \leq \text{nat } \lceil \log 2 \ (\text{real } n) \rceil$
by (*intro nat-mono ceiling-mono*) *simp*
hence $a: (\text{nat } \lceil \log 2 \ (\text{real } X) - 8 \rceil \leq \max (\text{nat } \lceil \log 2 \ (\text{real } n) \rceil) \ 1)$
by *simp*

have $b: \text{real } (\text{nat } (\lceil \log 2 \ (\text{real } X) \rceil - 8)) \leq \log 2 \ (\text{real } X) - 7$
using *True* **by** *linarith*

have $2^{\wedge} 7 = \text{real } X / (2 \ \text{powr } (\log 2 \ X) * 2 \ \text{powr } (-7))$
using *X-ge-1* **by** *simp*
also have $\dots = \text{real } X / (2 \ \text{powr } (\log 2 \ X - 7))$
by (*subst powr-add[symmetric]*) *simp*
also have $\dots \leq \text{real } X / (2 \ \text{powr } (\text{real } (\text{nat } \lceil \log 2 \ (\text{real } X) - 8 \rceil)))$
using b **by** (*intro divide-left-mono powr-mono*) *auto*
also have $\dots = \text{real } X / 2^{\wedge} \text{nat } \lceil \log 2 \ (\text{real } X) - 8 \rceil$

by (subst powr-realpow) auto
 finally have $2^{\wedge} 7 \leq \text{real } X / 2^{\wedge} \text{nat } \lceil \log 2 (\text{real } X) - 8 \rceil$
 by simp
 hence exp-Z-gt-2-7: $(\int \omega. Z \omega \partial \Psi_1) \geq 2^{\wedge} 7$
 using a unfolding Z-def r-exp by simp

have var-Z-le-exp-Z: *measure-pmf.variance* $\Psi_1 Z \leq (\int \omega. Z \omega \partial \Psi_1)$
 unfolding Z-def by (intro r-var)

have ?L $\leq \text{measure } \Psi_1 \{f. \text{of-nat } (\text{Max } (f \text{ ' } A)) < \log 2 (\text{real } X) - 8\}$
 unfolding t-def by (intro pmf-mono) (auto simp add:int-of-nat-def)
 also have ... $\leq \text{measure } \Psi_1 \{f \in \text{space } \Psi_1. (\int \omega. Z \omega \partial \Psi_1) \leq |Z f - (\int \omega. Z \omega \partial \Psi_1)|\}$
 proof (rule pmf-mono)
 fix f assume f $\in \text{set-pmf } (\text{sample-pro } \Psi_1)$
 have fin-f-A: *finite* (f ' A) using fin-A finite-imageI by blast
 assume f $\in \{f. \text{real } (\text{Max } (f \text{ ' } A)) < \log 2 (\text{real } X) - 8\}$
 hence real (Max (f ' A)) $< \log 2 (\text{real } X) - 8$ by auto
 hence real (f a) $< \log 2 (\text{real } X) - 8$ if a $\in A$ for a
 using Max-ge[OF fin-f-A] imageI[OF that] order-less-le-trans by fastforce
 hence of-nat (f a) $< \lceil \log 2 (\text{real } X) - 8 \rceil$ if a $\in A$ for a
 using that by (subst less-ceiling-iff) auto
 hence f a $< \text{nat } \lceil \log 2 (\text{real } X) - 8 \rceil$ if a $\in A$ for a
 using that True by fastforce
 hence r (nat $\lceil \log 2 (\text{real } X) - 8 \rceil$) f = 0
 unfolding r-def card-eq-0-iff using not-less by auto
 hence Z f = 0
 unfolding Z-def by simp
 thus f $\in \{f \in \text{space } \Psi_1. (\int \omega. Z \omega \partial \Psi_1) \leq |Z f - (\int \omega. Z \omega \partial \Psi_1)|\}$
 by auto

qed
 also have ... $\leq \text{measure-pmf.variance } \Psi_1 Z / (\int \omega. Z \omega \partial \Psi_1)^{\wedge} 2$
 using exp-Z-gt-2-7 by (intro measure-pmf.second-moment-method) simp-all
 also have ... $\leq (\int \omega. Z \omega \partial \Psi_1) / (\int \omega. Z \omega \partial \Psi_1)^{\wedge} 2$
 by (intro divide-right-mono var-Z-le-exp-Z) simp
 also have ... = 1 / $(\int \omega. Z \omega \partial \Psi_1)$
 using exp-Z-gt-2-7 by (simp add:power2-eq-square)
 also have ... $\leq ?R$
 using exp-Z-gt-2-7 by (intro divide-left-mono) auto
 finally show ?thesis by simp

next
 case False
 have ?L $\leq \text{measure } \Psi_1 \{f. \text{of-nat } (\text{Max } (f \text{ ' } A)) < \log 2 (\text{real } X) - 8\}$
 unfolding t-def by (intro pmf-mono) (auto simp add:int-of-nat-def)
 also have ... $\leq \text{measure } \Psi_1 \{\}$
 using False by (intro pmf-mono) simp
 also have ... = 0
 by simp
 also have ... $\leq ?R$ by simp
 finally show ?thesis by simp

qed

lemma t-high:
measure $\Psi_1 \{f. \text{of-int } (t f) > \log 2 (\text{real } X) + 16 - b\text{-exp}\} \leq 1/2^{\wedge} 7$ (is ?L $\leq ?R$)
 proof -
 define Z :: (nat \Rightarrow nat) \Rightarrow real where Z = r (nat $\lfloor \log 2 (\text{real } X) + 8 \rfloor$)

 have Z-nonneg: Z f ≥ 0 for f
 unfolding Z-def r-def by simp

have $(\int \omega. Z \omega \partial \Psi_1) \leq \text{real } X / (2^{\wedge} \text{nat } \lfloor \log 2 (\text{real } X) + 8 \rfloor)$
unfolding *Z-def r-exp* **by** *simp*
also have $\dots \leq \text{real } X / (2^{\text{powr}} (\text{real } (\text{nat } \lfloor \log 2 (\text{real } X) + 8 \rfloor)))$
by *(subst powr-realpow) auto*
also have $\dots \leq \text{real } X / (2^{\text{powr}} \lfloor \log 2 (\text{real } X) + 8 \rfloor)$
by *(intro divide-left-mono powr-mono) auto*
also have $\dots \leq \text{real } X / (2^{\text{powr}} (\log 2 (\text{real } X) + 7))$
by *(intro divide-left-mono powr-mono, linarith) auto*
also have $\dots = \text{real } X / 2^{\text{powr}} (\log 2 (\text{real } X)) / 2^{\text{powr}} 7$
by *(subst powr-add) simp*
also have $\dots \leq 1/2^{\text{powr}} 7$
using *X-ge-1* **by** *(subst powr-log-cancel) auto*
finally have *Z-exp*: $(\int \omega. Z \omega \partial \Psi_1) \leq 1/2^{\wedge} 7$
by *simp*

have $?L \leq \text{measure } \Psi_1 \{f. \text{of-nat } (\text{Max } (f ' A)) > \log 2 (\text{real } X) + 7\}$
unfolding *t-def* **by** *(intro pmf-mono) (auto simp add:int-of-nat-def)*
also have $\dots \leq \text{measure } \Psi_1 \{f. Z f \geq 1\}$
proof *(rule pmf-mono)*
fix *f* **assume** $f \in \text{set-pmf } (\text{sample-pro } \Psi_1)$
assume $f \in \{f. \text{real } (\text{Max } (f ' A)) > \log 2 (\text{real } X) + 7\}$
hence $\text{real } (\text{Max } (f ' A)) > \log 2 (\text{real } X) + 7$ **by** *simp*
hence $\text{int } (\text{Max } (f ' A)) \geq \lfloor \log 2 (\text{real } X) + 8 \rfloor$
by *linarith*
hence $\text{Max } (f ' A) \geq \text{nat } \lfloor \log 2 (\text{real } X) + 8 \rfloor$
by *simp*
moreover have $f ' A \neq \{\}$ *finite* $(f ' A)$
using *fin-A finite-imageI A-nonempty* **by** *auto*
ultimately obtain *fa* **where** $fa \in f ' A$ $fa \geq \text{nat } \lfloor \log 2 (\text{real } X) + 8 \rfloor$
using *Max-in* **by** *auto*
then obtain *ae* **where** *ae-def*: $ae \in A$ $\text{nat } \lfloor \log 2 (\text{real } X) + 8 \rfloor \leq f ae$
by *auto*
hence $r (\text{nat } \lfloor \log 2 (\text{real } X) + 8 \rfloor) f > 0$
unfolding *r-def card-gt-0-iff* **using** *fin-A* **by** *auto*
hence $Z f \geq 1$
unfolding *Z-def* **by** *simp*
thus $f \in \{f. 1 \leq Z f\}$ **by** *simp*
qed
also have $\dots \leq (\int \omega. Z \omega \partial \Psi_1) / 1$
using *Z-nonneg* **by** *(intro pmf-markov) auto*
also have $\dots \leq ?R$
using *Z-exp* **by** *simp*
finally show *?thesis* **by** *simp*
qed

lemma *e-1*: $\text{measure } \Psi \{\psi. \neg E_1 \psi\} \leq 1/2^{\wedge} 6$

proof –

have $\text{measure } \Psi_1 \{f. 2^{\text{powr}} (\text{of-int } (-t f)) * \text{real } X \notin \{\text{real } b/2^{\wedge} 16 .. \text{real } b/2\}\} \leq$
 $\text{measure } \Psi_1 \{f. 2^{\text{powr}} (\text{of-int } (-t f)) * \text{real } X < \text{real } b/2^{\wedge} 16\} +$
 $\text{measure } \Psi_1 \{f. 2^{\text{powr}} (\text{of-int } (-t f)) * \text{real } X > \text{real } b/2\}$
by *(intro pmf-add) auto*
also have $\dots \leq \text{measure } \Psi_1 \{f. \text{of-int } (t f) > \log 2 X + 16 - b\text{-exp}\} +$
 $\text{measure } \Psi_1 \{f. \text{of-int } (t f) < \log 2 X + 1 - b\text{-exp}\}$
proof *(rule add-mono)*
show $\text{measure } \Psi_1 \{f. 2^{\text{powr}} (\text{of-int } (-t f)) * \text{real } X < \text{real } b/2^{\wedge} 16\} \leq$
 $\text{measure } \Psi_1 \{f. \text{of-int } (t f) > \log 2 X + 16 - b\text{-exp}\}$
proof *(rule pmf-mono)*

```

fix f assume f ∈ {f. 2 powr real-of-int (-t f) * real X < real b / 2 ^ 16}
hence 2 powr real-of-int (-t f) * real X < real b / 2 ^ 16
  by simp
hence log 2 (2 powr of-int (-t f) * real X) < log 2 (real b / 2 ^ 16)
  using b-min X-ge-1 by (intro iffD2[OF log-less-cancel-iff]) auto
hence of-int (-t f) + log 2 (real X) < log 2 (real b / 2 ^ 16)
  using X-ge-1 by (subst (asm) log-mult) auto
also have ... = real b-exp - log 2 (2 powr 16)
  unfolding b-def by (subst log-divide) auto
also have ... = real b-exp - 16
  by (subst log-powr-cancel) auto
finally have of-int (-t f) + log 2 (real X) < real b-exp - 16 by simp
thus f ∈ {f. of-int (t f) > log 2 (real X) + 16 - b-exp}
  by simp
qed
next
show measure Ψ1 {f. 2 powr of-int (-t f) * real X > real b/2} ≤
  measure Ψ1 {f. of-int (t f) < log 2 X + 1 - b-exp}
proof (rule pmf-mono)
fix f assume f ∈ {f. 2 powr real-of-int (-t f) * real X > real b / 2}
hence 2 powr real-of-int (-t f) * real X > real b / 2
  by simp
hence log 2 (2 powr of-int (-t f) * real X) > log 2 (real b / 2)
  using b-min X-ge-1 by (intro iffD2[OF log-less-cancel-iff]) auto
hence of-int (-t f) + log 2 (real X) > log 2 (real b / 2)
  using X-ge-1 by (subst (asm) log-mult) auto
hence of-int (-t f) + log 2 (real X) > real b-exp - 1
  unfolding b-def by (subst (asm) log-divide) auto
hence of-int (t f) < log 2 (real X) + 1 - b-exp
  by simp
thus f ∈ {f. of-int (t f) < log 2 (real X) + 1 - b-exp}
  by simp
qed
qed
also have ... ≤ 1/27 + 1/27
  by (intro add-mono t-low t-high)
also have ... = 1/26 by simp
finally have measure Ψ1 {f. 2 powr of-int (-t f) * real X ∉ {real b/216..real b/2}} ≤ 1/26
  by simp

thus ?thesis
  unfolding sample-pro-Ψ E1-def case-prod-beta
  by (subst pair-pmf-prob-left)
qed

definition E2 where E2 = (λ(f,g,h). |card (R f) - X / 2(s f)| ≤ ε/3 * X / 2(s f))

lemma e-2: measure Ψ {ψ. E1 ψ ∧ ¬E2 ψ} ≤ 1/26 (is ?L ≤ ?R)
proof -
define tm :: int where tm = [log 2 (real X)] + 16 - b-exp

have t-m-bound: tm ≤ [log 2 (real X)] - 10
  unfolding tm-def using b-exp-ge-26 by simp

have real b / 216 = (real X * (1/ X)) * (real b / 216)
  using X-ge-1 by simp
also have ... = (real X * 2 powr (-log 2 X)) * (real b / 216)
  using X-ge-1 by (subst powr-minus-divide) simp

```

also have $\dots \leq (\text{real } X * 2 \text{ powr } (- \lfloor \log 2 (\text{real } X) \rfloor)) * (2 \text{ powr } b\text{-exp} / 2^{16})$
unfolding *b-def* **using** *powr-realpow*
by (*intro mult-mono powr-mono*) *auto*
also have $\dots = \text{real } X * (2 \text{ powr } (- \lfloor \log 2 (\text{real } X) \rfloor) * 2 \text{ powr}(\text{real } b\text{-exp}-16))$
by (*subst powr-diff*) *simp*
also have $\dots = \text{real } X * 2 \text{ powr } (- \lfloor \log 2 (\text{real } X) \rfloor + (\text{int } b\text{-exp} - 16))$
by (*subst powr-add[symmetric]*) *simp*
also have $\dots = \text{real } X * 2 \text{ powr } (-t_m)$
unfolding *t_m-def* **by** (*simp add:algebra-simps*)
finally have $c:\text{real } b / 2^{16} \leq \text{real } X * 2 \text{ powr } (-t_m)$ **by** *simp*

define $T :: \text{nat set}$ **where** $T = \{x. (\text{real } X / 2^x \geq \text{real } b / 2^{16})\}$

have $x \in T \longleftrightarrow \text{int } x \leq t_m$ **for** x

proof –

have $x \in T \longleftrightarrow 2^x \leq \text{real } X * 2^{16} / b$
using *b-min* **by** (*simp add: field-simps T-def*)
also have $\dots \longleftrightarrow \log 2 (2^x) \leq \log 2 (\text{real } X * 2^{16} / b)$
using *X-ge-1 b-min* **by** (*intro log-le-cancel-iff[symmetric] divide-pos-pos*) *auto*
also have $\dots \longleftrightarrow x \leq \log 2 (\text{real } X * 2^{16}) - \log 2 b$
using *X-ge-1 b-min* **by** (*subst log-divide*) *auto*
also have $\dots \longleftrightarrow x \leq \log 2 (\text{real } X) + \log 2 (2 \text{ powr } 16) - b\text{-exp}$
unfolding *b-def* **using** *X-ge-1* **by** (*subst log-mult*) *auto*
also have $\dots \longleftrightarrow x \leq \lfloor \log 2 (\text{real } X) + \log 2 (2 \text{ powr } 16) - b\text{-exp} \rfloor$
by *linarith*
also have $\dots \longleftrightarrow x \leq \lfloor \log 2 (\text{real } X) + 16 - \text{real-of-int } (\text{int } b\text{-exp}) \rfloor$
by (*subst log-powr-cancel*) *auto*
also have $\dots \longleftrightarrow x \leq t_m$
unfolding *t_m-def* **by** *linarith*
finally show *?thesis* **by** *simp*

qed

hence $T\text{-eq}: T = \{x. \text{int } x \leq t_m\}$ **by** *auto*

have $T = \{x. \text{int } x < t_m + 1\}$

unfolding $T\text{-eq}$ **by** *simp*

also have $\dots = \{x. x < \text{nat } (t_m + 1)\}$

unfolding *zless-nat-eq-int-zless* **by** *simp*

finally have $T\text{-eq-2}: T = \{x. x < \text{nat } (t_m + 1)\}$

by *simp*

have *inj-1: inj-on* ($(-) (\text{nat } t_m)$) T

unfolding $T\text{-eq}$ **by** (*intro inj-onI*) *simp*

have *fin-T: finite* T

unfolding $T\text{-eq-2}$ **by** *simp*

have *r-exp: $(\int \omega. \text{real } (r \ t \ \omega) \ \partial\Psi_1) = \text{real } X / 2^t$* **if** $t \in T$ **for** t

proof –

have $t \leq t_m$

using *that* **unfolding** $T\text{-eq}$ **by** *simp*

also have $\dots \leq \lfloor \log 2 (\text{real } X) \rfloor - 10$

using *t-m-bound* **by** *simp*

also have $\dots \leq \lfloor \log 2 (\text{real } X) \rfloor$

by *simp*

also have $\dots \leq \lfloor \log 2 (\text{real } n) \rfloor$

using *X-le-n X-ge-1* **by** (*intro floor-mono log-mono*) *auto*

also have $\dots \leq \lceil \log 2 (\text{real } n) \rceil$

by *simp*

finally have $t \leq \lceil \log 2 (\text{real } n) \rceil$ **by** *simp*

hence $t \leq \max(\text{nat } \lceil \log 2(\text{real } n) \rceil)$ 1by simp
 thus ?thesis
 unfolding r-exp by simp
 qed

have r-var: measure-pmf.variance $\Psi_1(\lambda\omega. \text{real}(r\ t\ \omega)) \leq \text{real } X / 2^t$ if $t \in T$ for t
 using r-exp[OF that] r-var by metis

have $g = C_4 / \varepsilon^2 * \varepsilon^{2/2^23}$
 using ε -gt-0 by (simp add:C4-def)
 also have $\dots = 2^{\text{powr}(\log 2(C_4 / \varepsilon^2))} * \varepsilon^{2/2^23}$
 using ε -gt-0 C4-def by (subst powr-log-cancel) auto
 also have $\dots \leq 2^{\text{powr } b\text{-exp}} * \varepsilon^{2/2^23}$
 unfolding b-exp-def
 by (intro divide-right-mono mult-right-mono powr-mono, linarith) auto
 also have $\dots = b * \varepsilon^{2/2^23}$
 using powr-realpow unfolding b-def by simp
 also have $\dots = (b/2^{16}) * (\varepsilon^{2/2^7})$
 by simp
 also have $\dots \leq (X * 2^{\text{powr}(-t_m)}) * (\varepsilon^{2/2^7})$
 by (intro mult-mono c) auto
 also have $\dots = X * (2^{\text{powr}(-t_m)} * 2^{\text{powr}(-7)}) * \varepsilon^2$
 using powr-realpow by simp
 also have $\dots = 2^{\text{powr}(-t_m-7)} * (\varepsilon^2 * X)$
 by (subst powr-add[symmetric]) (simp)
 finally have $g \leq 2^{\text{powr}(-t_m-7)} * (\varepsilon^2 * X)$ by simp
 hence b: $g / (\varepsilon^2 * X) \leq 2^{\text{powr}(-t_m-7)}$
 using ε -gt-0 X-ge-1
 by (subst pos-divide-le-eq) auto

have a: measure $\Psi_1 \{f. |\text{real}(r\ t\ f) - \text{real } X / 2^t| > \varepsilon/3 * \text{real } X / 2^t\} \leq 2^{\text{powr}(\text{real } t - t_m - 7)}$
 (is ?L1 \leq ?R1) if $t \in T$ for t

proof -

have ?L1 $\leq \mathcal{P}(f \text{ in } \Psi_1. |\text{real}(r\ t\ f) - \text{real } X / 2^t| \geq \varepsilon/3 * \text{real } X / 2^t)$
 by (intro pmf-mono) auto
 also have $\dots = \mathcal{P}(f \text{ in } \Psi_1. |\text{real}(r\ t\ f) - (\int \omega. \text{real}(r\ t\ \omega) \partial \Psi_1)| \geq \varepsilon/3 * \text{real } X / 2^t)$
 by (simp add: r-exp[OF that])
 also have $\dots \leq \text{measure-pmf.variance } \Psi_1(\lambda\omega. \text{real}(r\ t\ \omega)) / (\varepsilon/3 * \text{real } X / 2^t)^2$
 using X-ge-1 ε -gt-0
 by (intro measure-pmf.Chebyshev-inequality divide-pos-pos mult-pos-pos) auto
 also have $\dots \leq (X / 2^t) / (\varepsilon/3 * X / 2^t)^2$
 by (intro divide-right-mono r-var[OF that]) simp
 also have $\dots = 2^t * (9 / (\varepsilon^2 * X))$
 by (simp add:power2-eq-square algebra-simps)
 also have $\dots \leq 2^t * (2^{\text{powr}(-t_m-7)})$
 by (intro mult-left-mono b) simp
 also have $\dots = 2^{\text{powr } t} * 2^{\text{powr}(-t_m-7)}$
 by (subst powr-realpow[symmetric]) auto
 also have $\dots = ?R1$
 by (subst powr-add[symmetric]) (simp add:algebra-simps)
 finally show ?L1 \leq ?R1 by simp
 qed

have $\exists y < \text{nat}(t_m + 1). x = \text{nat } t_m - y$ if $x < \text{nat}(t_m + 1)$ for x
 using that by (intro exI[where $x = \text{nat } t_m - x$]) simp
 hence T-reindex: $(-)(\text{nat } t_m) \{x. x < \text{nat}(t_m + 1)\} = \{.. < \text{nat}(t_m + 1)\}$
 by (auto simp add: set-eq-iff image-iff)

have $?L \leq \text{measure } \Psi \{ \psi. (\exists t \in T. |\text{real } (r \ t \ (fst \ \psi)) - \text{real } X / 2^t| > \varepsilon / 3 * \text{real } X / 2^t) \}$
proof (*rule pmf-mono*)
fix ψ
assume $\psi \in \text{set-pmf } (\text{sample-pro } \Psi)$
obtain $f \ g \ h$ **where** $\psi\text{-def: } \psi = (f, g, h)$ **by** (*metis prod-cases3*)
assume $\psi \in \{ \psi. E_1 \ \psi \wedge \neg E_2 \ \psi \}$
hence $a: 2 \text{ powr } (-\text{real-of-int } (t \ f)) * \text{real } X \in \{ \text{real } b / 2^{16} .. \text{real } b / 2 \}$ **and**
 $b: |\text{card } (R \ f) - \text{real } X / 2^{(s \ f)}| > \varepsilon / 3 * X / 2^{(s \ f)}$
unfolding $E_1\text{-def } E_2\text{-def}$ **by** (*auto simp add: \psi-def*)
have $|\text{card } (R \ f) - X / 2^{(s \ f)}| = 0$ **if** $s \ f = 0$
using *that* **by** (*simp add: R-def X-def*)
moreover **have** $(\varepsilon / 3) * (X / 2^{s \ f}) \geq 0$
using $\varepsilon\text{-gt-0 } X\text{-ge-1}$ **by** (*intro mult-nonneg-nonneg*) *auto*
ultimately **have** *False* **if** $s \ f = 0$
using *b that* **by** *simp*
hence $s \ f > 0$ **by** *auto*
hence $t \ f = s \ f$ **unfolding** *s-def* **by** *simp*
hence $2 \text{ powr } (-\text{real } (s \ f)) * X \geq b / 2^{16}$
using *a* **by** *simp*
hence $X / 2 \text{ powr } (\text{real } (s \ f)) \geq b / 2^{16}$
by (*simp add: divide-powr-uminus mult.commute*)
hence $\text{real } X / 2^{(s \ f)} \geq b / 2^{16}$
by (*subst (asm) powr-realpow, auto*)
hence $s \ f \in T$ **unfolding** *T-def* **by** *simp*
moreover **have** $|r \ (s \ f) \ f - X / 2^{s \ f}| > \varepsilon / 3 * X / 2^{s \ f}$
using *R-def r-def b* **by** *simp*
ultimately **have** $\exists t \in T. |r \ t \ (fst \ \psi) - X / 2^t| > \varepsilon / 3 * X / 2^t$
using $\psi\text{-def}$ **by** (*intro bexI[where x=s f]*) *simp*
thus $\psi \in \{ \psi. (\exists t \in T. |r \ t \ (fst \ \psi) - X / 2^t| > \varepsilon / 3 * X / 2^t) \}$ **by** *simp*
qed
also **have** $\dots = \text{measure } \Psi_1 \{ f. (\exists t \in T. |\text{real } (r \ t \ f) - \text{real } X / 2^t| > \varepsilon / 3 * \text{real } X / 2^t) \}$
unfolding *sample-pro-Ψ* **by** (*intro pair-pmf-prob-left*)
also **have** $\dots = \text{measure } \Psi_1 (\bigcup t \in T. \{ f. |\text{real } (r \ t \ f) - \text{real } X / 2^t| > \varepsilon / 3 * \text{real } X / 2^t \})$
by (*intro measure-pmf-cong*) *auto*
also **have** $\dots \leq (\sum t \in T. \text{measure } \Psi_1 \{ f. |\text{real } (r \ t \ f) - \text{real } X / 2^t| > \varepsilon / 3 * \text{real } X / 2^t \})$
by (*intro measure-UNION-le fin-T*) (*simp*)
also **have** $\dots \leq (\sum t \in T. 2 \text{ powr } (\text{real } t - \text{of-int } t_m - 7))$
by (*intro sum-mono a*)
also **have** $\dots = (\sum t \in T. 2 \text{ powr } (-\text{int } (\text{nat } t_m - t) - 7))$
unfolding *T-eq*
by (*intro sum.cong refl arg-cong2[where f=(powr)]*) *simp*
also **have** $\dots = (\sum x \in (\lambda x. \text{nat } t_m - x) ' T. 2 \text{ powr } (-\text{real } x - 7))$
by (*subst sum.reindex[OF inj-1]*) *simp*
also **have** $\dots = (\sum x \in (\lambda x. \text{nat } t_m - x) ' T. 2 \text{ powr } (-7) * 2 \text{ powr } (-\text{real } x))$
by (*subst powr-add[symmetric]*) (*simp add: algebra-simps*)
also **have** $\dots = 1 / 2^7 * (\sum x \in (\lambda x. \text{nat } t_m - x) ' T. 2 \text{ powr } (-\text{real } x))$
by (*subst sum-distrib-left*) *simp*
also **have** $\dots = 1 / 2^7 * (\sum x < \text{nat } (t_m + 1). 2 \text{ powr } (-\text{real } x))$
unfolding *T-eq-2 T-reindex*
by (*intro arg-cong2[where f=(*)] sum.cong*) *auto*
also **have** $\dots = 1 / 2^7 * (\sum x < \text{nat } (t_m + 1). (2 \text{ powr } (-1)) \text{ powr } (\text{real } x))$
by (*subst powr-powr*) *simp*
also **have** $\dots = 1 / 2^7 * (\sum x < \text{nat } (t_m + 1). (1 / 2)^x)$
using *powr-realpow* **by** *simp*
also **have** $\dots \leq 1 / 2^7 * 2$
by (*subst geometric-sum*) *auto*
also **have** $\dots = 1 / 2^6$ **by** *simp*
finally **show** *?thesis* **by** *simp*

qed

definition E_3 where $E_3 = (\lambda(f,g,h). \text{inj-on } g (R f))$

lemma *R-bound*:

fixes $f g h$
assumes $E_1 (f,g,h)$
assumes $E_2 (f,g,h)$
shows $\text{card } (R f) \leq 2/3 * b$

proof –

have $\text{real } (\text{card } (R f)) \leq (\varepsilon / 3) * (\text{real } X / 2^s f) + \text{real } X / 2^s f$
using *assms(2) unfolding E_2 -def by simp*
also have $\dots \leq (1/3) * (\text{real } X / 2^s f) + \text{real } X / 2^s f$
using ε -lt-1 by (*intro add-mono mult-right-mono*) auto
also have $\dots = (4/3) * (\text{real } X / 2^{\text{pow } s} f)$
using *pow-realpow by simp*
also have $\dots \leq (4/3) * (\text{real } X / 2^{\text{pow } t} f)$
unfolding *s-def*
by (*intro mult-left-mono divide-left-mono pow-mono*) auto
also have $\dots = (4/3) * (2^{\text{pow } (-\text{of-int } (t f))} * \text{real } X)$
by (*subst pow-minus-divide*) simp
also have $\dots = (4/3) * (2^{\text{pow } (-t f)} * \text{real } X)$
by *simp*
also have $\dots \leq (4/3) * (b/2)$
using *assms(1) unfolding E_1 -def*
by (*intro mult-left-mono*) auto
also have $\dots \leq (2/3) * b$ by *simp*
finally show *?thesis* by *simp*

qed

lemma *e-3*: $\text{measure } \Psi \{ \psi. E_1 \psi \wedge E_2 \psi \wedge \neg E_3 \psi \} \leq 1/2^6$ (is $?L \leq ?R$)

proof –

let $? \alpha = (\lambda(z,x,y) f. z < C_7 * b^2 \wedge x \in R f \wedge y \in R f \wedge x < y)$
let $? \beta = (\lambda(z,x,y) g. g x = z \wedge g y = z)$

have β -prob: $\text{measure } \Psi_2 \{ g. ? \beta \omega g \} \leq (1/\text{real } (C_7 * b^2)^2)$
if $? \alpha \omega f$ for ωf

proof –

obtain $x y z$ where ω -def: $\omega = (z,x,y)$ by (*metis prod-cases3*)
have a : *prob-space.indep-vars* Ψ_2 ($\lambda i. \text{discrete}$) ($\lambda x \omega. \omega x = z$) I
if $I \subseteq \{..<n\}$ $\text{card } I \leq 2$ for I
by (*intro prob-space.indep-vars-compose2[OF - hash-pro-indep[OF Ψ_2]] that*)
(*simp-all add:prob-space-measure-pmf*)

have $u \in R f \implies u < n$ for u
unfolding *R-def* using *A-range* by *auto*
hence b : $x < n y < n \text{card } \{x, y\} = 2$
using *that* ω -def by *auto*
have c : $z < C_7 * b^2$ using ω -def *that* by *simp*

have $\text{measure } \Psi_2 \{ g. ? \beta \omega g \} = \text{measure } \Psi_2 \{ g. (\forall \xi \in \{x,y\}. g \xi = z) \}$
by (*simp add: ω -def*)
also have $\dots = (\prod \xi \in \{x,y\}. \text{measure } \Psi_2 \{ g. g \xi = z \})$
using b by (*intro measure-pmf.split-indep-events[OF refl, where $I=\{x,y\}$] a*)
(*simp-all add:prob-space-measure-pmf*)
also have $\dots = (\prod \xi \in \{x,y\}. \text{measure } (\text{map-pmf } (\lambda \omega. \omega \xi) (\text{sample-pro } \Psi_2))) \{ g. g = z \}$
by (*simp add:vimage-def*)
also have $\dots = (\prod \xi \in \{x,y\}. \text{measure } (\mathcal{N} (C_7 * b^2))) \{ g. g = z \}$

using b *hash-pro-component*[$OF \Psi_2$] **by** (*intro prod.cong*) *fastforce+*
also have $\dots = (\prod \xi \in \{x,y\}. \text{measure } (\text{pmf-of-set } \{..<C_7 * b^2\}) \{z\})$
by (*subst nat-pro*) (*simp-all add:C7-def b-def*)
also have $\dots = (\text{measure } (\text{pmf-of-set } \{..<C_7 * b^2\}) \{z\})^{\wedge 2}$
using b **by** *simp*
also have $\dots \leq (1 / (C_7 * b^2))^{\wedge 2}$
using c **by** (*subst measure-pmf-of-set*) *auto*
also have $\dots = (1 / (C_7 * b^2))^{\wedge 2}$
by (*simp add:algebra-simps power2-eq-square*)
finally show *?thesis* **by** *simp*
qed

have α -*card*: $\text{card } \{\omega. ?\alpha \omega f\} \leq (C_7 * b^{\wedge 2}) * (\text{card } (R f) * (\text{card } (R f) - 1) / 2)$
(is ?TL ≤ ?TR) **and** *fin-α*: *finite* $\{\omega. ?\alpha \omega f\}$ **(is ?T2)** **for** f
proof –
have $t1$: $\{\omega. ?\alpha \omega f\} \subseteq \{..<C_7 * b^{\wedge 2}\} \times \{(x,y) \in R f \times R f. x < y\}$
by (*intro subsetI*) *auto*
moreover have $\text{card } (\{..<C_7 * b^{\wedge 2}\} \times \{(x,y) \in R f \times R f. x < y\}) = ?TR$
using *card-ordered-pairs* [**where** $M=R f$]
by (*simp add: card-cartesian-product*)
moreover have *finite* $(R f)$
unfolding *R-def* **using** *fin-A finite-subset* **by** *simp*
hence *finite* $\{(x, y). (x, y) \in R f \times R f \wedge x < y\}$
by (*intro finite-subset* [**where** $B=R f \times R f$, *OF - finite-cartesian-product*]) *auto*
hence $t2$: *finite* $(\{..<C_7 * b^{\wedge 2}\} \times \{(x,y) \in R f \times R f. x < y\})$
by (*intro finite-cartesian-product*) *auto*
ultimately show $?TL \leq ?TR$
using *card-mono of-nat-le-iff* **by** (*metis (no-types, lifting)*)
show $?T2$
using *finite-subset* [*OF t1 t2*] **by** *simp*
qed

have $?L \leq \text{measure } \Psi \{(f,g,h). \text{card } (R f) \leq b \wedge (\exists x y z. ?\alpha (x,y,z) f \wedge ?\beta (x,y,z) g)\}$
proof (*rule pmf-mono*)
fix ψ **assume** $b:\psi \in \text{set-pmf } (\text{sample-pro } \Psi)$
obtain $f g h$ **where** ψ -*def*: $\psi = (f,g,h)$ **by** (*metis prod-cases3*)
have $(f,g,h) \in \text{pro-set } \Psi$ **using** b ψ -*def* **by** *simp*
hence $c:g x < C_7 * b^{\wedge 2}$ **for** x
using *g-range* **by** *simp*

assume $a:\psi \in \{\psi. E_1 \psi \wedge E_2 \psi \wedge \neg E_3 \psi\}$
hence $\text{card } (R f) \leq 2/3 * b$
using *R-bound* ψ -*def* **by** *force*
moreover have $\exists a b. a \in R f \wedge b \in R f \wedge a \neq b \wedge g a = g b$
using a **unfolding** ψ -*def* E_3 -*def inj-on-def* **by** *auto*
hence $\exists x y. x \in R f \wedge y \in R f \wedge x < y \wedge g x = g y$
by (*metis not-less-iff-gr-or-eq*)
hence $\exists x y z. ?\alpha (x,y,z) f \wedge ?\beta (x,y,z) g$
using c **by** *blast*
ultimately show $\psi \in \{(f, g, h). \text{card } (R f) \leq b \wedge (\exists x y z. ?\alpha (x,y,z) f \wedge ?\beta (x,y,z) g)\}$
unfolding ψ -*def* **by** *auto*
qed

also have $\dots = (\int f. \text{measure } (\text{pair-pmf } \Psi_2 \Psi_3) \{g. \text{card } (R f) \leq b \wedge (\exists x y z. ?\alpha (x,y,z) f \wedge ?\beta (x,y,z) (fst g))\}) \partial \Psi_1$
unfolding *sample-pro-Ψ split-pair-pmf* **by** (*simp add: case-prod-beta*)
also have
 $\dots = (\int f. \text{measure } \Psi_2 \{g. \text{card } (R f) \leq b \wedge (\exists x y z. ?\alpha (x,y,z) f \wedge ?\beta (x,y,z) g)\}) \partial \Psi_1$
by (*subst pair-pmf-prob-left*) *simp*

also have ... $\leq (\int f. 1/\text{real } (2 * C_7) \partial \Psi_1)$
proof (*rule pmf-exp-mono*[*OF integrable-sample-pro integrable-sample-pro*])
fix f **assume** $f \in \text{set-pmf } (\text{sample-pro } \Psi_1)$
show $\text{measure } \Psi_2 \{g. \text{card } (R f) \leq b \wedge (\exists x y z. ?\alpha (x,y,z) f \wedge ?\beta (x,y,z) g)\} \leq 1 / \text{real } (2$
 $* C_7)$
(is $?L1 \leq ?R1)$
proof (*cases card* $(R f) \leq b$)
case *True*
have $?L1 \leq \text{measure } \Psi_2 (\bigcup \omega \in \{\omega. ?\alpha \omega f\}. \{g. ?\beta \omega g\})$
by (*intro pmf-mono*) *auto*
also have ... $\leq (\sum \omega \in \{\omega. ?\alpha \omega f\}. \text{measure } \Psi_2 \{g. ?\beta \omega g\})$
by (*intro measure-UNION-le fin- α*) *auto*
also have ... $\leq (\sum \omega \in \{\omega. ?\alpha \omega f\}. (1/\text{real } (C_7 * b^2)^2))$
by (*intro sum-mono β -prob*) *auto*
also have ... $= \text{card } \{\omega. ?\alpha \omega f\} / (C_7 * b^2)^2$
by *simp*
also have ... $\leq (C_7 * b^2) * (\text{card } (R f) * (\text{card } (R f) - 1) / 2) / (C_7 * b^2)^2$
by (*intro α -card divide-right-mono*) *simp*
also have ... $\leq (C_7 * b^2) * (b * b / 2) / (C_7 * b^2)^2$
unfolding $C_7\text{-def}$ **using** *True*
by (*intro divide-right-mono Nat.of-nat-mono mult-mono*) *auto*
also have ... $= 1 / (2 * C_7)$
using *b-min* **by** (*simp add: algebra-simps power2-eq-square*)
finally show $?thesis$ **by** *simp*
next
case *False*
then show $?thesis$ **by** *simp*
qed
qed
also have ... $\leq 1 / 2^6$
unfolding $C_7\text{-def}$ **by** *simp*
finally show $?thesis$ **by** *simp*
qed

definition E_4 **where** $E_4 = (\lambda(f,g,h). |p(f,g,h) - q(\text{card } (R f))| \leq \varepsilon / 12 * \text{card } (R f))$

lemma *e-4-h*: $9 / \text{sqrt } b \leq \varepsilon / 12$

proof –

have $108 \leq \text{sqrt } (C_4)$
unfolding $C_4\text{-def}$ **by** (*approximation 5*)
also have ... $\leq \text{sqrt}(\varepsilon^2 * \text{real } b)$
using *b-lower-bound ε -gt-0*
by (*intro real-sqrt-le-mono*) (*simp add: pos-divide-le-eq algebra-simps*)
also have ... $= \varepsilon * \text{sqrt } b$
using *ε -gt-0* **by** (*simp add: real-sqrt-mult*)
finally have $108 \leq \varepsilon * \text{sqrt } b$ **by** *simp*
thus $?thesis$
using *b-min* **by** (*simp add: pos-divide-le-eq*)
qed

lemma *e-4*: $\text{measure } \Psi \{\psi. E_1 \psi \wedge E_2 \psi \wedge E_3 \psi \wedge \neg E_4 \psi\} \leq 1 / 2^6$ (**is** $?L \leq ?R$)

proof –

have $a: \text{measure } \Psi_3 \{h. E_1(f,g,h) \wedge E_2(f,g,h) \wedge E_3(f,g,h) \wedge \neg E_4(f,g,h)\} \leq 1 / 2^6$
(is $?L1 \leq ?R1)$ **if** $f \in \text{set-pmf } (\text{sample-pro } \Psi_1)$ $g \in \text{set-pmf}(\text{sample-pro } \Psi_2)$
for $f g$
proof (*cases card* $(R f) \leq b \wedge \text{inj-on } g (R f)$)
case *True*

have $g\text{-inj}$: $\text{inj-on } g \ (R \ f)$
using True **by** simp

have fin-R : $\text{finite } (g \ ' \ R \ f)$
unfolding $R\text{-def}$ **using** fin-A
by $(\text{intro } \text{finite-imageI}) \ \text{simp}$

interpret B : $\text{balls-and-bins-abs } g \ ' \ R \ f \ \{..<b\}$
using $\text{fin-R } b\text{-ne}$ **by** $\text{unfold-locales auto}$

have $\text{range } g \subseteq \{..<C_7 * b^2\}$
using $g\text{-range-1 that(2)}$ **by** auto
hence $g\text{-ran}$: $g \ ' \ R \ f \subseteq \{..<C_7 * b^2\}$
by auto

have $\text{sample-pro } (\mathcal{N} \ b) = \text{pmf-of-set } \{..<b\}$
by $(\text{intro } \text{nat-pro}) \ (\text{simp } \text{add:b-def})$
hence $\text{map-pmf } (\lambda\omega. \omega \ x) \ (\text{sample-pro } (\mathcal{H} \ k \ (C_7 * b^2) \ (\mathcal{N} \ b))) = \text{pmf-of-set } \{..<b\}$
if $x \in g \ ' \ R \ f$ **for** x
using $g\text{-ran } \text{hash-pro-component}[OF \ \Psi_3 \ - \ k\text{-gt-0}]$ **that** **by** auto
moreover **have** $\text{prob-space.k-wise-indep-vars } \Psi_3 \ k \ (\lambda-. \ \text{discrete}) \ (\lambda x \ \omega. \ \omega \ x) \ (g \ ' \ R \ f)$
by $(\text{intro } \text{prob-space.k-wise-indep-subset}[OF \ - \ - \ \text{hash-pro-k-indep}[OF \ \Psi_3]]) \ g\text{-ran}$
 $\text{prob-space-measure-pmf}$
ultimately **have** $\text{lim-balls-and-bins}$: $B.\text{lim-balls-and-bins } k \ (\text{sample-pro } (\mathcal{H} \ k \ (C_7 * b^2) \ (\mathcal{N} \ b)))$
unfolding $B.\text{lim-balls-and-bins-def}$ **by** auto

have card-g-R : $\text{card } (g \ ' \ R \ f) = \text{card } (R \ f)$
using $\text{True } \text{card-image}$ **by** auto
hence $b\text{-mu}$: $\rho \ (\text{card } (R \ f)) = B.\mu$
unfolding $B.\mu\text{-def } \rho\text{-def}$ **using** $b\text{-min}$ **by** $(\text{simp } \text{add:powr-realpow})$

have card-g-le-b : $\text{card } (g \ ' \ R \ f) \leq \text{card } \{..<b\}$
unfolding card-g-R **using** True **by** simp

have $?L1 \leq \text{measure } \Psi_3 \ \{h. |B.Y \ h - B.\mu| > 9 * \text{real } (\text{card } (g \ ' \ R \ f)) / \text{sqrt } (\text{card } \{..<b\})\}$
proof $(\text{rule } \text{pmf-mono})$
fix h **assume** $h \in \{h. E_1 \ (f,g,h) \wedge E_2 \ (f,g,h) \wedge E_3 \ (f,g,h) \wedge \neg E_4 \ (f,g,h)\}$
hence b : $|p \ (f,g,h) - \rho \ (\text{card } (R \ f))| > \varepsilon/12 * \text{card } (R \ f)$
unfolding $E_4\text{-def}$ **by** simp
assume $h \in \text{set-pmf } (\text{sample-pro } \Psi_3)$
hence $h\text{-range}$: $h \ x < b$ **for** x **using** $h\text{-range-1}$ **by** simp

have $\{j \in \{..<b\}. \text{int } (s \ f) \leq \tau_1 \ (f, \ g, \ h) \ A \ 0 \ j\} =$
 $\{j \in \{..<b\}. \text{int } (s \ f) \leq \text{max } (\text{Max } (\{\text{int } (f \ a) \mid a. a \in A \wedge h \ (g \ a) = j\} \cup \{-1\})) \ (- \ 1)\}$
unfolding $\tau_1\text{-def}$ **by** simp
also **have** $\dots = \{j \in \{..<b\}. \text{int } (s \ f) \leq \text{Max } (\{\text{int } (f \ a) \mid a. a \in A \wedge h \ (g \ a) = j\} \cup \{-1\})\}$
using fin-A **by** $(\text{subst } \text{max-absorb1}) \ (\text{auto } \text{intro: } \text{Max-ge})$
also **have** $\dots = \{j \in \{..<b\}. (\exists a \in R \ f. h \ (g \ a) = j)\}$
unfolding $R\text{-def}$ **using** fin-A **by** $(\text{subst } \text{Max-ge-iff}) \ \text{auto}$
also **have** $\dots = \{j. \exists a \in R \ f. h \ (g \ a) = j\}$
using $h\text{-range}$ **by** auto
also **have** $\dots = (h \circ g) \ ' \ (R \ f)$
by $(\text{auto } \text{simp } \text{add:set-eq-iff } \text{image-iff})$
also **have** $\dots = h \ ' \ (g \ ' \ (R \ f))$
by $(\text{simp } \text{add:image-image})$
finally **have** c : $\{j \in \{..<b\}. \text{int } (s \ f) \leq \tau_1 \ (f, \ g, \ h) \ A \ 0 \ j\} = h \ ' \ (g \ ' \ (R \ f))$
by simp

have $9 * \text{real} (\text{card} (g \text{ ‘ } R f)) / \text{sqrt} (\text{card} \{..<b\}) = 9 / \text{sqrt} b * \text{real} (\text{card} (R f))$
using *card-image[OF g-inj]* **by** *simp*
also have $... \leq \varepsilon / 12 * \text{card} (R f)$
by (*intro mult-right-mono e-4-h*) *simp*
also have $... < |B.Y h - B.\mu|$
using *b c unfolding B.Y-def p-def b-mu* **by** *simp*
finally show $h \in \{h. |B.Y h - B.\mu| > 9 * \text{real} (\text{card} (g \text{ ‘ } R f)) / \text{sqrt} (\text{card} \{..<b\})\}$
by *simp*
qed
also have $... \leq 1/2^6$
using *k-min*
by (*intro B.deviation-bound[OF card-g-le-b lim-balls-and-bins]*) *auto*
finally show *?thesis* **by** *simp*
next
case *False*
have $?L1 \leq \text{measure } \Psi_3 \{ \}$
proof (*rule pmf-mono*)
fix *h* **assume** $b: h \in \{h. E_1 (f, g, h) \wedge E_2 (f, g, h) \wedge E_3 (f, g, h) \wedge \neg E_4 (f, g, h)\}$
hence $\text{card} (R f) \leq (2/3)*b$
by (*auto intro!: R-bound[simplified]*)
hence $\text{card} (R f) \leq b$
by *simp*
moreover have *inj-on g (R f)*
using *b* **by** (*simp add:E3-def*)
ultimately have *False* **using** *False* **by** *simp*
thus $h \in \{ \}$ **by** *simp*
qed
also have $... = 0$ **by** *simp*
finally show *?thesis* **by** *simp*
qed
have $?L = (\int f. (\int g. \text{measure } \Psi_3 \{h. E_1 (f,g,h) \wedge E_2 (f,g,h) \wedge E_3 (f,g,h) \wedge \neg E_4 (f,g,h)\} \partial\Psi_2) \partial\Psi_1)$
unfolding *sample-pro-Ψ split-pair-pmf* **by** *simp*
also have $... \leq (\int f. (\int g. 1/2^6 \partial\Psi_2) \partial\Psi_1)$
using *a* **by** (*intro integral-mono-AE AE-pmfI*) *simp-all*
also have $... = 1/2^6$
by *simp*
finally show *?thesis* **by** *simp*
qed
lemma *ρ-inverse*: $\rho\text{-inv} (\rho x) = x$
proof –
have $a: 1 - 1/b \neq 0$
using *b-min* **by** *simp*
have $\rho x = b * (1 - (1 - 1/b) \text{powr } x)$
unfolding *ρ-def* **by** *simp*
hence $\rho x / \text{real } b = 1 - (1 - 1/b) \text{powr } x$ **by** *simp*
hence $\ln (1 - \rho x / \text{real } b) = \ln ((1 - 1/b) \text{powr } x)$ **by** *simp*
also have $... = x * \ln (1 - 1/b)$
using *a* **by** (*intro ln-powr*)
finally have $\ln (1 - \rho x / \text{real } b) = x * \ln (1 - 1/b)$
by *simp*
moreover have $\ln (1 - 1/b) < 0$
using *b-min* **by** (*subst ln-less-zero-iff*) *auto*
ultimately show *?thesis*
using *ρ-inv-def* **by** *simp*

qed

lemma *rho-mono*:

assumes $x \leq y$
shows $\varrho x \leq \varrho y$

proof –

have $(1 - 1 / \text{real } b) \text{ powr } y \leq (1 - 1 / \text{real } b) \text{ powr } x$
using *b-min*
by (*intro powr-mono-rev assms*) auto
thus ?thesis
unfolding ϱ -def by (*intro mult-left-mono*) auto

qed

lemma *rho-two-thirds*: $\varrho (2/3 * b) \leq 3/5 * b$

proof –

have $1/3 \leq \exp (- 13 / 12::\text{real})$
by (*approximation 8*)
also have $\dots \leq \exp (- 1 - 2 / \text{real } b)$
using *b-min* by (*intro iffD2[OF exp-le-cancel-iff]*) (*simp add: algebra-simps*)
also have $\dots \leq \exp (b * (-(1/\text{real } b) - 2*(1/\text{real } b)^2))$
using *b-min* by (*simp add: algebra-simps power2-eq-square*)
also have $\dots \leq \exp (b * \ln (1 - 1/\text{real } b))$
using *b-min*
by (*intro iffD2[OF exp-le-cancel-iff] mult-left-mono ln-one-minus-pos-lower-bound*) auto
also have $\dots = \exp (\ln ((1 - 1/\text{real } b) \text{ powr } b))$
using *b-min* by (*subst ln-powr*) auto
also have $\dots = (1 - 1/\text{real } b) \text{ powr } b$
using *b-min* by (*subst exp-ln*) auto
finally have $a: 1/3 \leq (1 - 1/\text{real } b) \text{ powr } b$ by *simp*

have $2/5 \leq (1/3) \text{ powr } (2/3::\text{real})$
by (*approximation 5*)
also have $\dots \leq ((1 - 1/\text{real } b) \text{ powr } b) \text{ powr } (2/3)$
by (*intro powr-mono2 a*) auto
also have $\dots = (1 - 1/\text{real } b) \text{ powr } (2/3 * \text{real } b)$
by (*subst powr-powr*) (*simp add: algebra-simps*)
finally have $2/5 \leq (1 - 1 / \text{real } b) \text{ powr } (2 / 3 * \text{real } b)$ by *simp*
hence $1 - (1 - 1 / \text{real } b) \text{ powr } (2 / 3 * \text{real } b) \leq 3/5$
by *simp*
hence $\varrho (2/3 * b) \leq b * (3/5)$
unfolding ϱ -def by (*intro mult-left-mono*) auto
thus ?thesis
by *simp*

qed

definition ϱ -inv' :: $\text{real} \Rightarrow \text{real}$

where ϱ -inv' $x = -1 / (\text{real } b * (1 - x / \text{real } b) * \ln (1 - 1 / \text{real } b))$

lemma ϱ -inv'-bound:

assumes $x \geq 0$
assumes $x \leq 59/90 * b$
shows $|\varrho$ -inv' $x| \leq 4$

proof –

have $c: \ln (1 - 1 / \text{real } b) < 0$
using *b-min*
by (*subst ln-less-zero-iff*) auto
hence $d: \text{real } b * (1 - x / \text{real } b) * \ln (1 - 1 / \text{real } b) < 0$
using *b-min assms* by (*intro Rings.mult-pos-neg*) auto

have $(1::\text{real}) \leq 31/30$ **by** *simp*
also have $\dots \leq (31/30) * (b * -(-1 / \text{real } b))$
using *b-min* **by** *simp*
also have $\dots \leq (31/30) * (b * -\ln(1 + (-1 / \text{real } b)))$
using *b-min*
by (*intro mult-left-mono le-imp-neg-le ln-add-one-self-le-self2*) *auto*
also have $\dots \leq 3 * (31/90) * (-b * \ln(1 - 1 / \text{real } b))$
by *simp*
also have $\dots \leq 3 * (1 - x / \text{real } b) * (-b * \ln(1 - 1 / \text{real } b))$
using *assms b-min pos-divide-le-eq* [**where** $c=b$] *c*
by (*intro mult-right-mono mult-left-mono mult-nonpos-nonpos*) *auto*
also have $\dots \leq 3 * (\text{real } b * (1 - x / \text{real } b) * (-\ln(1 - 1 / \text{real } b)))$
by (*simp add:algebra-simps*)
finally have $3 * (\text{real } b * (1 - x / \text{real } b) * (-\ln(1 - 1 / \text{real } b))) \geq 1$ **by** *simp*
hence $3 * (\text{real } b * (1 - x / \text{real } b) * \ln(1 - 1 / \text{real } b)) \leq -1$ **by** *simp*
hence $\varrho\text{-inv}' x \leq 3$
unfolding *ϱ-inv'-def* **using** *d*
by (*subst neg-divide-le-eq*) *auto*
moreover have $\varrho\text{-inv}' x > 0$
unfolding *ϱ-inv'-def* **using** *d* **by** (*intro divide-neg-neg*) *auto*
ultimately show *?thesis* **by** *simp*
qed

lemma *ϱ-inv'*:

fixes $x :: \text{real}$
assumes $x < b$
shows *DERIV ϱ-inv x :> ϱ-inv' x*

proof –

have *DERIV (ln ∘ (λx. (1 - x / real b))) x :> 1 / (1 - x / real b) * (0 - 1/b)*
using *assms b-min*
by (*intro DERIV-chain DERIV-ln-divide DERIV-cdivide derivative-intros*) *auto*
hence *DERIV ϱ-inv x :> (1 / (1 - x / real b) * (-1/b)) / ln(1 - 1/real b)*
unfolding *comp-def ϱ-inv-def* **by** (*intro DERIV-cdivide*) *auto*
thus *?thesis*
by (*simp add:ϱ-inv'-def algebra-simps*)

qed

lemma *accuracy-without-cutoff*:

*measure Ψ { (f,g,h). |Y (f,g,h) - real X| > ε * X ∨ s f < q-max } ≤ 1/2⁴*
(is ?L ≤ ?R)

proof –

have $?L \leq \text{measure } \Psi \{ \psi. \neg E_1 \psi \vee \neg E_2 \psi \vee \neg E_3 \psi \vee \neg E_4 \psi \}$
proof (*rule pmf-rev-mono*)
fix ψ **assume** $\psi \in \text{set-pmf (sample-pro } \Psi)$
obtain $f g h$ **where** $\psi\text{-def: } \psi = (f,g,h)$ **by** (*metis prod-cases3*)

assume $\psi \notin \{ \psi. \neg E_1 \psi \vee \neg E_2 \psi \vee \neg E_3 \psi \vee \neg E_4 \psi \}$
hence *assms: E₁ (f,g,h) E₂ (f,g,h) E₃ (f,g,h) E₄ (f,g,h)*
unfolding $\psi\text{-def}$ **by** *auto*

define $I :: \text{real set}$ **where** $I = \{0..59/90*b\}$

have $p(f,g,h) \leq \varrho(\text{card}(R f)) + \varepsilon/12 * \text{card}(R f)$
using *assms(4) E₄-def* **unfolding** *abs-le-iff* **by** *simp*
also have $\dots \leq \varrho(2/3*b) + 1/12 * (2/3*b)$
using $\varepsilon\text{-lt-1 } R\text{-bound}$ [*OF assms(1,2)*]
by (*intro add-mono rho-mono mult-mono*) *auto*

also have $\dots \leq 3/5 * b + 1/18*b$
by *(intro add-mono rho-two-thirds)* *auto*
also have $\dots \leq 59/90 * b$
by *simp*
finally have $p(f,g,h) \leq 59/90 * b$ **by** *simp*
hence *p-in-I*: $p(f,g,h) \in I$
unfolding *I-def* **by** *simp*

have $\varrho(\text{card}(R f)) \leq \varrho(2/3 * b)$
using *R-bound[OF assms(1,2)]*
by *(intro rho-mono)* *auto*
also have $\dots \leq 3/5 * b$
using *rho-two-thirds* **by** *simp*
also have $\dots \leq b * 59/90$ **by** *simp*
finally have $\varrho(\text{card}(R f)) \leq b * 59/90$ **by** *simp*
moreover have $(1 - 1 / \text{real } b) \text{ powr } (\text{real } (\text{card}(R f))) \leq 1 \text{ powr } (\text{real } (\text{card}(R f)))$
using *b-min* **by** *(intro powr-mono2)* *auto*
hence $\varrho(\text{card}(R f)) \geq 0$
unfolding *ϱ-def* **by** *(intro mult-nonneg-nonneg)* *auto*
ultimately have $\varrho(\text{card}(R f)) \in I$
unfolding *I-def* **by** *simp*

moreover have *interval I*
unfolding *I-def interval-def* **by** *simp*
moreover have $59 / 90 * b < b$
using *b-min* **by** *simp*
hence *DERIV* $\varrho\text{-inv } x :> \varrho\text{-inv}' x$ **if** $x \in I$ **for** x
using *that I-def* **by** *(intro ϱ-inv')* *simp*
ultimately obtain $\xi :: \text{real}$ **where** $\xi\text{-def}$: $\xi \in I$
 $\varrho\text{-inv}(p(f,g,h)) - \varrho\text{-inv}(\varrho(\text{card}(R f))) = (p(f,g,h) - \varrho(\text{card}(R f))) * \varrho\text{-inv}' \xi$
using *p-in-I MVT-interval* **by** *blast*

have $|\varrho\text{-inv}(p(f,g,h)) - \text{card}(R f)| = |\varrho\text{-inv}(p(f,g,h)) - \varrho\text{-inv}(\varrho(\text{card}(R f)))|$
by *(subst ϱ-inverse)* *simp*
also have $\dots = |(p(f,g,h) - \varrho(\text{card}(R f)))| * |\varrho\text{-inv}' \xi|$
using $\xi\text{-def}(2)$ *abs-mult* **by** *simp*
also have $\dots \leq |p(f,g,h) - \varrho(\text{card}(R f))| * 4$
using $\xi\text{-def}(1)$ *I-def*
by *(intro mult-left-mono ϱ-inv'-bound)* *auto*
also have $\dots \leq (\varepsilon/12 * \text{card}(R f)) * 4$
using *assms(4)* *E4-def* **by** *(intro mult-right-mono)* *auto*
also have $\dots = \varepsilon/3 * \text{card}(R f)$ **by** *simp*
finally have $b: |\varrho\text{-inv}(p(f,g,h)) - \text{card}(R f)| \leq \varepsilon/3 * \text{card}(R f)$ **by** *simp*

have $|\varrho\text{-inv}(p(f,g,h)) - X / 2 \wedge (s f)| \leq$
 $|\varrho\text{-inv}(p(f,g,h)) - \text{card}(R f)| + |\text{card}(R f) - X / 2 \wedge (s f)|$
by *simp*
also have $\dots \leq \varepsilon/3 * \text{card}(R f) + |\text{card}(R f) - X / 2 \wedge (s f)|$
by *(intro add-mono b)* *auto*
also have $\dots = \varepsilon/3 * |X / 2 \wedge (s f) + (\text{card}(R f) - X / 2 \wedge (s f))| +$
 $|\text{card}(R f) - X / 2 \wedge (s f)|$ **by** *simp*
also have $\dots \leq \varepsilon/3 * (|X / 2 \wedge (s f)| + |\text{card}(R f) - X / 2 \wedge (s f)|) +$
 $|\text{card}(R f) - X / 2 \wedge (s f)|$
using $\varepsilon\text{-gt-0}$ **by** *(intro mult-left-mono add-mono abs-triangle-ineq)* *auto*
also have $\dots \leq \varepsilon/3 * |X / 2 \wedge (s f)| + (1 + \varepsilon/3) * |\text{card}(R f) - X / 2 \wedge (s f)|$
using $\varepsilon\text{-gt-0}$ $\varepsilon\text{-lt-1}$ **by** *(simp add:algebra-simps)*
also have $\dots \leq \varepsilon/3 * |X / 2 \wedge s f| + (4/3) * (\varepsilon / 3 * \text{real } X / 2 \wedge s f)$
using *assms(2)* $\varepsilon\text{-gt-0}$ $\varepsilon\text{-lt-1}$

unfolding E_2 -def **by** (intro add-mono mult-mono) auto
also have ... = $(7/9) * \varepsilon * \text{real } X / 2^{\wedge} s f$
using X -ge-1 **by** (subst abs-of-nonneg) auto
also have ... $\leq 1 * \varepsilon * \text{real } X / 2^{\wedge} s f$
using ε -gt-0 **by** (intro mult-mono divide-right-mono) auto
also have ... = $\varepsilon * \text{real } X / 2^{\wedge} s f$ **by** simp
finally have $a: |\varrho\text{-inv}(p(f,g,h)) - X / 2^{\wedge}(s f)| \leq \varepsilon * X / 2^{\wedge}(s f)$
by simp

have $|Y(f, g, h) - \text{real } X| = |2^{\wedge}(s f) * |\varrho\text{-inv}(p(f,g,h)) - \text{real } X / 2^{\wedge}(s f)|$
unfolding Y -def **by** (subst abs-mult[symmetric]) (simp add: algebra-simps powr-add[symmetric])
also have ... $\leq 2^{\wedge}(s f) * (\varepsilon * X / 2^{\wedge}(s f))$
by (intro mult-mono a) auto
also have ... = $\varepsilon * X$
by (simp add: algebra-simps powr-add[symmetric])
finally have $|Y(f, g, h) - \text{real } X| \leq \varepsilon * X$ **by** simp
moreover have $2 \text{ powr } (\lceil \log 2 (\text{real } X) \rceil - t f) \leq 2 \text{ powr } b\text{-exp}$ (is ?L1 \leq ?R1)
proof -
have ?L1 $\leq 2 \text{ powr } (1 + \log 2 (\text{real } X) - t f)$
by (intro powr-mono, linarith) auto
also have ... = $2 \text{ powr } 1 * 2 \text{ powr } (\log 2 (\text{real } X)) * 2 \text{ powr } (- t f)$
unfolding powr-add[symmetric] **by** simp
also have ... = $2 * (2 \text{ powr } (- t f) * X)$
using X -ge-1 **by** simp
also have ... $\leq 2 * (b/2)$
using assms(1) **unfolding** E_1 -def **by** (intro mult-left-mono) auto
also have ... = b **by** simp
also have ... = ?R1
unfolding b -def **by** (simp add: powr-realpow)
finally show ?thesis **by** simp
qed
hence $\lceil \log 2 (\text{real } X) \rceil - t f \leq \text{real } b\text{-exp}$
unfolding not-less[symmetric] **using** powr-less-mono[where $x=2$] **by** simp
hence $s f \geq q\text{-max}$ **unfolding** s -def $q\text{-max}$ -def **by** (intro nat-mono) auto
ultimately show $\psi \notin \{(f, g, h). \varepsilon * X < |Y(f, g, h) - \text{real } X| \vee s f < q\text{-max}\}$
unfolding ψ -def **by** auto
qed
also have ... \leq
 $\text{measure } \Psi \{\psi. \neg E_1 \psi \vee \neg E_2 \psi \vee \neg E_3 \psi\} + \text{measure } \Psi \{\psi. E_1 \psi \wedge E_2 \psi \wedge E_3 \psi \wedge \neg E_4 \psi\}$
by (intro pmf-add) auto
also have ... $\leq (\text{measure } \Psi \{\psi. \neg E_1 \psi \vee \neg E_2 \psi\} + \text{measure } \Psi \{\psi. E_1 \psi \wedge E_2 \psi \wedge \neg E_3 \psi\})$
 $+ 1/2^{\wedge}6$
by (intro add-mono e-4 pmf-add) auto
also have ... $\leq ((\text{measure } \Psi \{\psi. \neg E_1 \psi\} + \text{measure } \Psi \{\psi. E_1 \psi \wedge \neg E_2 \psi\}) + 1/2^{\wedge}6) + 1/2^{\wedge}6$
by (intro add-mono e-3 pmf-add) auto
also have ... $\leq ((1/2^{\wedge}6 + 1/2^{\wedge}6) + 1/2^{\wedge}6) + 1/2^{\wedge}6$
by (intro add-mono e-2 e-1) auto
also have ... = ?R **by** simp
finally show ?thesis **by** simp
qed

end

end

8 Cutoff Level

This section verifies that the cutoff will be below $q\text{-max}$ with high probability. The result will be needed in Section 9, where it is shown that the estimates will be accurate for any cutoff below $q\text{-max}$.

```

theory Distributed-Distinct-Elements-Cutoff-Level
  imports
    Distributed-Distinct-Elements-Accuracy-Without-Cutoff
    Distributed-Distinct-Elements-Tail-Bounds
begin

hide-const (open) Quantum.Z

unbundle intro-cong-syntax

lemma mono-real-of-int: mono real-of-int
  unfolding mono-def by auto

lemma Max-le-Sum:
  fixes  $f :: 'a \Rightarrow \text{int}$ 
  assumes finite A
  assumes  $\bigwedge a. a \in A \implies f\ a \geq 0$ 
  shows  $\text{Max} (\text{insert } 0 (f\ 'A)) \leq (\sum a \in A . f\ a)$  (is  $?L \leq ?R$ )
proof (cases A $\neq\{\}$ )
  case True

  have  $0: f\ a \leq (\sum a \in A . f\ a)$  if  $a \in A$  for  $a$ 
    using that assms by (intro member-le-sum) auto

  have  $?L = \text{max } 0 (\text{Max} (f\ 'A))$ 
    using True assms(1) by (subst Max-insert) auto
  also have  $\dots = \text{Max} (\text{max } 0\ 'f\ 'A)$ 
    using assms True by (intro mono-Max-commute monoI) auto
  also have  $\dots = \text{Max} (f\ 'A)$ 
    unfolding image-image using assms
    by (intro arg-cong[where f=Max] image-cong) auto
  also have  $\dots \leq ?R$ 
    using  $0$  True assms(1)
    by (intro iffD2[OF Max-le-iff]) auto
  finally show ?thesis by simp
next
  case False
  hence  $A = \{\}$  by simp
  then show ?thesis by simp
qed

context inner-algorithm-fix-A
begin

```

The following inequality is true for base e on the entire domain ($x > 0$). It is shown in *ln-add-one-self-le-self*. In the following it is established for base 2, where it holds for $x \geq 1$.

```

lemma log-2-estimate:
  assumes  $x \geq (1::\text{real})$ 
  shows  $\log 2 (1+x) \leq x$ 
proof –
  define  $f$  where  $f\ x = x - \log 2 (1+x)$  for  $x :: \text{real}$ 
  define  $f'$  where  $f'\ x = 1 - 1/((x+1)*\ln 2)$  for  $x :: \text{real}$ 

```

have $0:(f \text{ has-real-derivative } (f' x)) \text{ (at } x) \text{ if } x > 0 \text{ for } x$
unfolding $f\text{-def } f'\text{-def}$ **using** $that$
by $(auto \text{ intro!}: \text{ derivative-eq-intros})$

have $f' x \geq 0 \text{ if } 1 \leq x \text{ for } x :: \text{ real}$
proof –
 have $(1::\text{real}) \leq 2*\ln 2$
by $(\text{approximation } 5)$
also have $\dots \leq (x+1)*\ln 2$
using $that$ **by** $(\text{intro mult-right-mono}) \text{ auto}$
finally have $1 \leq (x+1)*\ln 2$ **by** simp
hence $1/((x+1)*\ln 2) \leq 1$
by simp
thus $?thesis$
unfolding $f'\text{-def}$ **by** simp
qed

hence $\exists y. (f \text{ has-real-derivative } y) \text{ (at } x) \wedge 0 \leq y \text{ if } x \geq 1 \text{ for } x :: \text{ real}$
using $that \text{ order-less-le-trans}[OF \text{ exp-gt-zero}]$
by $(\text{intro exI}[\text{where } x=f' x] \text{ conjI } 0) \text{ auto}$
hence $f 1 \leq f x$
by $(\text{intro DERIV-nonneg-imp-nondecreasing}[OF \text{ assms}]) \text{ auto}$
thus $?thesis$
unfolding $f\text{-def}$ **by** simp
qed

lemma cutoff-eq-7 :
 $\text{real } X * 2 \text{ powr } (-\text{real } q\text{-max}) / b \leq 1$
proof –
 have $\text{real } X = 2 \text{ powr } (\log 2 X)$
using $X\text{-ge-1}$ **by** $(\text{intro powr-log-cancel}[\text{symmetric}]) \text{ auto}$
also have $\dots \leq 2 \text{ powr } (\text{nat } \lceil \log 2 X \rceil)$
by $(\text{intro powr-mono}) \text{ linarith+}$
also have $\dots = 2^{\wedge} (\text{nat } \lceil \log 2 X \rceil)$
by $(\text{subst powr-realpow}) \text{ auto}$
also have $\dots = \text{real } (2^{\wedge} (\text{nat } \lceil \log 2 (\text{real } X) \rceil))$
by simp
also have $\dots \leq \text{real } (2^{\wedge} (b\text{-exp} + \text{nat } (\lceil \log 2 (\text{real } X) \rceil) - \text{int } b\text{-exp}))$
by $(\text{intro Nat.of-nat-mono power-increasing}) \text{ linarith+}$
also have $\dots = b * 2^{\wedge} q\text{-max}$
unfolding $q\text{-max-def } b\text{-def}$ **by** $(\text{simp add: power-add})$

finally have $\text{real } X \leq b * 2^{\wedge} q\text{-max}$ **by** simp
thus $?thesis$
using $b\text{-min}$
unfolding $\text{powr-minus inverse-eq-divide}$
by $(\text{simp add:field-simps powr-realpow})$
qed

lemma cutoff-eq-6 :
fixes k
assumes $a \in A$
shows $(f f. \text{real-of-int } (\text{max } 0 (\text{int } (f a) - \text{int } k)) \partial\Psi_1) \leq 2 \text{ powr } (-\text{real } k) \text{ (is } ?L \leq ?R)$
proof $(\text{cases } k \leq n\text{-exp} - 1)$
case True
have $a\text{-le-n}: a < n$
using $\text{assms } A\text{-range}$ **by** auto

have ?L = ($\int x. \text{real-of-int } (\max 0 (\text{int } x - k)) \partial \text{map-pmf } (\lambda x. x a) \Psi_1$)
by *simp*
also have ... = ($\int x. \text{real-of-int } (\max 0 (\text{int } x - k)) \partial (\mathcal{G} \text{ n-exp})$)
by (*subst hash-pro-component*[OF Ψ_1 a-le-n]) *auto*
also have ... = ($\int x. \max 0 (\text{real } x - \text{real } k) \partial (\mathcal{G} \text{ n-exp})$)
unfolding *max-of-mono*[OF *mono-real-of-int,symmetric*] **by** *simp*
also have ... = ($\sum x < n\text{-exp}. \max 0 (\text{real } x - \text{real } k) * \text{pmf } (\mathcal{G} \text{ n-exp}) x$)
using *geom-pro-range* **by** (*intro integral-measure-pmf-real*) *auto*
also have ... = ($\sum x = k+1..n\text{-exp}. (\text{real } x - \text{real } k) * \text{pmf } (\mathcal{G} \text{ n-exp}) x$)
by (*intro sum.mono-neutral-cong-right*) *auto*
also have ... = ($\sum x = k+1..n\text{-exp}. (\text{real } x - \text{real } k) * \text{measure } (\mathcal{G} \text{ n-exp}) \{x\}$)
unfolding *measure-pmf-single* **by** *simp*
also have ... = ($\sum x = k+1..n\text{-exp}. (\text{real } x - \text{real } k) * (\text{measure } (\mathcal{G} \text{ n-exp}) (\{\omega. \omega \geq x\} - \{\omega. \omega \geq (x+1)\}))$)
by (*intro sum.cong arg-cong2*[**where** $f=(*)$] *measure-pmf-cong*) *auto*
also have ... = ($\sum x = k+1..n\text{-exp}. (\text{real } x - \text{real } k) * (\text{measure } (\mathcal{G} \text{ n-exp}) \{\omega. \omega \geq x\} - \text{measure } (\mathcal{G} \text{ n-exp}) \{\omega. \omega \geq (x+1)\})$)
by (*intro sum.cong arg-cong2*[**where** $f=(*)$] *measure-Diff*) *auto*
also have ... = ($\sum x = k+1..n\text{-exp}. (\text{real } x - \text{real } k) * (1/2^{\wedge}x - \text{of-bool}(x+1 \leq n\text{-exp})/2^{\wedge}(x+1))$)
unfolding *geom-pro-prob* **by** (*intro-cong* [$\sigma_2 (*)$, $\sigma_2 (-)$, $\sigma_2 (/)$] *more:sum.cong*) *auto*
also have ... =
($\sum x = k+1..n\text{-exp}. (\text{real } x - k)/2^{\wedge}x - (\sum x = k+1..n\text{-exp}. (\text{real } x - k) * \text{of-bool}(x+1 \leq n\text{-exp})/2^{\wedge}(x+1))$)
by (*simp add:algebra-simps sum-subtractf*)
also have ... = ($\sum x = k+1..n\text{-exp}. (\text{real } x - k)/2^{\wedge}x - (\sum x = k+1..n\text{-exp}-1. (\text{real } x - k)/2^{\wedge}(x+1))$)
by (*intro arg-cong2*[**where** $f=(-)$] *refl sum.mono-neutral-cong-right*) *auto*
also have ... = ($\sum x = k+1..(n\text{-exp}-1)+1. (\text{real } x - k)/2^{\wedge}x - (\sum x = k+1..n\text{-exp}-1. (\text{real } x - k)/2^{\wedge}(x+1))$)
using *n-exp-gt-0* **by** (*intro arg-cong2*[**where** $f=(-)$] *refl sum.cong*) *auto*
also have ... = ($\sum x \in \text{insert } k \{k+1..n\text{-exp}-1\}. (\text{real } (x+1) - k)/2^{\wedge}(x+1) - (\sum x = k+1..n\text{-exp}-1. (\text{real } x - k)/2^{\wedge}(x+1))$)
unfolding *sum.shift-bounds-cl-nat-ivl* **using** *True*
by (*intro arg-cong2*[**where** $f=(-)$] *sum.cong*) *auto*
also have ... = $1/2^{\wedge}(k+1) + (\sum x = k+1..n\text{-exp}-1. (\text{real } (x+1) - k)/2^{\wedge}(x+1) - (\text{real } x - k)/2^{\wedge}(x+1))$
by (*subst sum.insert*) (*auto simp add:sum-subtractf*)
also have ... = $1/2^{\wedge}(k+1) + (\sum x = k+1..n\text{-exp}-1. (1/2^{\wedge}(x+1)))$
by (*intro arg-cong2*[**where** $f=(+)$] *sum.cong refl*) (*simp add:field-simps*)
also have ... = ($\sum x \in \text{insert } k \{k+1..n\text{-exp}-1\}. (1/2^{\wedge}(x+1))$)
by (*subst sum.insert*) *auto*
also have ... = ($\sum x = 0+k..(n\text{-exp}-1-k)+k. 1/2^{\wedge}(x+1)$)
using *True* **by** (*intro sum.cong*) *auto*
also have ... = ($\sum x < n\text{-exp}-k. 1/2^{\wedge}(x+k+1)$)
unfolding *sum.shift-bounds-cl-nat-ivl* **using** *True n-exp-gt-0* **by** (*intro sum.cong*) *auto*
also have ... = $(1/2)^{\wedge}(k+1) * (\sum x < n\text{-exp}-k. (1/2)^{\wedge}x)$
unfolding *sum-distrib-left power-add*[*symmetric*] **by** (*simp add:power-divide ac-simps*)
also have ... = $(1/2)^{\wedge}(k+1) * 2 * (1 - (1/2)^{\wedge}(n\text{-exp} - k))$
by (*subst geometric-sum*) *auto*
also have ... $\leq (1/2)^{\wedge}(k+1) * 2 * (1 - 0)$
by (*intro mult-left-mono diff-mono*) *auto*
also have ... = $(1/2)^{\wedge}k$
unfolding *power-add* **by** *simp*
also have ... = ?R
unfolding *powr-minus* **by** (*simp add:powr-realpow inverse-eq-divide power-divide*)
finally show ?thesis
by *simp*
next
case *False*
hence *k-ge-n-exp*: $k \geq n\text{-exp}$
by *simp*
have *a-lt-n*: $a < n$

```

using assms A-range by auto

have ?L = (∫ x. real-of-int (max 0 (int x - k)) ∂map-pmf (λx. x a) Ψ1)
  by simp
also have ... = (∫ x. real-of-int (max 0 (int x - k)) ∂(G n-exp))
  by (subst hash-pro-component[OF Ψ1 a-lt-n]) auto
also have ... = (∫ x. real-of-int 0 ∂(G n-exp))
  using geom-pro-range k-ge-n-exp
  by (intro integral-cong-AE AE-pmfI iffD2[OF of-int-eq-iff] max-absorb1) force+
also have ... = 0 by simp
finally show ?thesis by simp
qed

lemma cutoff-eq-5:
  assumes x ≥ (-1 :: real)
  shows real-of-int [log 2 (x+2)] ≤ (real c+2) + max (x - 2^c) 0 (is ?L ≤ ?R)
proof -
  have 0: 1 ≤ 2^1 * ln (2::real)
    by (approximation 5)

  consider (a) c = 0 ∧ x ≥ 2^c+1 | (b) c > 0 ∧ x ≥ 2^c+1 | (c) x ≤ 2^c+1
    by linarith
  hence log 2 (x+2) ≤ ?R
  proof (cases)
    case a
    have log 2 (x+2) = log 2 (1+(x+1))
      by (simp add:algebra-simps)
    also have ... ≤ x+1
      using a by (intro log-2-estimate) auto
    also have ... = ?R
      using a by auto
    finally show ?thesis by simp
  next
    case b
    have 0 < 0 + (1::real)
      by simp
    also have ... ≤ 2^c+(1::real)
      by (intro add-mono) auto
    also have ... ≤ x
      using b by simp
    finally have x-gt-0: x > 0
      by simp

    have log 2 (x+2) = log 2 ((x+2)/2^c) + c
      using x-gt-0 by (subst log-divide) auto
    also have ... = log 2 (1+(x+2-2^c)/2^c) + c
      by (simp add:divide-simps)
    also have ... ≤ (x+2-2^c)/2^c / ln 2 + c
      using b unfolding log-def
      by (intro add-mono divide-right-mono ln-add-one-self-le-self divide-nonneg-pos) auto
    also have ... = (x+2-2^c)/(2^c*ln 2) + c
      by simp
    also have ... ≤ (x+2-2^c)/(2^1*ln 2)+c
      using b by (intro add-mono divide-left-mono mult-right-mono power-increasing) simp-all
    also have ... ≤ (x+2-2^c)/1 + c
      using b by (intro add-mono divide-left-mono 0) auto
    also have ... ≤ (c+2) + max (x - 2^c) 0
      using b by simp

```

finally show *?thesis* **by** *simp*
next
case *c*
hence $\log 2 (x+2) \leq \log 2 ((2^c+1)+2)$
using *assms* **by** (*intro log-mono add-mono*) *auto*
also have $\dots = \log 2 (2^c*(1+3/2^c))$
by (*simp add:algebra-simps*)
also have $\dots = c + \log 2 (1+3/2^c)$
by (*subst log-mult*) (*auto intro:add-pos-nonneg*)
also have $\dots \leq c + \log 2 (1+3/2^0)$
by (*intro add-mono log-mono divide-left-mono power-increasing add-pos-nonneg*) *auto*
also have $\dots = c + \log 2 (2*2)$
by *simp*
also have $\dots = \text{real } c + 2$
by (*subst log-mult*) *auto*
also have $\dots \leq (c+2) + \max (x - 2^c) 0$
by *simp*
finally show *?thesis*
by *simp*
qed
moreover have $\lfloor \log 2 (x+2) \rfloor \leq \log 2 (x+2)$
by *simp*
ultimately show *?thesis* **using** *order-trans* **by** *blast*
qed

lemma *cutoff-level*:

measure $\Omega \{ \omega. q \ \omega \ A > q\text{-max} \} \leq \delta/2$ (**is** *?L* \leq *?R*)

proof –

have *C₁-est*: $C_1 * l \leq 30 * \text{real } l$

unfolding *C₁-def*

by (*intro mult-right-mono of-nat-0-le-iff*) (*approximation 10*)

define *Z* **where** $Z \ \omega = (\sum j < b. \text{real-of-int } \lfloor \log 2 (\text{of-int } (\max (\tau_1 \ \omega \ A \ q\text{-max } j) (-1)) + 2) \rfloor)$
for ω

define *V* **where** $V \ \omega = Z \ \omega / \text{real } b - 3$ **for** ω

have $2: Z \ \psi \leq \text{real } b * (\text{real } c + 2) + \text{of-int } (\sum a \in A. \max 0 (\text{int } (\text{fst } \psi \ a) - q\text{-max} - 2^c))$
(is *?L1* \leq *?R1*) **if** $\psi \in \text{sample-pro } \Psi$ **for** $c \ \psi$

proof –

obtain *f g h* **where** $\psi\text{-def}$: $\psi = (f, g, h)$

using *prod-cases3* **by** *blast*

have $\psi\text{-range}$: $(f, g, h) \in \text{sample-pro } \Psi$

using *that* **unfolding** $\psi\text{-def}$ **by** *simp*

have $-1 - 2^c \leq -1 - (1::\text{real})$

by (*intro diff-mono*) *auto*

also have $\dots \leq 0$ **by** *simp*

finally have $-1 - 2^c \leq (0::\text{real})$ **by** *simp*

hence *aux3*: $\max (-1 - 2^c) 0 = (0::\text{real})$

by (*intro max-absorb2*)

have $-1 - \text{int } q\text{-max} - 2^c \leq -1 - 0 - 1$

by (*intro diff-mono*) *auto*

also have $\dots \leq 0$ **by** *simp*

finally have $-1 - \text{int } q\text{-max} - 2^c \leq 0$ **by** *simp*

hence *aux3-2*: $\max 0 (-1 - \text{int } q\text{-max} - 2^c) = 0$

by (*intro max-absorb1*)

have ?L1 $\leq (\sum j < b. (\text{real } c + 2) + \max (\text{real-of-int } (\max (\tau_1 \psi A q\text{-max } j) (-1)) - 2^{\wedge}c) 0)$
unfolding Z-def **by** (intro sum-mono cutoff-eq-5) auto
also have ... = $(\sum j < b. (\text{real } c + 2) + \max (\tau_0 \psi A j - q\text{-max} - 2^{\wedge}c) 0)$
unfolding τ_1 -def max-of-mono[OF mono-real-of-int,symmetric]
by (intro-cong $[\sigma_2 (+)]$ more:sum.cong) (simp add:max-diff-distrib-left max.assoc aux3)
also have ... = $\text{real } b * (\text{real } c + 2) +$
 $\text{of-int } (\sum j < b. (\max 0 (\text{Max } (\text{insert } (-1) \{ \text{int } (f a) \mid a. a \in A \wedge h (g a) = j \}) - q\text{-max} - 2^{\wedge}c)))$
unfolding ψ -def **by** (simp add:max.commute)
also have ... = $\text{real } b * (\text{real } c + 2) +$
 $\text{of-int } (\sum j < b. \max 0 (\text{Max } ((\lambda x. x - q\text{-max} - 2^{\wedge}c) (\text{insert } (-1) \{ \text{int } (f a) \mid a. a \in A \wedge h (g a) = j \}))))$
using fin-A
by (intro-cong $[\sigma_2 (+), \sigma_1 \text{ of-int}, \sigma_2 \text{ max}]$ more:sum.cong mono-Max-commute) (auto simp:monoI)
also have ... = $\text{real } b * (\text{real } c + 2) +$
 $\text{of-int } (\sum j < b. \max 0 (\text{Max } (\text{insert } (-1 - q\text{-max} - 2^{\wedge}c) \{ \text{int } (f a) - q\text{-max} - 2^{\wedge}c \mid a. a \in A \wedge h (g a) = j \})))$
by (intro-cong $[\sigma_2 (+), \sigma_1 \text{ of-int}, \sigma_2 \text{ max}, \sigma_1 \text{ Max}]$ more:sum.cong) auto
also have ... = $\text{real } b * (\text{real } c + 2) + \text{of-int}$
 $(\sum j < b. \text{Max } ((\max 0) (\text{insert } (-1 - q\text{-max} - 2^{\wedge}c) \{ \text{int } (f a) - q\text{-max} - 2^{\wedge}c \mid a. a \in A \wedge h (g a) = j \})))$
using fin-A **by** (intro-cong $[\sigma_2 (+), \sigma_1 \text{ of-int}]$ more:sum.cong mono-Max-commute)
(auto simp add:monoI setcompr-eq-image)
also have ... = $\text{real } b * (\text{real } c + 2) +$
 $\text{of-int } (\sum j < b. \text{Max } (\text{insert } 0 \{ \max 0 (\text{int } (f a) - q\text{-max} - 2^{\wedge}c) \mid a. a \in A \wedge h (g a) = j \}))$
using aux3-2 **by** (intro-cong $[\sigma_2 (+), \sigma_1 \text{ of-int}, \sigma_1 \text{ Max}]$ more:sum.cong)
(simp add:setcompr-eq-image image-image)
also have ... $\leq b * (\text{real } c + 2) + \text{of-int } (\sum j < b. (\sum a \mid a \in A \wedge h (g a) = j. \max 0 (\text{int } (f a) - q\text{-max} - 2^{\wedge}c)))$
using fin-A Max-le-Sum **unfolding** setcompr-eq-image
by (intro add-mono iffD2[OF of-int-le-iff] sum-mono Max-le-Sum) (simp-all)
also have ... = $\text{real } b * (\text{real } c + 2) +$
 $\text{of-int } (\sum a \in (\bigcup j \in \{.. < b\}. \{ a. a \in A \wedge h (g a) = j \}). \max 0 (\text{int } (f a) - q\text{-max} - 2^{\wedge}c))$
using fin-A
by (intro-cong $[\sigma_2 (+), \sigma_1 \text{ of-int}]$ more:sum.UNION-disjoint[symmetric]) auto
also have ... = $\text{real } b * (\text{real } c + 2) + \text{of-int } (\sum a \in A. \max 0 (\text{int } (f a) - q\text{-max} - 2^{\wedge}c))$
using h-range[OF ψ -range] **by** (intro-cong $[\sigma_2 (+), \sigma_1 \text{ of-int}]$ more:sum.cong) auto
also have ... = ?R1
unfolding ψ -def **by** simp
finally show ?thesis
by simp
qed

have 1: *measure* $\Psi \{ \psi. \text{real } c \leq V \psi \} \leq 2 \text{ powr } (- (2^{\wedge}c))$ (**is** ?L1 \leq ?R1) **for** c
proof –

have ?L1 = *measure* $\Psi \{ \psi. \text{real } b * (\text{real } c + 3) \leq Z \psi \}$
unfolding V-def **using** b-min **by** (intro measure-pmf-cong) (simp add:field-simps)
also have ... $\leq \text{measure } \Psi$
 $\{ \psi. \text{real } b * (\text{real } c + 3) \leq \text{real } b * (\text{real } c + 2) + \text{of-int } (\sum a \in A. \max 0 (\text{int } (f a) - q\text{-max} - 2^{\wedge}c)) \}$
using 2 order-trans **by** (intro pmf-mono) blast
also have ... = *measure* $\Psi \{ \psi. \text{real } b \leq (\sum a \in A. \text{of-int } (\max 0 (\text{int } (f a) - q\text{-max} - 2^{\wedge}c))) \}$
by (intro measure-pmf-cong) (simp add:algebra-simps)
also have ... $\leq (\int \psi. (\sum a \in A. \text{of-int } (\max 0 (\text{int } (f a) - q\text{-max} - 2^{\wedge}c))) \partial \Psi) / \text{real } b$
using b-min **by** (intro pmf-markov sum-nonneg) simp-all
also have ... = $(\sum a \in A. (\int \psi. \text{of-int } (\max 0 (\text{int } (f a) - q\text{-max} - 2^{\wedge}c)) \partial \Psi)) / \text{real } b$
by (intro-cong $[\sigma_2 (/)]$ more:Bochner-Integration.integral-sum) simp

also have ... = $(\sum a \in A. (\int f. \text{of-int} (\max 0 (\text{int} (f a) - q\text{-max} - 2^{\wedge}c)) \partial(\text{map-pmf fst } \Psi))) / \text{real } b$

b

by *simp*

also have ... = $(\sum a \in A. (\int f. \text{of-int} (\max 0 (\text{int} (f a) - (q\text{-max} + 2^{\wedge}c))) \partial\Psi_1)) / \text{real } b$

unfolding *sample-pro-Ψ map-fst-pair-pmf* by *(simp add: algebra-simps)*

also have ... ≤ $(\sum a \in A. 2^{\text{powr } -\text{real } (q\text{-max} + 2^{\wedge}c)}) / \text{real } b$

using *b-min* by *(intro sum-mono divide-right-mono cutoff-eq-6) auto*

also have ... = $\text{real } X * 2^{\text{powr } (-\text{real } q\text{-max} + (- (2^{\wedge}c)))} / \text{real } b$

unfolding *X-def* by *simp*

also have ... = $(\text{real } X * 2^{\text{powr } (-\text{real } q\text{-max})} / b) * 2^{\text{powr } (- (2^{\wedge}c))}$

unfolding *powr-add* by *(simp add: algebra-simps)*

also have ... ≤ $1 * 2^{\text{powr } (- (2^{\wedge}c))}$

using *cutoff-eq-7* by *(intro mult-right-mono) auto*

finally show *?thesis*

by *simp*

qed

have 0: *measure* $\Psi \{\psi. x \leq V \psi\} \leq \exp(-x * \ln x^{\wedge}3)$ (is *?L1* ≤ *?R1*) if $x \geq 20$ for x
proof –

define *c* where $c = \text{nat } \lfloor x \rfloor$

have $x * \ln x^{\wedge}3 \leq \exp(x * \ln 2) * \ln 2 / 2$ if $x \geq 150$ for $x::\text{real}$

proof –

have *aux-aux-0*: $x^{\wedge}4 \geq 0$

by *simp*

have $x * \ln x^{\wedge}3 \leq x * x^{\wedge}3$

using *that* by *(intro mult-left-mono power-mono ln-bound) auto*

also have ... = $x^{\wedge}4 * 1$

by *(simp add: numeral-eq-Suc)*

also have ... ≤ $x^{\wedge}4 * ((\ln 2 / 10)^{\wedge}4 * (150 * (\ln 2 / 10))^{\wedge}6 * (\ln 2 / 2))$

by *(intro mult-left-mono aux-aux-0) (approximation 8)*

also have ... = $(x * (\ln 2 / 10))^{\wedge}4 * (150 * (\ln 2 / 10))^{\wedge}6 * (\ln 2 / 2)$

unfolding *power-mult-distrib* by *(simp add: algebra-simps)*

also have ... ≤ $(x * (\ln 2 / 10))^{\wedge}4 * (x * (\ln 2 / 10))^{\wedge}6 * (\ln 2 / 2)$

by *(intro mult-right-mono mult-left-mono power-mono that) auto*

also have ... = $(0 + x * (\ln 2 / 10))^{\wedge}10 * (\ln 2 / 2)$

unfolding *power-add[symmetric]* by *simp*

also have ... ≤ $(1 + x * \ln 2 / 10)^{\wedge}10 * (\ln 2 / 2)$

using *that* by *(intro mult-right-mono power-mono add-mono) auto*

also have ... ≤ $\exp(x * \ln 2 / 10)^{\wedge}10 * (\ln 2 / 2)$

using *that* by *(intro mult-right-mono power-mono exp-ge-add-one-self) auto*

also have ... = $\exp(x * \ln 2) * (\ln 2 / 2)$

unfolding *exp-of-nat-mult[symmetric]* by *simp*

finally show *?thesis* by *simp*

qed

moreover have $x * \ln x^{\wedge}3 \leq \exp(x * \ln 2) * \ln 2 / 2$ if $x \in \{20..150\}$

using *that* by *(approximation 10 splitting: x=1)*

ultimately have $x * \ln x^{\wedge}3 \leq \exp(x * \ln 2) * \ln 2 / 2$

using *that* by *fastforce*

also have ... = $2^{\text{powr } (x-1)} * \ln 2$

unfolding *powr-diff* unfolding *powr-def* by *simp*

also have ... ≤ $2^{\text{powr } c} * \ln 2$

unfolding *c-def* using *that*

by *(intro mult-right-mono powr-mono) auto*

also have ... = $2^{\wedge}c * \ln 2$

using *powr-realpow* by *simp*

```

finally have aux0:  $x * \ln x^3 \leq 2^c * \ln 2$ 
  by simp
have real c  $\leq x$ 
  using that unfolding c-def by linarith
hence ?L1  $\leq$  measure  $\Psi \{ \psi. \text{real } c \leq V \psi \}$ 
  by (intro pmf-mono) auto
also have ...  $\leq 2 \text{ powr } (-(2^c))$ 
  by (intro 1)
also have ... = exp  $(- (2^c * \ln 2))$ 
  by (simp add:powr-def)
also have ...  $\leq$  exp  $(- (x * \ln x^3))$ 
  using aux0 by (intro iffD2[OF exp-le-cancel-iff]) auto
also have ... = ?R1
  by simp
finally show ?thesis
  by simp
qed

have ?L  $\leq$  measure  $\Omega \{ \omega. \text{is-too-large } (\tau_2 \omega A \text{ q-max}) \}$ 
  using lt-s-too-large
  by (intro pmf-mono) (simp del:is-too-large.simps)
also have ... = measure  $\Omega$ 
   $\{ \omega. (\sum (i,j) \in \{..<l\} \times \{..<b\}. \lfloor \log 2 (\text{of-int } (\max (\tau_2 \omega A \text{ q-max } i j) (-1)) + 2) \rfloor) > C_5 * b$ 
  *l}
  by simp
also have ... = measure  $\Omega \{ \omega. \text{real-of-int } (\sum (i,j) \in \{..<l\} \times \{..<b\}. \lfloor \log 2 (\text{of-int } (\max (\tau_2 \omega A \text{ q-max } i j) (-1)) + 2) \rfloor) > \text{of-int } (C_5 * b * l) \}$ 
  unfolding of-int-less-iff by simp
also have ... = measure  $\Omega \{ \omega. \text{real-of-int } C_5 * \text{real } b * \text{real } l < \text{of-int } (\sum x \in \{..<l\} \times \{..<b\}. \lfloor \log 2 (\text{real-of-int } (\tau_1 (\omega (\text{fst } x)) A \text{ q-max } (\text{snd } x)) + 2) \rfloor) \}$ 
  by (intro-cong  $[\sigma_2 \text{ measure}, \sigma_1 \text{ Collect}, \sigma_1 \text{ of-int}, \sigma_2 (<)] \text{ more:ext sum.cong}$ )
  (auto simp add:case-prod-beta  $\tau_2$ -def  $\tau_1$ -def)

also have ... = measure  $\Omega \{ \omega. (\sum i < l. Z (\omega i)) > \text{of-int } C_5 * \text{real } b * \text{real } l \}$ 
  unfolding Z-def sum.cartesian-product  $\tau_1$ -def by (simp add:case-prod-beta)
also have ... = measure  $\Omega \{ \omega. (\sum i < l. V (\omega i) + 3) > \text{of-int } C_5 * \text{real } l \}$ 
  unfolding V-def using b-min
  by (intro measure-pmf-cong) (simp add:sum-divide-distrib[symmetric] field-simps sum.distrib)
also have ... = measure  $\Omega \{ \omega. (\sum i < l. V (\omega i)) > \text{of-int } (C_5 - 3) * \text{real } l \}$ 
  by (simp add:sum.distrib algebra-simps)
also have ...  $\leq$  measure  $\Omega \{ \omega. (\sum i < l. V (\omega i)) \geq C_1 * \text{real } l \}$ 
  unfolding  $C_5$ -def using  $C_1$ -est by (intro pmf-mono) auto
also have ...  $\leq$  exp  $(- \text{real } l)$ 
  by (intro deviation-bound l-gt-0 0) (simp-all add:  $\Lambda$ -def)
also have ...  $\leq$  exp  $(- (C_6 * \ln (2 / \delta)))$ 
  using l-bound by (intro iffD2[OF exp-le-cancel-iff]) auto
also have ...  $\leq$  exp  $(- (1 * \ln (2 / \delta)))$ 
  unfolding  $C_6$ -def using  $\delta$ -gt-0  $\delta$ -lt-1
  by (intro iffD2[OF exp-le-cancel-iff] le-imp-neg-le mult-right-mono ln-ge-zero) auto
also have ... = exp  $(\ln (\delta / 2))$ 
  using  $\delta$ -gt-0 by (simp add: ln-div)
also have ... =  $\delta / 2$ 
  using  $\delta$ -gt-0 by simp
finally show ?thesis
  by simp
qed

end

```


unbundle *no-intro-cong-syntax*

end

9 Accuracy with cutoff

This section verifies that each of the l estimate have the required accuracy with high probability assuming as long as the cutoff is below $q-max$, generalizing the result from Section 7.

theory *Distributed-Distinct-Elements-Accuracy*

imports

Distributed-Distinct-Elements-Accuracy-Without-Cutoff

Distributed-Distinct-Elements-Cutoff-Level

begin

unbundle *intro-cong-syntax*

lemma (**in** *semilattice-set*) *Union:*

assumes *finite I I* $\neq \{\}$

assumes $\bigwedge i. i \in I \implies \text{finite } (Z\ i)$

assumes $\bigwedge i. i \in I \implies Z\ i \neq \{\}$

shows $F (\bigcup (Z\ ' I)) = F ((\lambda i. (F (Z\ i)))\ ' I)$

using *assms(1,2,3,4)*

proof (*induction I rule:finite-ne-induct*)

case (*singleton x*)

then show *?case* **by** *simp*

next

case (*insert x I*)

have $F (\bigcup (Z\ ' \text{insert } x\ I)) = F ((Z\ x) \cup (\bigcup (Z\ ' I)))$

by *simp*

also have $\dots = f (F (Z\ x)) (F (\bigcup (Z\ ' I)))$

using *insert* **by** (*intro union finite-UN-I*) *auto*

also have $\dots = f (F \{F (Z\ x)\}) (F ((\lambda i. F (Z\ i))\ ' I))$

using *insert(5,6)* **by** (*subst insert(4)*) *auto*

also have $\dots = F (\{F (Z\ x)\} \cup (\lambda i. F (Z\ i))\ ' I)$

using *insert(1,2)* **by** (*intro union[symmetric] finite-imageI*) *auto*

also have $\dots = F ((\lambda i. F (Z\ i))\ ' \text{insert } x\ I)$

by *simp*

finally show *?case* **by** *simp*

qed

This is similar to the existing *hom-Max-commute* with the crucial difference that it works even if the function is a homomorphism between distinct lattices. An example application is $\text{Max } (\text{int}\ ' A) = \text{int } (\text{Max } A)$.

lemma *hom-Max-commute'*:

assumes *finite A A* $\neq \{\}$

assumes $\bigwedge x\ y. x \in A \implies y \in A \implies \text{max } (f\ x) (f\ y) = f (\text{max } x\ y)$

shows $\text{Max } (f\ ' A) = f (\text{Max } A)$

using *assms* **by** (*induction A rule:finite-ne-induct*) *auto*

context *inner-algorithm-fix-A*

begin

definition t_c

where $t_c\ \psi\ \sigma = (\text{Max } ((\lambda j. \tau_1\ \psi\ A\ \sigma\ j + \sigma)\ ' \{..<b\})) - b\text{-exp} + 9$

definition s_c

where $s_c \psi \sigma = \text{nat } (t_c \psi \sigma)$

definition p_c

where $p_c \psi \sigma = \text{card } \{j \in \{..<b\}. \tau_1 \psi A \sigma j + \sigma \geq s_c \psi \sigma\}$

definition Y_c

where $Y_c \psi \sigma = 2 \wedge s_c \psi \sigma * \varrho\text{-inv } (p_c \psi \sigma)$

lemma $s_c\text{-eq-s}$:

assumes $(f,g,h) \in \text{sample-pro } \Psi$

assumes $\sigma \leq s f$

shows $s_c (f,g,h) \sigma = s f$

proof –

have $\text{int } (\text{Max } (f ' A)) - \text{int } b\text{-exp} + 9 \leq \text{int } (\text{Max } (f ' A)) - 26 + 9$

using $b\text{-exp-ge-26}$ by $(\text{intro } \text{add-mono } \text{diff-left-mono}) \text{ auto}$

also have $\dots \leq \text{int } (\text{Max } (f ' A))$ by simp

finally have $1:\text{int } (\text{Max } (f ' A)) - \text{int } b\text{-exp} + 9 \leq \text{int } (\text{Max } (f ' A))$

by simp

have $\sigma \leq \text{int } (s f)$ using $\text{assms}(2)$ by simp

also have $\dots = \text{max } 0 (t f)$

unfolding $s\text{-def}$ by simp

also have $\dots \leq \text{max } 0 (\text{int } (\text{Max } (f ' A)))$

unfolding $t\text{-def}$ using 1 by simp

also have $\dots = \text{int } (\text{Max } (f ' A))$

by simp

finally have $\sigma \leq \text{int } (\text{Max } (f ' A))$

by simp

hence $0:\text{int } \sigma - 1 \leq \text{int } (\text{Max } (f ' A))$

by simp

have $c:h \in \text{sample-pro } (\mathcal{H} k (C_7 * b^2) (\mathcal{N} b))$

using $\text{assms}(1)$ $\text{sample-set-}\Psi$ by auto

hence $h\text{-range}: h x < b$ for x

using $h\text{-range-1}$ by simp

have $(\text{MAX } j \in \{..<b\}. \tau_1 (f, g, h) A \sigma j + \text{int } \sigma) =$

$(\text{MAX } x \in \{..<b\}. \text{Max } (\{\text{int } (f a) \mid a. a \in A \wedge h (g a) = x\} \cup \{-1\} \cup \{\text{int } \sigma - 1\}))$

using $\text{fin-f}[OF \text{ assms}(1)]$ by $(\text{simp } \text{add:max-add-distrib-left } \text{max.commute } \tau_1\text{-def})$

also have $\dots = \text{Max } (\bigcup x < b. \{\text{int } (f a) \mid a. a \in A \wedge h (g a) = x\} \cup \{-1\} \cup \{\text{int } \sigma - 1\})$

using $\text{fin-f}[OF \text{ assms}(1)]$ $b\text{-ne}$ by $(\text{intro } \text{Max.Union}[\text{symmetric}]) \text{ auto}$

also have $\dots = \text{Max } (\{\text{int } (f a) \mid a. a \in A\} \cup \{-1, \text{int } \sigma - 1\})$

using $h\text{-range}$ by $(\text{intro } \text{arg-cong}[\text{where } f=\text{Max}]) \text{ auto}$

also have $\dots = \text{max } (\text{Max } (\text{int } ' f ' A)) (\text{int } \sigma - 1)$

using $A\text{-nonempty}$ fin-A unfolding Setcompr-eq-image image-image

by $(\text{subst } \text{Max.union}) \text{ auto}$

also have $\dots = \text{max } (\text{int } (\text{Max } (f ' A))) (\text{int } \sigma - 1)$

using fin-A $A\text{-nonempty}$ by $(\text{subst } \text{hom-Max-commute}') \text{ auto}$

also have $\dots = \text{int } (\text{Max } (f ' A))$

by $(\text{intro } \text{max-absorb1 } 0)$

finally have $(\text{MAX } j \in \{..<b\}. \tau_1 (f, g, h) A \sigma j + \text{int } \sigma) = \text{Max } (f ' A)$ by simp

thus $?thesis$

unfolding $s_c\text{-def}$ $t_c\text{-def}$ $s\text{-def}$ $t\text{-def}$ by simp

qed

lemma $p_c\text{-eq-p}$:

assumes $(f,g,h) \in \text{sample-pro } \Psi$
assumes $\sigma \leq s f$
shows $p_c (f,g,h) \sigma = p (f,g,h)$
proof –
have $\{j \in \{..<b\}. \text{int } (s f) \leq \max (\tau_0 (f, g, h) A j) (\text{int } \sigma - 1)\} =$
 $\{j \in \{..<b\}. \text{int } (s f) \leq \max (\tau_0 (f, g, h) A j) (-1)\}$
using *assms(2)* **unfolding** *le-max-iff-disj* **by** *simp*
thus *?thesis*
unfolding *p_c-def p-def s_c-eq-s[OF assms]*
by (*simp add:max-add-distrib-left τ₁-def del:τ₀.simps*)
qed

lemma *Y_c-eq-Y*:
assumes $(f,g,h) \in \text{sample-pro } \Psi$
assumes $\sigma \leq s f$
shows $Y_c (f,g,h) \sigma = Y (f,g,h)$
unfolding *Y_c-def Y-def s_c-eq-s[OF assms] p_c-eq-p[OF assms]* **by** *simp*

lemma *accuracy-single: measure Ψ {ψ. ∃σ ≤ q-max. |Y_c ψ σ - real X| > ε * X} ≤ 1/2⁴*
(is ?L ≤ ?R)

proof –
have *measure Ψ {ψ. ∃σ ≤ q-max. |Y_c ψ σ - real X| > ε * real X} ≤*
*measure Ψ {(f,g,h). |Y (f,g,h) - real X| > ε * real X ∨ s f < q-max}*
proof (*rule pmf-mono*)
fix ψ
assume $a:\psi \in \{\psi. \exists \sigma \leq q\text{-max}. \varepsilon * \text{real } X < |Y_c \psi \sigma - \text{real } X|\}$
assume $d:\psi \in \text{set-pmf } (\text{sample-pro } \Psi)$
obtain σ **where** $b:\sigma \leq q\text{-max}$ **and** $c:\varepsilon * \text{real } X < |Y_c \psi \sigma - \text{real } X|$
using *a* **by** *auto*
obtain $f g h$ **where** $\psi\text{-def: } \psi = (f,g,h)$ **by** (*metis prod-cases3*)
hence $e:(f,g,h) \in \text{sample-pro } \Psi$ **using** *d* **by** *simp*

show $\psi \in \{(f, g, h). \varepsilon * \text{real } X < |Y (f, g, h) - \text{real } X| \vee s f < q\text{-max}\}$
proof (*cases s f ≥ q-max*)
case *True*
hence $f:\sigma \leq s f$ **using** *b* **by** *simp*
have $\varepsilon * \text{real } X < |Y \psi - \text{real } X|$
using *Y_c-eq-Y[OF e f] c* **unfolding** $\psi\text{-def}$ **by** *simp*
then show *?thesis* **unfolding** $\psi\text{-def}$ **by** *simp*
next
case *False*
then show *?thesis* **unfolding** $\psi\text{-def}$ **by** *simp*
qed
qed
also have $\dots \leq 1/2^4$
using *accuracy-without-cutoff* **by** *simp*
finally show *?thesis* **by** *simp*
qed

lemma *estimate1-eq*:
assumes $j < l$
shows *estimate1* $(\tau_2 \omega A \sigma, \sigma) j = Y_c (\omega j) \sigma$ **(is ?L = ?R)**
proof –
define t **where** $t = \max 0 (\text{Max } ((\tau_2 \omega A \sigma j) ' \{..<b\}) + \sigma - \lfloor \log 2 b \rfloor + 9)$
define p **where** $p = \text{card } \{k. k \in \{..<b\} \wedge (\tau_2 \omega A \sigma j k) + \sigma \geq t\}$

have $0: \text{int } (\text{nat } x) = \max 0 x$ **for** x
by *simp*

have 1: $\lfloor \log 2 b \rfloor = b\text{-exp}$
 unfolding *b-def* by *simp*

have $b > 0$
 using *b-min* by *simp*
 hence 2: $\{..<b\} \neq \{\}$ by *auto*

have $t = \text{int } (\text{nat } (\text{Max } ((\tau_2 \omega A \sigma j) ' \{..<b\}) + \sigma - b\text{-exp} + 9))$
 unfolding *t-def 0 1* by (*rule refl*)
 also have $\dots = \text{int } (\text{nat } (\text{Max } ((\lambda x. x + \sigma) ' (\tau_2 \omega A \sigma j) ' \{..<b\}) - b\text{-exp} + 9))$
 by (*intro-cong* [$\sigma_1 \text{ int}, \sigma_1 \text{ nat}, \sigma_2(+), \sigma_2(-)$] *more:hom-Max-commute*) (*simp-all add:2*)
 also have $\dots = \text{int } (s_c (\omega j) \sigma)$
 using *assms*
 unfolding *s_c-def t_c-def τ_2 -def image-image* by *simp*
 finally have 3: $t = \text{int } (s_c (\omega j) \sigma)$
 by *simp*

have 4: $p = p_c (\omega j) \sigma$
 using *assms* unfolding *p-def p_c-def 3 τ_2 -def* by *simp*

have $?L = 2 \text{ powr } t * \ln (1-p/b) / \ln(1-1/b)$
 unfolding *estimate1.simps τ -def τ_3 -def*
 by (*simp only:t-def p-def Let-def*)
 also have $\dots = 2 \text{ powr } (s_c (\omega j) \sigma) * \varrho\text{-inv } p$
 unfolding 3 *ϱ -inv-def* by (*simp*)
 also have $\dots = ?R$
 unfolding *Y_c-def 3 4* by (*simp add:powr-realpow*)
 finally show *?thesis*
 by *blast*

qed

lemma *estimate-result-1*:

$\text{measure } \Omega \{ \omega. (\exists \sigma \leq q\text{-max}. \varepsilon * X < |\text{estimate } (\tau_2 \omega A \sigma, \sigma) - X|) \} \leq \delta/2$ (is $?L \leq ?R$)

proof –

define $I :: \text{real set}$ where $I = \{x. |x - \text{real } X| \leq \varepsilon * X\}$

define μ where $\mu = \text{measure } \Psi \{ \psi. \exists \sigma \leq q\text{-max}. Y_c \psi \sigma \notin I \}$

have *int-I*: *interval I*
 unfolding *interval-def I-def* by *auto*

have $\mu = \text{measure } \Psi \{ \psi. \exists \sigma \leq q\text{-max}. |Y_c \psi \sigma - \text{real } X| > \varepsilon * X \}$
 unfolding *μ -def I-def* by (*simp add:not-le*)
 also have $\dots \leq 1 / 2^4$
 by (*intro accuracy-single*)
 also have $\dots = 1 / 16$
 by *simp*
 finally have 1: $\mu \leq 1 / 16$ by *simp*

have $(\mu + \Lambda) \leq 1/16 + 1/16$
 unfolding *Λ -def* by (*intro add-mono 1*) *auto*
 also have $\dots \leq 1/8$
 by *simp*
 finally have 2: $(\mu + \Lambda) \leq 1/8$
 by *simp*

hence 0: $(\mu + \Lambda) \leq 1/2$
 by *simp*

have $\mu \geq 0$
unfolding μ -def **by** *simp*
hence $\exists: \mu + \Lambda > 0$
by (*intro add-nonneg-pos* Λ -gt-0)

have $?L = \text{measure } \Omega \{ \omega. (\exists \sigma \leq q\text{-max}. \varepsilon * X < |\text{median } l (\text{estimate1 } (\tau_2 \omega A \sigma, \sigma)) - X|) \}$
by *simp*
also have $\dots = \text{measure } \Omega \{ \omega. (\exists \sigma \leq q\text{-max}. \text{median } l (\text{estimate1 } (\tau_2 \omega A \sigma, \sigma)) \notin I) \}$
unfolding *I*-def **by** (*intro measure-pmf-cong*) *auto*
also have $\dots \leq \text{measure } \Omega \{ \omega. \text{real}(\text{card}\{i \in \{..<l\}. (\exists \sigma \leq q\text{-max}. Y_c (\omega i) \sigma \notin I)\}) \geq \text{real } l/2 \}$
proof (*rule pmf-mono*)
fix ω
assume $\omega \in \text{set-pmf } \Omega \omega \in \{ \omega. \exists \sigma \leq q\text{-max}. \text{median } l (\text{estimate1 } (\tau_2 \omega A \sigma, \sigma)) \notin I \}$
then obtain σ **where** σ -def: $\text{median } l (\text{estimate1 } (\tau_2 \omega A \sigma, \sigma)) \notin I \sigma \leq q\text{-max}$
by *auto*
hence $\text{real } l \leq \text{real } (2 * \text{card } \{i. i < l \wedge \text{estimate1 } (\tau_2 \omega A \sigma, \sigma) i \notin I\})$
by (*intro of-nat-mono median-est-rev*[*OF int-I*])
also have $\dots = 2 * \text{real } (\text{card } \{i \in \{..<l\}. \text{estimate1 } (\tau_2 \omega A \sigma, \sigma) i \notin I\})$
by *simp*
also have $\dots = 2 * \text{real } (\text{card } \{i \in \{..<l\}. Y_c (\omega i) \sigma \notin I\})$
using *estimate1-eq* **by** (*intro-cong* [σ_2 (*), σ_1 *of-nat*, σ_1 *card*] *more:restr-Collect-cong*) *auto*
also have $\dots \leq 2 * \text{real } (\text{card } \{i \in \{..<l\}. (\exists \sigma \leq q\text{-max}. Y_c (\omega i) \sigma \notin I)\})$
using σ -def(2) **by** (*intro mult-left-mono Nat.of-nat-mono card-mono*) *auto*
finally have $\text{real } l \leq 2 * \text{real } (\text{card } \{i \in \{..<l\}. (\exists \sigma \leq q\text{-max}. Y_c (\omega i) \sigma \notin I)\})$
by *simp*
thus $\omega \in \{ \omega. \text{real } l/2 \leq \text{real } (\text{card } \{i \in \{..<l\}. \exists \sigma \leq q\text{-max}. Y_c (\omega i) \sigma \notin I\}) \}$
by *simp*
qed

also have $\dots = \text{measure } \Omega \{ \omega. \text{real } (\text{card}\{i \in \{..<l\}. (\exists \sigma \leq q\text{-max}. Y_c (\omega i) \sigma \notin I)\}) \geq (1/2) * \text{real } l \}$
unfolding *p*-def **by** *simp*
also have $\dots \leq \exp (- \text{real } l * ((1/2) * \ln (1 / (\mu + \Lambda)) - 2 * \exp (- 1)))$
using 0 **unfolding** μ -def **by** (*intro walk-tail-bound l-gt-0* Λ -gt-0) *auto*
also have $\dots = \exp (- (\text{real } l * ((1/2) * \ln (1 / (\mu + \Lambda)) - 2 * \exp (- 1))))$
by *simp*
also have $\dots \leq \exp (- (\text{real } l * ((1/2) * \ln 8 - 2 * \exp (- 1))))$
using 2 3 *l-gt-0* **by** (*intro iffD2*[*OF exp-le-cancel-iff*] *le-imp-neg-le mult-left-mono diff-mono*)
(auto simp add:divide-simps)
also have $\dots \leq \exp (- (\text{real } l * (1/4)))$
by (*intro iffD2*[*OF exp-le-cancel-iff*] *le-imp-neg-le mult-left-mono of-nat-0-le-iff*)
(approximation 5)
also have $\dots \leq \exp (- (C_6 * \ln (2 / \delta) * (1/4)))$
by (*intro iffD2*[*OF exp-le-cancel-iff*] *le-imp-neg-le mult-right-mono l-lbound*) *auto*
also have $\dots = \exp (- \ln (2 / \delta))$
unfolding C_6 -def **by** *simp*
also have $\dots = ?R$
using δ -gt-0 **by** (*subst ln-inverse*[*symmetric*]) *auto*
finally show *?thesis*
by *simp*
qed

theorem *estimate-result*:

$\text{measure } \Omega \{ \omega. |\text{estimate } (\tau \omega A) - X| > \varepsilon * X \} \leq \delta$
(is ?L ≤ ?R)

proof –

let $?P = \text{measure } \Omega$

have $?L \leq ?P \{ \omega. (\exists \sigma \leq q\text{-max}. \varepsilon * \text{real } X < |\text{estimate } (\tau_2 \omega A \sigma, \sigma) - \text{real } X|) \vee q \omega A > q\text{-max} \}$

```

    unfolding  $\tau$ -def  $\tau_3$ -def not-le[symmetric]
    by (intro pmf-mono) auto
    also have ...  $\leq$  ?P { $\omega$ . ( $\exists \sigma \leq q$ -max.  $\varepsilon * \text{real } X < |\text{estimate } (\tau_2 \ \omega \ A \ \sigma, \sigma) - X|$ )} + ?P { $\omega$ .  $q \ \omega \ A >$ 
 $q$ -max}
    by (intro pmf-add) auto
    also have ...  $\leq$   $\delta/2 + \delta/2$ 
    by (intro add-mono cutoff-level estimate-result-1)
    also have ... =  $\delta$ 
    by simp
    finally show ?thesis
    by simp
qed
end

```

```

lemma (in inner-algorithm) estimate-result:
  assumes  $A \subseteq \{..<n\}$   $A \neq \{\}$ 
  shows  $\text{measure } \Omega \ \{\omega. |\text{estimate } (\tau \ \omega \ A) - \text{real } (\text{card } A)| > \varepsilon * \text{real } (\text{card } A)\} \leq \delta$  (is ?L  $\leq$  ?R)
proof -
  interpret inner-algorithm-fix-A
  using assms by unfold-locales auto
  have ?L =  $\text{measure } \Omega \ \{\omega. |\text{estimate } (\tau \ \omega \ A) - X| > \varepsilon * X\}$ 
  unfolding X-def by simp
  also have ...  $\leq$  ?R
  by (intro estimate-result)
  finally show ?thesis
  by simp
qed

```

```

unbundle no-intro-cong-syntax

```

```

end

```

10 Outer Algorithm

This section introduces the final solution with optimal size space usage. Internally it relies on the inner algorithm described in Section 6, depending on the parameters n , ε and δ it either uses the inner algorithm directly or if ε^{-1} is larger than $\ln n$ it runs $\frac{\varepsilon^{-1}}{\ln \ln n}$ copies of the inner algorithm (with the modified failure probability $\frac{1}{\ln n}$) using an expander to select its seeds. The theorems below verify that the probability that the relative accuracy of the median of the copies is too large is below ε .

```

theory Distributed-Distinct-Elements-Outer-Algorithm

```

```

  imports

```

```

    Distributed-Distinct-Elements-Accuracy

```

```

    Prefix-Free-Code-Combinators.Prefix-Free-Code-Combinators

```

```

    Frequency-Moments.Landau-Ext

```

```

    Landau-Symbols.Landau-More

```

```

begin

```

```

unbundle intro-cong-syntax

```

The following are non-asymptotic hard bounds on the space usage for the sketches and seeds respectively. The end of this section contains a proof that the sum is asymptotically in $\mathcal{O}(\ln(\varepsilon^{-1})\delta^{-1} + \ln n)$.

```

definition state-space-usage =  $(\lambda(n, \varepsilon, \delta). 2^{40} * (\ln(1/\delta) + 1) / \varepsilon^2 + \log 2 (\log 2 n + 3))$ 

```

```

definition seed-space-usage =  $(\lambda(n, \varepsilon, \delta). 2^{30} + 2^{23} * \ln n + 48 * (\log 2 (1/\varepsilon) + 16)^2 + 336 * \ln (1/\delta))$ 

```

```

locale outer-algorithm =
  fixes  $n :: \text{nat}$ 
  fixes  $\delta :: \text{real}$ 
  fixes  $\varepsilon :: \text{real}$ 
  assumes  $n\text{-gt-0}: n > 0$ 
  assumes  $\delta\text{-gt-0}: \delta > 0$  and  $\delta\text{-lt-1}: \delta < 1$ 
  assumes  $\varepsilon\text{-gt-0}: \varepsilon > 0$  and  $\varepsilon\text{-lt-1}: \varepsilon < 1$ 
begin

definition  $n_0$  where  $n_0 = \max (\text{real } n) (\exp (\exp 5))$ 
definition stage-two where  $\text{stage-two} = (\delta < (1 / \ln n_0))$ 
definition  $\delta_i :: \text{real}$  where  $\delta_i = (\text{if } \text{stage-two} \text{ then } (1 / \ln n_0) \text{ else } \delta)$ 
definition  $m :: \text{nat}$  where  $m = (\text{if } \text{stage-two} \text{ then } \text{nat } \lceil 4 * \ln (1 / \delta) / \ln (\ln n_0) \rceil \text{ else } 1)$ 
definition  $\alpha$  where  $\alpha = (\text{if } \text{stage-two} \text{ then } (1 / \ln n_0) \text{ else } 1)$ 

lemma m-lbound:
  assumes stage-two
  shows  $m \geq 4 * \ln (1 / \delta) / \ln (\ln n_0)$ 
proof –
  have  $m = \text{real } (\text{nat } \lceil 4 * \ln (1 / \delta) / \ln (\ln n_0) \rceil)$ 
    using assms unfolding m-def by simp
  also have  $\dots \geq 4 * \ln (1 / \delta) / \ln (\ln n_0)$ 
    by linarith
  finally show ?thesis by simp
qed

lemma n-lbound:
   $n_0 \geq \exp (\exp 5) \ln n_0 \geq \exp 5 \ 5 \leq \ln (\ln n_0) \ln n_0 > 1 \ n_0 > 1$ 
proof –
  show  $0 : n_0 \geq \exp (\exp 5)$ 
    unfolding n0-def by simp
  have  $(1 :: \text{real}) \leq \exp (\exp 5)$ 
    by (approximation 5)
  hence  $n_0 \geq 1$ 
    using  $0$  by argo
  thus  $1 : \ln n_0 \geq \exp 5$ 
    using  $0$  by (intro iffD2[OF ln-ge-iff]) auto
  moreover have  $1 < \exp (5 :: \text{real})$ 
    by (approximation 5)
  ultimately show  $2 : \ln n_0 > 1$ 
    by argo
  show  $5 \leq \ln (\ln n_0)$ 
    using  $1 \ 2$  by (subst ln-ge-iff) simp
  have  $(1 :: \text{real}) < \exp (\exp 5)$ 
    by (approximation 5)
  thus  $n_0 > 1$ 
    using  $0$  by argo
qed

lemma  $\delta 1\text{-gt-0}: 0 < \delta_i$ 
  using n-lbound(4)  $\delta\text{-gt-0}$  unfolding  $\delta_i\text{-def}$ 
  by (cases stage-two) simp-all

lemma  $\delta 1\text{-lt-1}: \delta_i < 1$ 
  using n-lbound(4)  $\delta\text{-lt-1}$  unfolding  $\delta_i\text{-def}$ 
  by (cases stage-two) simp-all

```

lemma *m-gt-0-aux*:
assumes *stage-two*
shows $1 \leq \ln (1 / \delta) / \ln (\ln n_0)$
proof –
have $\ln n_0 \leq 1 / \delta$
using *n-lbound(4)*
using *assms unfolding pos-le-divide-eq[OF δ -gt-0] stage-two-def*
by (*simp add:divide-simps ac-simps*)
hence $\ln (\ln n_0) \leq \ln (1 / \delta)$
using *n-lbound(4) δ -gt-0* **by** (*intro iffD2[OF ln-le-cancel-iff] divide-pos-pos*) *auto*
thus $1 \leq \ln (1 / \delta) / \ln (\ln n_0)$
using *n-lbound(3)*
by (*subst pos-le-divide-eq*) *auto*
qed

lemma *m-gt-0: m > 0*
proof (*cases stage-two*)
case *True*
have $0 < 4 * \ln (1 / \delta) / \ln (\ln n_0)$
using *m-gt-0-aux[OF True]* **by** *simp*
also have $\dots \leq m$
using *m-lbound[OF True]* **by** *simp*
finally have $0 < \text{real } m$
by *simp*
then show *?thesis* **by** *simp*
next
case *False*
then show *?thesis* **unfolding** *m-def* **by** *simp*
qed

lemma *α -gt-0: $\alpha > 0$*
using *n-lbound(4) unfolding α -def*
by (*cases stage-two*) *auto*

lemma *α -le-1: $\alpha \leq 1$*
using *n-lbound(4) unfolding α -def*
by (*cases stage-two*) *simp-all*

sublocale *I: inner-algorithm n $\delta_i \varepsilon$*
unfolding *inner-algorithm-def* **using** *n-gt-0 ε -gt-0 ε -lt-1 δ 1-gt-0 δ 1-lt-1* **by** *auto*

abbreviation Θ **where** $\Theta \equiv \mathcal{E} \ m \ \alpha \ I.\Omega$

lemma Θ : $m > 0 \ \alpha > 0$ **using** *α -gt-0 m-gt-0* **by** *auto*

type-synonym *state = inner-algorithm.state list*

fun *single* :: *nat* \Rightarrow *nat* \Rightarrow *state* **where**
single $\vartheta \ x = \text{map } (\lambda j. I.\text{single } (\text{pro-select } \Theta \ \vartheta \ j) \ x) \ [0..<m]$

fun *merge* :: *state* \Rightarrow *state* \Rightarrow *state* **where**
merge $x \ y = \text{map } (\lambda(x,y). I.\text{merge } x \ y) \ (\text{zip } x \ y)$

fun *estimate* :: *state* \Rightarrow *real* **where**
estimate $x = \text{median } m \ (\lambda i. I.\text{estimate } (x \ ! \ i))$

definition ν :: *nat* \Rightarrow *nat set* \Rightarrow *state*
where $\nu \ \vartheta \ A = \text{map } (\lambda i. I.\tau \ (\text{pro-select } \Theta \ \vartheta \ i) \ A) \ [0..<m]$

The following three theorems verify the correctness of the algorithm. The term τ is a mathematical description of the sketch for a given subset, while *local.single*, *local.merge* are the actual functions that compute the sketches.

theorem *merge-result*: $\text{merge } (\nu \omega A) (\nu \omega B) = \nu \omega (A \cup B)$ (is ?L = ?R)

proof –

have $0: \text{zip } [0..<m] [0..<m] = \text{map } (\lambda x. (x,x)) [0..<m]$ **for** m
by (*induction m, auto*)

have $?L = \text{map } (\lambda x. I.\text{merge } (I.\tau (\text{pro-select } \Theta \omega x) A) (I.\tau (\text{pro-select } \Theta \omega x) B)) [0..<m]$
unfolding $\nu\text{-def}$

by (*simp add:zip-map-map 0 comp-def case-prod-beta*)

also have $\dots = \text{map } (\lambda x. I.\tau (\text{pro-select } \Theta \omega x) (A \cup B)) [0..<m]$

by (*intro map-cong refl I.merge-result expander-pro-range[OF Θ]*)

also have $\dots = ?R$

unfolding $\nu\text{-def}$ **by** *simp*

finally show *?thesis* **by** *simp*

qed

theorem *single-result*: $\text{single } \omega x = \nu \omega \{x\}$ (is ?L = ?R)

proof –

have $?L = \text{map } (\lambda j. I.\text{single } (\text{pro-select } \Theta \omega j) x) [0..<m]$

by (*simp del:I.single.simps*)

also have $\dots = ?R$

unfolding $\nu\text{-def}$ **by** (*intro map-cong I.single-result expander-pro-range[OF Θ] auto*)

finally show *?thesis* **by** *simp*

qed

theorem *estimate-result*:

assumes $A \subseteq \{..<n\}$ $A \neq \{\}$

defines $p \equiv (\text{pmf-of-set } \{..<\text{pro-size } \Theta\})$

shows $\text{measure } p \{ \omega. |\text{estimate } (\nu \omega A) - \text{real } (\text{card } A)| > \varepsilon * \text{real } (\text{card } A) \} \leq \delta$ (is ?L \leq ?R)

proof (*cases stage-two*)

case *True*

define I **where** $I = \{x. |x - \text{real } (\text{card } A)| \leq \varepsilon * \text{real } (\text{card } A)\}$

have *int-I*: *interval I*

unfolding *interval-def I-def* **by** *auto*

define μ **where** $\mu = \text{measure } I.\Omega \{ \omega. I.\text{estimate } (I.\tau \omega A) \notin I \}$

have $0: \mu + \alpha > 0$

unfolding $\mu\text{-def}$

by (*intro add-nonneg-pos $\alpha\text{-gt-0}$ auto*)

have $\mu \leq \delta_i$

unfolding $\mu\text{-def I-def}$ **using** *I.estimate-result[OF assms(1,2)]*

by (*simp add: not-le del:I.estimate.simps*)

also have $\dots = 1/\ln n_0$

using *True* **unfolding** $\delta_i\text{-def}$ **by** *simp*

finally have $\mu \leq 1/\ln n_0$ **by** *simp*

hence $\mu + \alpha \leq 1/\ln n_0 + 1/\ln n_0$

unfolding $\alpha\text{-def}$ **using** *True* **by** (*intro add-mono auto*)

also have $\dots = 2/\ln n_0$

by *simp*

finally have $1: \mu + \alpha \leq 2 / \ln n_0$

by *simp*

hence $2: \ln n_0 \leq 2 / (\mu + \alpha)$

using 0 *n-lbound* **by** (*simp add:field-simps*)

have $\mu + \alpha \leq 2/\ln n_0$
by (*intro 1*)
also have $\dots \leq 2/\exp 5$
using *n-lbound* **by** (*intro divide-left-mono*) *simp-all*
also have $\dots \leq 1/2$
by (*approximation 5*)
finally have $3:\mu + \alpha \leq 1/2$ **by** *simp*

have $4: 2 * \ln 2 + 8 * \exp (-1) \leq (5::\text{real})$
by (*approximation 5*)

have $?L = \text{measure } p \{ \omega. \text{median } m (\lambda i. I.\text{estimate } (\nu \omega A ! i)) \notin I \}$
unfolding *I-def* **by** (*simp add:not-le*)
also have $\dots \leq$
 $\text{measure } p \{ \vartheta. \text{real } (\text{card } \{ i \in \{..<m\}. I.\text{estimate } (I.\tau (\text{pro-select } \Theta \vartheta i) A) \notin I \}) \geq \text{real } m/2 \}$
proof (*rule pmf-mono*)
fix ϑ **assume** $\vartheta \in \text{set-pmf } p$
assume $a:\vartheta \in \{ \omega. \text{median } m (\lambda i. I.\text{estimate } (\nu \omega A ! i)) \notin I \}$
hence $\text{real } m \leq \text{real } (2 * \text{card } \{ i. i < m \wedge I.\text{estimate } (\nu \vartheta A ! i) \notin I \})$
by (*intro of-nat-mono median-est-rev int-I*) *auto*
also have $\dots = 2 * \text{real } (\text{card } \{ i \in \{..<m\}. I.\text{estimate } (\nu \vartheta A ! i) \notin I \})$
by *simp*
also have $\dots = 2 * \text{real } (\text{card } \{ i \in \{..<m\}. I.\text{estimate } (I.\tau (\text{pro-select } \Theta \vartheta i) A) \notin I \})$
unfolding $\nu\text{-def}$ **by** (*intro-cong* [σ_2 (*), σ_1 *of-nat*, σ_1 *card*] *more:restr-Collect-cong*)
(*simp del:I.estimate.simps*)
finally have $\text{real } m \leq 2 * \text{real } (\text{card } \{ i \in \{..<m\}. I.\text{estimate } (I.\tau (\text{pro-select } \Theta \vartheta i) A) \notin I \})$
by *simp*
thus $\vartheta \in \{ \vartheta. \text{real } m / 2 \leq \text{real } (\text{card } \{ i \in \{..<m\}. I.\text{estimate } (I.\tau (\text{pro-select } \Theta \vartheta i) A) \notin I \}) \}$
by *simp*
qed

also have $\dots = \text{measure } \Theta \{ \vartheta. \text{real } (\text{card } \{ i \in \{..<m\}. I.\text{estimate } (I.\tau (\vartheta i) A) \notin I \}) \geq (1/2) * \text{real } m \}$
unfolding *sample-pro-alt p-def* **by** (*simp del:I.estimate.simps*)
also have $\dots \leq \exp (-\text{real } m * ((1/2) * \ln (1 / (\mu + \alpha)) - 2 * \exp (-1)))$
using 3 *m-gt-0* α *gt-0* **unfolding** $\mu\text{-def}$ **by** (*intro walk-tail-bound*) *force+*
also have $\dots \leq \exp (-\text{real } m * ((1/2) * \ln (\ln n_0 / 2) - 2 * \exp (-1)))$
using 0 2 3 *n-lbound*
by (*intro iffD2*[*OF exp-le-cancel-iff*] *mult-right-mono mult-left-mono-neg*[**where** $c = -\text{real } m$] *diff-mono mult-left-mono iffD2*[*OF ln-le-cancel-iff*]) (*simp-all*)
also have $\dots = \exp (-\text{real } m * (\ln (\ln n_0) / 2 - (\ln 2/2 + 2 * \exp (-1))))$
using *n-lbound* **by** (*subst ln-div*) (*simp-all add:algebra-simps*)
also have $\dots \leq \exp (-\text{real } m * (\ln (\ln n_0) / 2 - (\ln (\ln (\exp(\exp 5))) / 4)))$
using 4
by (*intro iffD2*[*OF exp-le-cancel-iff*] *mult-left-mono-neg*[**where** $c = -\text{real } m$] *diff-mono*) *simp-all*
also have $\dots \leq \exp (-\text{real } m * (\ln (\ln n_0) / 2 - (\ln (\ln n_0) / 4)))$
using *n-lbound*
by (*intro iffD2*[*OF exp-le-cancel-iff*] *mult-left-mono-neg*[**where** $c = -\text{real } m$] *diff-mono*) *simp-all*
also have $\dots = \exp (-\text{real } m * (\ln (\ln n_0) / 4))$
by (*simp add:algebra-simps*)
also have $\dots \leq \exp (- (4 * \ln (1 / \delta) / \ln (\ln n_0)) * (\ln (\ln n_0) / 4))$
using *m-lbound*[*OF True*] *n-lbound*
by (*intro iffD2*[*OF exp-le-cancel-iff*] *mult-right-mono divide-nonneg-pos*) *simp-all*
also have $\dots = \exp (-\ln (1 / \delta))$
using *n-lbound* **by** *simp*
also have $\dots = \delta$
using δ *gt-0* **by** (*subst ln-inverse*[*symmetric*]) *auto*

finally show *?thesis* **by simp**
next
case *False*
have *m-eq*: $m = 1$
unfolding *m-def* **using** *False* **by simp**
hence $?L = \text{measure } p \{ \omega. \varepsilon * \text{real}(\text{card } A) < |I.\text{estimate}(\nu \omega A ! 0) - \text{real}(\text{card } A)| \}$
unfolding *estimate.simps* *m-eq* *median-def* **by simp**
also have $\dots = \text{measure } p \{ \omega. \varepsilon * \text{card } A < |I.\text{estimate}(I.\tau(\text{pro-select } \Theta \omega 0) A) - \text{real}(\text{card } A)| \}$
unfolding *ν -def* *m-eq* **by** (*simp del: I.estimate.simps*)
also have $\dots = \text{measure } \Theta \{ \omega. \varepsilon * \text{real}(\text{card } A) < |I.\text{estimate}(I.\tau(\omega 0) A) - \text{real}(\text{card } A)| \}$
unfolding *sample-pro-alt* *p-def* **by** (*simp del: I.estimate.simps*)
also have $\dots =$
 $\text{measure}(\text{map-pmf}(\lambda \vartheta. \vartheta 0) \Theta) \{ \omega. \varepsilon * \text{real}(\text{card } A) < |I.\text{estimate}(I.\tau \omega A) - \text{real}(\text{card } A)| \}$
by simp
also have $\dots = \text{measure } I.\Omega \{ \omega. \varepsilon * \text{real}(\text{card } A) < |I.\text{estimate}(I.\tau \omega A) - \text{real}(\text{card } A)| \}$
using *m-eq* **by** (*subst expander-uniform-property[OF Θ]*) *auto*
also have $\dots \leq \delta_i$
by (*intro I.estimate-result[OF assms(1,2)]*)
also have $\dots = ?R$
unfolding *δ_i -def* **using** *False* **by simp**
finally show *?thesis*
by simp
qed

The function *encode-state* can represent states as bit strings. This enables verification of the space usage.

definition *encode-state*
where *encode-state* = $Lf_e I.\text{encode-state } m$

lemma *encode-state: is-encoding encode-state*
unfolding *encode-state-def*
by (*intro fixed-list-encoding I.encode-state*)

lemma *state-bit-count*:

$\text{bit-count}(\text{encode-state}(\nu \omega A)) \leq \text{state-space-usage}(\text{real } n, \varepsilon, \delta)$
(is ?L ≤ ?R)

proof –

have 0 : $\text{length}(\nu \omega A) = m$
unfolding *ν -def* **by simp**
have $?L = (\sum x \leftarrow \nu \omega A. \text{bit-count}(I.\text{encode-state } x))$
using 0 **unfolding** *encode-state-def* *fixed-list-bit-count* **by simp**
also have $\dots = (\sum x \leftarrow [0..<m]. \text{bit-count}(I.\text{encode-state}(I.\tau(\text{pro-select } \Theta \omega x) A)))$
unfolding *ν -def* **by** (*simp add: comp-def*)
also have $\dots \leq (\sum x \leftarrow [0..<m]. \text{ereal}(2^{36} * (\ln(1/\delta_i) + 1)/\varepsilon^2 + \log 2(\log 2(\text{real } n) + 3)))$
using *I.state-bit-count* **by** (*intro sum-list-mono I.state-bit-count expander-pro-range[OF Θ]*)
also have $\dots = \text{ereal}(\text{real } m * (2^{36} * (\ln(1/\delta_i) + 1)/\varepsilon^2 + \log 2(\log 2(\text{real } n) + 3)))$
unfolding *sum-list-triv-ereal* **by simp**
also have $\dots \leq 2^{40} * (\ln(1/\delta) + 1) / \varepsilon^2 + \log 2(\log 2 n + 3)$ **(is ?L1 ≤ ?R1)**

proof (*cases stage-two*)

case *True*

have $\lceil 4 * \ln(1/\delta) / \ln(\ln n_0) \rceil \leq 4 * \ln(1/\delta) / \ln(\ln n_0) + 1$
by simp

also have $\dots \leq 4 * \ln(1/\delta) / \ln(\ln n_0) + \ln(1/\delta) / \ln(\ln n_0)$

using *m-gt-0-aux[OF True]* **by** (*intro add-mono*) *auto*

also have $\dots = 5 * \ln(1/\delta) / \ln(\ln n_0)$ **by simp**

finally have 3 : $\lceil 4 * \ln(1/\delta) / \ln(\ln n_0) \rceil \leq 5 * \ln(1/\delta) / \ln(\ln n_0)$
by simp

have 4: $0 \leq \log 2 (\log 2 (\text{real } n) + 3)$
using *n-gt-0*
by (*intro iffD2[OF zero-le-log-cancel-iff] add-nonneg-pos*) *auto*

have 5: $1 / \ln 2 + 3 / \exp 5 \leq \exp (1::\text{real}) \quad 1.2 / \ln 2 \leq (2::\text{real})$
by (*approximation 5*)+

have $\log 2 (\log 2 (\text{real } n) + 3) \leq \log 2 (\log 2 n_0 + 3)$
using *n-gt-0* **by** (*intro iffD2[OF log-le-cancel-iff] add-mono add-nonneg-pos*
iffD2[OF zero-le-log-cancel-iff]) (*simp-all add:n₀-def*)

also have $\dots = \ln (\ln n_0 / \ln 2 + 3) / \ln 2$
unfolding *log-def* **by** *simp*

also have $\dots \leq \ln (\ln n_0 / \ln 2 + (3 / \exp 5) * \ln n_0) / \ln 2$
using *n-lbound* **by** (*intro divide-right-mono iffD2[OF ln-le-cancel-iff] add-mono add-nonneg-pos*)
(simp-all add:divide-simps)

also have $\dots = \ln (\ln n_0 * (1 / \ln 2 + 3 / \exp 5)) / \ln 2$
by (*simp add:algebra-simps*)

also have $\dots \leq \ln (\ln n_0 * \exp 1) / \ln 2$
using *n-lbound* **by** (*intro divide-right-mono iffD2[OF ln-le-cancel-iff] add-mono*
mult-left-mono 5 Rings.mult-pos-pos add-pos-nonneg) *auto*

also have $\dots = (\ln (\ln n_0) + 1) / \ln 2$
using *n-lbound* **by** (*subst ln-mult*) *simp-all*

also have $\dots \leq (\ln (\ln n_0) + 0.2 * \ln (\ln n_0)) / \ln 2$
using *n-lbound* **by** (*intro divide-right-mono add-mono*) *auto*

also have $\dots = (1.2 / \ln 2) * \ln (\ln n_0)$
by *simp*

also have $\dots \leq 2 * \ln (\ln n_0)$
using *n-lbound* **by** (*intro mult-right-mono 5*) *simp*

finally have $\log 2 (\log 2 (\text{real } n) + 3) \leq 2 * \ln (\ln n_0)$
by *simp*

hence 6: $\log 2 (\log 2 (\text{real } n) + 3) / \ln (\ln n_0) \leq 2$
using *n-lbound* **by** (*subst pos-divide-le-eq*) *simp-all*

have ?L1 = $\text{real}(\text{nat } \lceil 4 * \ln (1/\delta) / \ln (\ln n_0) \rceil) * (2^{36} * (\ln (\ln n_0) + 1) / \varepsilon^2 + \log 2 (\log 2 (\text{real } n) + 3))$
using *True* **unfolding** *m-def* *δ_i -def* **by** *simp*

also have $\dots = \lceil 4 * \ln (1/\delta) / \ln (\ln n_0) \rceil * (2^{36} * (\ln (\ln n_0) + 1) / \varepsilon^2 + \log 2 (\log 2 (\text{real } n) + 3))$
using *m-gt-0-aux[OF True]* **by** (*subst of-nat-nat*) *simp-all*

also have $\dots \leq (5 * \ln (1/\delta) / \ln (\ln n_0)) * (2^{36} * (\ln (\ln n_0) + 1) / \varepsilon^2 + \log 2 (\log 2 (\text{real } n) + 3))$
using *n-lbound(3)* *ε -gt-0 4* **by** (*intro ereal-mono mult-right-mono*
add-nonneg-nonneg divide-nonneg-pos mult-nonneg-nonneg 3) *simp-all*

also have $\dots \leq (5 * \ln (1/\delta) / \ln (\ln n_0)) * ((2^{36} + 2^{36}) * \ln (\ln n_0) / \varepsilon^2 + \log 2 (\log 2 (\text{real } n) + 3))$
using *n-lbound* *δ -gt-0* *δ -lt-1*
by (*intro ereal-mono mult-left-mono add-mono divide-right-mono divide-nonneg-pos*) *auto*

also have $\dots = 5 * (2^{37}) * \ln (1/\delta) / \varepsilon^2 + (5 * \ln (1/\delta)) * (\log 2 (\log 2 (\text{real } n) + 3) / \ln (\ln n_0))$
using *n-lbound* **by** (*simp add:algebra-simps*)

also have $\dots \leq 5 * (2^{37}) * \ln (1/\delta) / \varepsilon^2 + (5 * \ln (1/\delta)) * 2$
using *δ -gt-0* *δ -lt-1* **by** (*intro add-mono ereal-mono order.refl mult-left-mono 6*) *auto*

also have $\dots = 5 * (2^{37}) * \ln (1/\delta) / \varepsilon^2 + 5 * 2 * \ln (1/\delta) / 1$
by *simp*

also have $\dots \leq 5 * (2^{37}) * \ln (1/\delta) / \varepsilon^2 + 5 * 2 * \ln (1/\delta) / \varepsilon^2$
using *ε -gt-0* *ε -lt-1* *δ -gt-0* *δ -lt-1*
by (*intro add-mono ereal-mono divide-left-mono Rings.mult-pos-pos power-le-one*) *auto*

also have $\dots = (5 * (2^{37} + 2)) * (\ln (1/\delta) + 0) / \varepsilon^2 + 0$
by (*simp add:algebra-simps*)

also have $\dots \leq 2^{40} * (\ln (1/\delta) + 1) / \varepsilon^2 + \log 2 (\log 2 (\text{real } n) + 3)$
using *ε -gt-0* *ε -lt-1* *δ -gt-0* *δ -lt-1* *n-gt-0* **by** (*intro add-mono ereal-mono divide-right-mono*)

$mult\text{-}right\text{-}mono$ iffD2[OF zero-le-log-cancel-iff] add-nonneg-pos) auto
finally show ?thesis **by** simp
next
case False
have ?L1 = $2^{36} * (\ln(1/\delta) + 1) / \varepsilon^2 + \log 2 (\log 2 (\text{real } n) + 3)$
using False **unfolding** $\delta_i\text{-def}$ $m\text{-def}$ **by** simp
also have ... \leq ?R1
using $\varepsilon\text{-gt-0}$ $\varepsilon\text{-lt-1}$ $\delta\text{-gt-0}$ $\delta\text{-lt-1}$
by (intro ereal-mono add-mono divide-right-mono mult-right-mono add-nonneg-nonneg) auto
finally show ?thesis **by** simp
qed
finally show ?thesis
unfolding state-space-usage-def **by** simp
qed

Encoding function for the seeds which are just natural numbers smaller than *pro-size* Θ .

definition encode-seed
where encode-seed = Nb_e (pro-size Θ)

lemma encode-seed:
is-encoding encode-seed
unfolding encode-seed-def **by** (intro bounded-nat-encoding)

lemma random-bit-count:
assumes $\omega <$ pro-size Θ
shows bit-count (encode-seed ω) \leq seed-space-usage (real n , ε , δ)
(is ?L \leq ?R)

proof –

have 0: pro-size $\Theta > 0$ **by** (intro pro-size-gt-0)
have 1: pro-size $I.\Omega > 0$ **by** (intro pro-size-gt-0)

have $(55 + 60 * \ln(\ln n_0))^3 \leq (180 + 60 * \ln(\ln n_0))^3$
using n-lbound **by** (intro power-mono add-mono) auto
also have ... = $180^3 * (1 + \ln(\ln n_0) / \text{real } 3)^3$
unfolding power-mult-distrib[symmetric] **by** simp
also have ... $\leq 180^3 * \exp(\ln(\ln n_0))$
using n-lbound **by** (intro mult-left-mono exp-ge-one-plus-x-over-n-power-n) auto
also have ... = $180^3 * \ln n_0$
using n-lbound **by** (subst exp-ln) auto
also have ... $\leq 180^3 * \max(\ln n) (\ln(\exp(\exp 5)))$
using n-gt-0 **unfolding** $n_0\text{-def}$ **by** (subst ln-max-swap) auto
also have ... $\leq 180^3 * (\ln n + \exp 5)$
using n-gt-0 **unfolding** ln-exp **by** (intro mult-left-mono) auto
finally have 2: $(55 + 60 * \ln(\ln n_0))^3 \leq 180^3 * \ln n + 180^3 * \exp 5$
by simp

have 3: $(1::\text{real}) + 180^3 * \exp 5 \leq 2^{30} (4::\text{real}) / \ln 2 + 180^3 \leq 2^{23}$
by (approximation 10)+

have ?L = ereal (real (floorlog 2 (pro-size $\Theta - 1$)))
using assms **unfolding** encode-seed-def bounded-nat-bit-count **by** simp
also have ... \leq ereal (real (floorlog 2 (pro-size Θ)))
by (intro ereal-mono Nat.of-nat-mono floorlog-mono) auto
also have ... = ereal (1 + of-int [log 2 (real (pro-size Θ))])
using 0 **unfolding** floorlog-def **by** simp
also have ... \leq ereal (1 + log 2 (real (pro-size Θ)))
by (intro add-mono ereal-mono) auto
also have ... = 1 + log 2 (real (pro-size $I.\Omega$) * $(2^4)^{((m-1) * \text{nat} [\ln \alpha / \ln 0.95])}$)

unfolding *expander-pro-size*[*OF* Θ] **by** *simp*
also have $\dots = 1 + \log 2 (\text{real } (\text{pro-size } I.\Omega) * 2^{\wedge} (4 * (m - 1) * \text{nat } \lceil \ln \alpha / \ln 0.95 \rceil))$
unfolding *power-mult* **by** *simp*
also have $\dots = 1 + \log 2 (\text{real } (\text{pro-size } I.\Omega)) + (4 * (m - 1) * \text{nat } \lceil \ln \alpha / \ln 0.95 \rceil)$
using *1* **by** (*subst log-mult*) *simp-all*
also have $\dots \leq 1 + \log 2 (2 \text{ powr } (4 * \log 2 n + 48 * (\log 2 (1/\varepsilon) + 16)^2 + (55 + 60 * \ln (1/\delta_i))^{\wedge} 3)) +$
 $(4 * (m - 1) * \text{nat } \lceil \ln \alpha / \ln 0.95 \rceil)$
using *1* **by** (*intro ereal-mono add-mono iffD2*[*OF log-le-cancel-iff*] *I.random-bit-count*) *auto*
also have $\dots = 1 + 4 * \log 2 n + 48 * (\log 2 (1/\varepsilon) + 16)^2 + (55 + 60 * \ln (1/\delta_i))^{\wedge} 3 + (4 * (m - 1) * \text{nat } \lceil \ln$
 $\alpha / \ln 0.95 \rceil)$
by (*subst log-powr-cancel*) *auto*
also have $\dots \leq 2^{\wedge} 30 + 2^{\wedge} 23 * \ln n + 48 * (\log 2 (1/\varepsilon) + 16)^2 + 336 * \ln (1/\delta)$ (**is** ?*L1* \leq ?*R1*)
proof (*cases stage-two*)
case *True*

have $-1 < (0::\text{real})$ **by** *simp*
also have $\dots \leq \ln \alpha / \ln 0.95$
using *α -gt-0 α -le-1* **by** (*intro divide-nonpos-neg*) *auto*
finally have *4*: $-1 < \ln \alpha / \ln 0.95$ **by** *simp*

have *5*: $-1 / \ln 0.95 \leq (20::\text{real})$
by (*approximation 10*)

have $(4 * (m - 1) * \text{nat } \lceil \ln \alpha / \ln 0.95 \rceil) = 4 * (\text{real } m - 1) * \text{of-int } \lceil \ln \alpha / \ln 0.95 \rceil$
using *4 m-gt-0* **unfolding** *of-nat-mult* **by** (*subst of-nat-nat*) *auto*
also have $\dots \leq 4 * (\text{real } m - 1) * (\ln \alpha / \ln 0.95 + 1)$
using *m-gt-0* **by** (*intro mult-left-mono*) *auto*
also have $\dots = 4 * (\text{real } m - 1) * (-\ln (\ln n_0) / \ln 0.95 + 1)$
using *n-lbound True* **unfolding** *α -def*
by (*subst ln-inverse*[*symmetric*]) (*simp-all add:inverse-eq-divide*)
also have $\dots = 4 * (\text{real } m - 1) * (\ln (\ln n_0) * (-1 / \ln 0.95) + 1)$
by *simp*
also have $\dots \leq 4 * (\text{real } m - 1) * (\ln (\ln n_0) * 20 + 1)$
using *n-lbound m-gt-0* **by** (*intro mult-left-mono add-mono 5*) *auto*
also have $\dots = 4 * (\text{real } (\text{nat } \lceil 4 * \ln (1 / \delta) / \ln (\ln n_0) \rceil) - 1) * (\ln (\ln n_0) * 20 + 1)$
using *True* **unfolding** *m-def* **by** *simp*
also have $\dots = 4 * (\text{real-of-int } \lceil 4 * \ln (1 / \delta) / \ln (\ln n_0) \rceil - 1) * (\ln (\ln n_0) * 20 + 1)$
using *m-gt-0-aux*[*OF True*] **by** (*subst of-nat-nat*) *simp-all*
also have $\dots \leq 4 * (4 * \ln (1 / \delta) / \ln (\ln n_0)) * (\ln (\ln n_0) * 20 + 1)$
using *n-lbound* **by** (*intro mult-left-mono mult-right-mono*) *auto*
also have $\dots \leq 4 * (4 * \ln (1 / \delta) / \ln (\ln n_0)) * (\ln (\ln n_0) * 20 + \ln (\ln n_0))$
using *δ -gt-0 δ -lt-1 n-lbound*
by (*intro mult-left-mono mult-right-mono add-mono divide-nonneg-pos Rings.mult-nonneg-nonneg*)
simp-all
also have $\dots = 336 * \ln (1 / \delta)$
using *n-lbound* **by** *simp*
finally have *6*: $4 * (m - 1) * \text{nat } \lceil \ln \alpha / \ln 0.95 \rceil \leq 336 * \ln (1/\delta)$
by *simp*

have ?*L1* $= 1 + 4 * \log 2 n + 48 * (\log 2 (1/\varepsilon) + 16)^2 + (55 + 60 * \ln (\ln n_0))^{\wedge} 3 + (4 * (m - 1) * \text{nat } \lceil \ln$
 $\alpha / \ln 0.95 \rceil)$
using *True* **unfolding** *δ_i -def* **by** *simp*
also have $\dots \leq 1 + 4 * \log 2 n + 48 * (\log 2 (1/\varepsilon) + 16)^2 + (180^{\wedge} 3 * \ln n + 180^{\wedge} 3 * \exp 5) + 336 * \ln$
 $(1/\delta)$
by (*intro add-mono 6 2 ereal-mono order.refl*)
also have $\dots = (1 + 180^{\wedge} 3 * \exp 5) + (4 / \ln 2 + 180^{\wedge} 3) * \ln n + 48 * (\log 2 (1/\varepsilon) + 16)^2 + 336 * \ln$
 $(1/\delta)$
by (*simp add:log-def algebra-simps*)

```

also have ... ≤ 230 + 223*ln n + 48*(log 2(1/ε)+16)2 + 336 * ln (1/δ)
  using n-gt-0 by (intro add-mono ereal-mono 3 order.refl mult-right-mono) auto
finally show ?thesis by simp
next
case False
hence 1 / δ ≤ ln n0
  using δ-gt-0 n-lbound
  unfolding stage-two-def not-less by (simp add:divide-simps ac-simps)
hence 7: ln (1 / δ) ≤ ln (ln n0)
  using n-lbound δ-gt-0 δ-lt-1
  by (intro iffD2[OF ln-le-cancel-iff]) auto

have 8: 0 ≤ 336*ln (1/δ)
  using δ-gt-0 δ-lt-1 by auto

have ?L1 = 1 + 4 * log 2 (real n) + 48 * (log 2 (1 / ε) + 16)2 + (55 + 60 * ln (1 / δ)) 3
  using False unfolding δi-def m-def by simp
also have ... ≤ 1 + 4 * log 2 (real n) + 48 * (log 2 (1 / ε) + 16)2 + (55 + 60 * ln (ln
n0))3
  using δ-gt-0 δ-lt-1
  by (intro add-mono order.refl ereal-mono power-mono mult-left-mono add-nonneg-nonneg 7)
auto
also have ... ≤ 1+4*log 2(real n)+48*(log 2 (1 / ε)+16)2+(1803*ln (real n) + 180 3 *
exp 5)
  by (intro add-mono ereal-mono 2 order.refl)
also have ... = (1+1803*exp 5)+ (4/ln 2 + 1803)*ln n+48*(log 2(1/ε)+16)2+ 0
  by (simp add:log-def algebra-simps)
also have ... ≤ 230 + 223*ln n + 48*(log 2(1/ε)+16)2 + 336*ln (1/δ)
  using n-gt-0 by (intro add-mono ereal-mono 3 order.refl mult-right-mono 8) auto
finally show ?thesis by simp
qed
also have ... = seed-space-usage (real n, ε, δ)
  unfolding seed-space-usage-def by simp
finally show ?thesis by simp
qed

```

The following is an alternative form expressing the correctness and space usage theorems. If x is expression formed by *local.single* and *local.merge* operations. Then x requires *state-space-usage* (real n , ε , δ) bits to encode and *estimate* x approximates the count of the distinct universe elements in the expression.

For example:

estimate (*local.merge* (*local.single* ω 1) (*local.merge* (*local.single* ω 5) (*local.single* ω 1))) approximates the cardinality of $\{1, 5, 1\}$ i.e. 2.

```
datatype sketch-tree = Single nat | Merge sketch-tree sketch-tree
```

```
fun eval :: nat ⇒ sketch-tree ⇒ state
where
  eval  $\omega$  (Single  $x$ ) = single  $\omega$   $x$  |
  eval  $\omega$  (Merge  $x$   $y$ ) = merge (eval  $\omega$   $x$ ) (eval  $\omega$   $y$ )
```

```
fun sketch-tree-set :: sketch-tree ⇒ nat set
where
  sketch-tree-set (Single  $x$ ) = { $x$ } |
  sketch-tree-set (Merge  $x$   $y$ ) = sketch-tree-set  $x$  ∪ sketch-tree-set  $y$ 
```

```
theorem correctness:
fixes  $X$ 
```

```

assumes sketch-tree-set  $t \subseteq \{..<n\}$ 
defines  $p \equiv \text{pmf-of-set } \{..<\text{pro-size } \Theta\}$ 
defines  $X \equiv \text{real } (\text{card } (\text{sketch-tree-set } t))$ 
shows  $\text{measure } p \{ \omega. |\text{estimate } (\text{eval } \omega t) - X| > \varepsilon * X \} \leq \delta$  (is  $?L \leq ?R$ )
proof –
define  $A$  where  $A = \text{sketch-tree-set } t$ 
have  $X\text{-eq}$ :  $X = \text{real } (\text{card } A)$ 
  unfolding  $X\text{-def } A\text{-def}$  by simp

have  $0$ :  $\text{eval } \omega t = \nu \omega A$  for  $\omega$ 
  unfolding  $A\text{-def}$  using single-result merge-result
  by (induction t) (auto simp del:merge.simps single.simps)

have  $1$ :  $A \subseteq \{..<n\}$ 
  using assms(1) unfolding  $A\text{-def}$  by blast

have  $2$ :  $A \neq \{\}$ 
  unfolding  $A\text{-def}$  by (induction t) auto

show  $?thesis$ 
  unfolding  $0 X\text{-eq } p\text{-def}$  by (intro estimate-result 1 2)
qed

```

theorem *space-usage*:

```

assumes  $\omega < \text{pro-size } \Theta$ 
shows
   $\text{bit-count } (\text{encode-state } (\text{eval } \omega t)) \leq \text{state-space-usage } (\text{real } n, \varepsilon, \delta)$  (is  $?A$ )
   $\text{bit-count } (\text{encode-seed } \omega) \leq \text{seed-space-usage } (\text{real } n, \varepsilon, \delta)$  (is  $?B$ )

```

proof–

```

define  $A$  where  $A = \text{sketch-tree-set } t$ 

have  $0$ :  $\text{eval } \omega t = \nu \omega A$  for  $\omega$ 
  unfolding  $A\text{-def}$  using single-result merge-result
  by (induction t) (auto simp del:merge.simps single.simps)

show  $?A$ 
  unfolding  $0$  by (intro state-bit-count)
show  $?B$ 
  using random-bit-count[OF assms] by simp
qed

```

end

The functions *state-space-usage* and *seed-space-usage* are exact bounds on the space usage for the state and the seed. The following establishes asymptotic bounds with respect to the limit $n, \delta^{-1}, \varepsilon^{-1} \rightarrow \infty$.

context
begin

Some local notation to ease proofs about the asymptotic space usage of the algorithm:

```

private definition  $n\text{-of} :: \text{real} \times \text{real} \times \text{real} \Rightarrow \text{real}$  where  $n\text{-of} = (\lambda(n, \varepsilon, \delta). n)$ 
private definition  $\delta\text{-of} :: \text{real} \times \text{real} \times \text{real} \Rightarrow \text{real}$  where  $\delta\text{-of} = (\lambda(n, \varepsilon, \delta). \delta)$ 
private definition  $\varepsilon\text{-of} :: \text{real} \times \text{real} \times \text{real} \Rightarrow \text{real}$  where  $\varepsilon\text{-of} = (\lambda(n, \varepsilon, \delta). \varepsilon)$ 

```

```

private abbreviation  $F :: (\text{real} \times \text{real} \times \text{real}) \text{ filter}$ 
where  $F \equiv (\text{at-top} \times_F \text{at-right } 0 \times_F \text{at-right } 0)$ 

```

private lemma *var-simps*:

$n\text{-of} = \text{fst}$
 $\varepsilon\text{-of} = (\lambda x. \text{fst} (\text{snd } x))$
 $\delta\text{-of} = (\lambda x. \text{snd} (\text{snd } x))$
unfolding $n\text{-of-def}$ $\varepsilon\text{-of-def}$ $\delta\text{-of-def}$ **by** (*auto simp add:case-prod-beta*)

private lemma evt-n : *eventually* $(\lambda x. n\text{-of } x \geq n)$ F
unfolding var-simps **by** (*intro eventually-prod1' eventually-prod2' eventually-ge-at-top*)
(simp add:prod-filter-eq-bot)

private lemma evt-n-1 : $\forall_F x \text{ in } F. 0 \leq \ln (n\text{-of } x)$
by (*intro eventually-mono[OF evt-n[of 1]] ln-ge-zero*) *simp*

private lemma evt-n-2 : $\forall_F x \text{ in } F. 0 \leq \ln (\ln (n\text{-of } x))$
using *order-less-le-trans[OF exp-gt-zero]*
by (*intro eventually-mono[OF evt-n[of exp 1]] ln-ge-zero iffD2[OF ln-ge-iff]*) *auto*

private lemma $\text{evt-}\varepsilon$: *eventually* $(\lambda x. 1/\varepsilon\text{-of } x \geq \varepsilon \wedge \varepsilon\text{-of } x > 0)$ F
unfolding var-simps **by** (*intro eventually-prod1' eventually-prod2' eventually-conj*
real-inv-at-right-0-inf eventually-at-right-less) (*simp-all add:prod-filter-eq-bot*)

private lemma $\text{evt-}\delta$: *eventually* $(\lambda x. 1/\delta\text{-of } x \geq \delta \wedge \delta\text{-of } x > 0)$ F
unfolding var-simps **by** (*intro eventually-prod1' eventually-prod2' eventually-conj*
real-inv-at-right-0-inf eventually-at-right-less) (*simp-all add:prod-filter-eq-bot*)

private lemma $\text{evt-}\delta\text{-1}$: $\forall_F x \text{ in } F. 0 \leq \ln (1 / \delta\text{-of } x)$
by (*intro eventually-mono[OF evt- δ [of 1]] ln-ge-zero*) *simp*

theorem *asymptotic-state-space-complexity*:
 $\text{state-space-usage} \in O[F](\lambda(n, \varepsilon, \delta). \ln (1/\delta)/\varepsilon^2 + \ln (\ln n))$
(is - $\in O[?F](?rhs)$)

proof –

have 0 : $(\lambda x. 1) \in O[?F](\lambda x. \ln (1 / \delta\text{-of } x))$
using *order-less-le-trans[OF exp-gt-zero]*
by (*intro landau-o.big-mono eventually-mono[OF evt- δ [of exp 1]]*)
(auto intro!: iffD2[OF ln-ge-iff] simp add:abs-ge-iff)

have 1 : $(\lambda x. 1) \in O[?F](\lambda x. \ln (n\text{-of } x))$
using *order-less-le-trans[OF exp-gt-zero]*
by (*intro landau-o.big-mono eventually-mono[OF evt-n[of exp 1]]*)
(auto intro!: iffD2[OF ln-ge-iff] simp add:abs-ge-iff)

have $\lambda x. ((\ln (1/\delta\text{-of } x)+1) * (1/\varepsilon\text{-of } x)^2) \in O[?F](\lambda x. \ln(1/\delta\text{-of } x) * (1/\varepsilon\text{-of } x)^2)$
by (*intro landau-o.mult sum-in-bigo 0*) *simp-all*

hence 2 : $(\lambda x. 2^4 * ((\ln (1/\delta\text{-of } x)+1) * (1/\varepsilon\text{-of } x)^2)) \in O[?F](\lambda x. \ln(1/\delta\text{-of } x) * (1/\varepsilon\text{-of } x)^2)$
unfolding *cmult-in-bigo-iff* **by** *simp*

have 3 : $(1::\text{real}) \leq \exp 2$
by (*approximation 5*)

have $(\lambda x. \ln (n\text{-of } x) / \ln 2 + 3) \in O[?F](\lambda x. \ln (n\text{-of } x))$
using 1 **by** (*intro sum-in-bigo*) *simp-all*
hence $(\lambda x. \ln (\ln (n\text{-of } x) / \ln 2 + 3)) \in O[?F](\lambda x. \ln (\ln (n\text{-of } x)))$
using *order-less-le-trans[OF exp-gt-zero]* *order-trans[OF 3]*
by (*intro landau-ln-2[where a=2] eventually-mono[OF evt-n[of exp 2]]*)
(auto intro!: iffD2[OF ln-ge-iff] add-nonneg-nonneg divide-nonneg-pos)
hence 4 : $(\lambda x. \log 2 (\log 2 (n\text{-of } x) + 3)) \in O[?F](\lambda x. \ln(\ln(n\text{-of } x)))$
unfolding *log-def* **by** *simp*

have 5: $\forall_F x \text{ in } ?F. 0 \leq \ln (1 / \delta\text{-of } x) * (1 / \varepsilon\text{-of } x)^2$
by (*intro eventually-mono*[*OF eventually-conj*[*OF evt- δ -1 evt- ε [of 1]]]]) *auto*

have *state-space-usage* = ($\lambda x. \text{state-space-usage } (n\text{-of } x, \varepsilon\text{-of } x, \delta\text{-of } x)$)
by (*simp add:case-prod-beta' n-of-def δ -of-def ε -of-def*)
also have ... = ($\lambda x. 2^{\wedge} 40 * ((\ln (1 / (\delta\text{-of } x)) + 1) * (1 / \varepsilon\text{-of } x)^2) + \log 2 (\log 2 (n\text{-of } x) + 3)$)
unfolding *state-space-usage-def* **by** (*simp add:divide-simps*)
also have ... $\in O[?F](\lambda x. \ln (1 / \delta\text{-of } x) * (1 / \varepsilon\text{-of } x)^2 + \ln (\ln (n\text{-of } x)))$
by (*intro landau-sum 2 4 5 evt-n-2*)
also have ... = $O[?F](?rhs)$
by (*simp add:case-prod-beta' n-of-def δ -of-def ε -of-def divide-simps*)
finally show *?thesis* **by** *simp*
qed*

theorem *asymptotic-seed-space-complexity:*

seed-space-usage $\in O[F](\lambda(n, \varepsilon, \delta). \ln (1 / \delta) + \ln (1 / \varepsilon)^{\wedge} 2 + \ln n)$
(is - $\in O[?F](?rhs)$)

proof –

have 0: $\forall_F x \text{ in } ?F. 0 \leq (\ln (1 / \varepsilon\text{-of } x))^2$
by *simp*

have 1: $\forall_F x \text{ in } ?F. 0 \leq \ln (1 / \delta\text{-of } x) + (\ln (1 / \varepsilon\text{-of } x))^2$
by (*intro eventually-mono*[*OF eventually-conj*[*OF evt- δ -1 0*]]] *add-nonneg-nonneg*) *auto*

have 2: $(\lambda x. 1) \in O[?F](\lambda x. \ln (1 / \varepsilon\text{-of } x))$
using *order-less-le-trans*[*OF exp-gt-zero*]
by (*intro landau-o.big-mono eventually-mono*[*OF evt- ε [of exp 1]*]])
(auto intro!:iffD2[*OF ln-ge-iff*] *simp add:abs-ge-iff*)

have $(\lambda x. 1) \in O[at\text{-top} \times_F at\text{-right } 0 \times_F at\text{-right } 0](\lambda x. \ln (n\text{-of } x))$
using *order-less-le-trans*[*OF exp-gt-zero*]
by (*intro landau-o.big-mono eventually-mono*[*OF evt-n*[of exp 1]])
(auto intro!:iffD2[*OF ln-ge-iff*] *simp add:abs-ge-iff*)

hence 3: $(\lambda x. 1) \in O[?F](\lambda x. \ln (1 / \delta\text{-of } x) + (\ln (1 / \varepsilon\text{-of } x))^2 + \ln (n\text{-of } x))$
by (*intro landau-sum-2 1 evt-n-1 0 evt- δ -1*) *simp*

have 4: $(\lambda x. \ln (n\text{-of } x)) \in O[?F](\lambda x. \ln (1 / \delta\text{-of } x) + (\ln (1 / \varepsilon\text{-of } x))^2 + \ln (n\text{-of } x))$
by (*intro landau-sum-2 1 evt-n-1*) *simp*

have $(\lambda x. \log 2 (1 / \varepsilon\text{-of } x) + 16) \in O[?F](\lambda x. \ln (1 / \varepsilon\text{-of } x))$
using 2 **unfolding** *log-def* **by** (*intro sum-in-bigo*) *simp-all*

hence 5: $(\lambda x. (\log 2 (1 / \varepsilon\text{-of } x) + 16)^2) \in O[?F](\lambda x. \ln (1 / \delta\text{-of } x) + (\ln (1 / \varepsilon\text{-of } x))^2)$
using 0 **unfolding** *power2-eq-square* **by** (*intro landau-sum-2 landau-o.mult evt- δ -1*) *simp-all*

have 6: $(\lambda x. (\log 2 (1 / \varepsilon\text{-of } x) + 16)^2) \in O[?F](\lambda x. \ln (1 / \delta\text{-of } x) + (\ln (1 / \varepsilon\text{-of } x))^2 + \ln (n\text{-of } x))$
by (*intro landau-sum-1*[*OF - - 5*] 1 *evt-n-1*)

have 7: $(\lambda x. \ln (1 / \delta\text{-of } x)) \in O[?F](\lambda x. \ln (1 / \delta\text{-of } x) + (\ln (1 / \varepsilon\text{-of } x))^2 + \ln (n\text{-of } x))$
by (*intro landau-sum-1 1 evt- δ -1 0 evt-n-1*) *simp*

have *seed-space-usage* = ($\lambda x. \text{seed-space-usage } (n\text{-of } x, \varepsilon\text{-of } x, \delta\text{-of } x)$)
by (*simp add:case-prod-beta' n-of-def δ -of-def ε -of-def*)

also have ... = ($\lambda x. 2^{\wedge} 30 + 2^{\wedge} 23 * \ln (n\text{-of } x) + 48 * (\log 2 (1 / (\varepsilon\text{-of } x)) + 16)^2 + 336 * \ln (1 / \delta\text{-of } x)$)

unfolding *seed-space-usage-def* **by** (*simp add:divide-simps*)

also have ... $\in O[?F](\lambda x. \ln (1 / \delta\text{-of } x) + \ln (1 / \varepsilon\text{-of } x)^{\wedge} 2 + \ln (n\text{-of } x))$
using 3 4 6 7 **by** (*intro sum-in-bigo*) *simp-all*

also have ... = $O[?F](?rhs)$

by (*simp add:case-prod-beta' n-of-def δ -of-def ε -of-def*)

finally show *?thesis* **by** *simp*

qed

definition *space-usage* $x = \text{state-space-usage } x + \text{seed-space-usage } x$

theorem *asymptotic-space-complexity*:

space-usage $\in O[\text{at-top} \times_F \text{at-right } 0 \times_F \text{at-right } 0](\lambda(n, \varepsilon, \delta). \ln(1/\delta)/\varepsilon^2 + \ln n)$

proof –

let $?f1 = (\lambda x. \ln(1/\delta\text{-of } x) * (1/\varepsilon\text{-of } x^2) + \ln(\ln(n\text{-of } x)))$

let $?f2 = (\lambda x. \ln(1/\delta\text{-of } x) + \ln(1/\varepsilon\text{-of } x)^2 + \ln(n\text{-of } x))$

have $0: \forall_F x \text{ in } F. 0 \leq (1 / (\varepsilon\text{-of } x)^2)$

unfolding *var-simps* **by** (*intro eventually-prod1' eventually-prod2' eventually-inv*)
(*simp-all add:prod-filter-eq-bot eventually-nonzero-simps*)

have $1: \forall_F x \text{ in } F. 0 \leq \ln(1 / \delta\text{-of } x) * (1 / (\varepsilon\text{-of } x)^2)$

by (*intro eventually-mono[OF eventually-conj[OF evt- δ -1 0]] mult-nonneg-nonneg*) *auto*

have $2: \forall_F x \text{ in } F. 0 \leq \ln(1 / \delta\text{-of } x) * (1 / (\varepsilon\text{-of } x)^2) + \ln(\ln(n\text{-of } x))$

by (*intro eventually-mono[OF eventually-conj[OF 1 evt-n-2]] add-nonneg-nonneg*) *auto*

have $3: \forall_F x \text{ in } F. 0 \leq \ln(1 / (\varepsilon\text{-of } x)^2)$

unfolding *power-one-over[symmetric]*
by (*intro eventually-mono[OF evt- ε [of 1]] ln-ge-zero*) *simp*

have $4: \forall_F x \text{ in } F. 0 \leq \ln(1 / \delta\text{-of } x) + (\ln(1 / \varepsilon\text{-of } x))^2 + \ln(n\text{-of } x)$

by (*intro eventually-mono[OF eventually-conj[OF evt-n-1 eventually-conj[OF evt- δ -1 3]]]*)
add-nonneg-nonneg) *auto*

have $5: (\lambda-. 1) \in O[F](\lambda x. 1 / (\varepsilon\text{-of } x)^2)$

unfolding *var-simps* **by** (*intro bigo-prod-1 bigo-prod-2 bigo-inv*)
(*simp-all add:power-divide prod-filter-eq-bot*)

have $6: (\lambda-. 1) \in O[F](\lambda x. \ln(1 / \delta\text{-of } x))$

unfolding *var-simps*
by (*intro bigo-prod-1 bigo-prod-2 bigo-inv*) (*simp-all add:prod-filter-eq-bot*)

have $7: \text{state-space-usage} \in O[F](\lambda x. \ln(1 / \delta\text{-of } x) * (1 / (\varepsilon\text{-of } x)^2) + \ln(\ln(n\text{-of } x)))$

using *asymptotic-state-space-complexity* **unfolding** $\delta\text{-of-def } \varepsilon\text{-of-def } n\text{-of-def}$
by (*simp add:case-prod-beta'*)

have $8: \text{seed-space-usage} \in O[F](\lambda x. \ln(1 / \delta\text{-of } x) + (\ln(1 / \varepsilon\text{-of } x))^2 + \ln(n\text{-of } x))$

using *asymptotic-seed-space-complexity* **unfolding** $\delta\text{-of-def } \varepsilon\text{-of-def } n\text{-of-def}$
by (*simp add:case-prod-beta'*)

have $9: (\lambda x. \ln(n\text{-of } x)) \in O[F](\lambda x. \ln(1 / \delta\text{-of } x) * (1 / (\varepsilon\text{-of } x)^2) + \ln(n\text{-of } x))$

by (*intro landau-sum-2 evt-n-1 1*) *simp*

have $(\lambda x. (\ln(1 / \varepsilon\text{-of } x))^2) \in O[F](\lambda x. 1 / \varepsilon\text{-of } x^2)$

unfolding *var-simps*
by (*intro bigo-prod-1 bigo-prod-2 bigo-inv*) (*simp-all add:power-divide prod-filter-eq-bot*)

hence $10: (\lambda x. (\ln(1 / \varepsilon\text{-of } x))^2) \in O[F](\lambda x. \ln(1 / \delta\text{-of } x) * (1 / \varepsilon\text{-of } x^2) + \ln(n\text{-of } x))$

by (*intro landau-sum-1 evt-n-1 1 landau-o.big-mult-1' 6*)

have $11: (\lambda x. \ln(1 / \delta\text{-of } x)) \in O[F](\lambda x. \ln(1 / \delta\text{-of } x) * (1 / \varepsilon\text{-of } x^2) + \ln(n\text{-of } x))$

by (*intro landau-sum-1 evt-n-1 1 landau-o.big-mult-1 5*) *simp*

have $12: (\lambda x. \ln(1/\delta\text{-of } x) * (1/\varepsilon\text{-of } x^2)) \in O[F](\lambda x. \ln(1/\delta\text{-of } x) * (1/\varepsilon\text{-of } x^2) + \ln(n\text{-of } x))$

by (*intro landau-sum-1 1 evt-n-1*) *simp*

have $(\lambda x. \ln(\ln(n\text{-of } x))) \in O[F](\lambda x. \ln(n\text{-of } x))$

unfolding *var-simps* **by** (*intro bigo-prod-1 bigo-prod-2*) (*simp-all add:prod-filter-eq-bot*)

hence 13: $(\lambda x. \ln (\ln (n\text{-of } x))) \in O[F](\lambda x. \ln (1 / \delta\text{-of } x) * (1 / \varepsilon\text{-of } x^2) + \ln (n\text{-of } x))$
by (*intro landau-sum-2 evt-n-1 1*)

have *space-usage* = $(\lambda x. \text{state-space-usage } x + \text{seed-space-usage } x)$

unfolding *space-usage-def* **by** *simp*

also have $\dots \in O[F](\lambda x. ?f1\ x + ?f2\ x)$

by (*intro landau-sum 2 4 7 8*)

also have $\dots \subseteq O[F](\lambda x. \ln (1 / \delta\text{-of } x) * (1/\varepsilon\text{-of } x^2) + \ln (n\text{-of } x))$

by (*intro landau-o.big.subsetI sum-in-bigo 9 10 11 12 13*)

also have $\dots = O[F](\lambda(n, \varepsilon, \delta). \ln (1/\delta)/\varepsilon^2 + \ln n)$

unfolding *delta-of-def* *epsilon-of-def* *n-of-def*

by (*simp add:case-prod-beta^*)

finally show *?thesis* **by** *simp*

qed

end

unbundle *no-intro-cong-syntax*

end

References

- [1] N. Alon, Y. Matias, and M. Szegedy. The space complexity of approximating the frequency moments. *Journal of Computer and System Sciences*, 58(1):137–147, 1999.
- [2] Z. Bar-Yossef, T. S. Jayram, R. Kumar, D. Sivakumar, and L. Trevisan. Counting distinct elements in a data stream. In *Randomization and Approximation Techniques in Computer Science*, pages 1–10. Springer Berlin Heidelberg, 2002.
- [3] J. Błasiok. Optimal streaming and tracking distinct elements with high probability. *ACM Trans. Algorithms*, 16(1):3:1–3:28, 2020.
- [4] P. Flajolet and G. Nigel Martin. Probabilistic counting algorithms for data base applications. *Journal of Computer and System Sciences*, 31(2):182–209, 1985.
- [5] P. B. Gibbons and S. Tirthapura. Estimating simple functions on the union of data streams. In *Proceedings of the Thirteenth Annual ACM Symposium on Parallel Algorithms and Architectures*, SPAA '01, pages 281–291, 2001.
- [6] V. Guruswami, C. Umans, and S. Vadhan. Unbalanced expanders and randomness extractors from parvaresh–vardy codes. *J. ACM*, 56(4), jul 2009.
- [7] D. M. Kane, J. Nelson, and D. P. Woodruff. An optimal algorithm for the distinct elements problem. In *Proceedings of the Twenty-Ninth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, PODS '10, pages 41–52, New York, 2010.
- [8] E. Karayel. Finite fields. *Archive of Formal Proofs*, June 2022. https://isa-afp.org/entries/Finite_Fields.html, Formal proof development.
- [9] E. Karayel. Formalization of randomized approximation algorithms for frequency moments. *Archive of Formal Proofs*, April 2022. https://isa-afp.org/entries/Frequency_Moments.html, Formal proof development.
- [10] E. Karayel. Expander graphs. *Archive of Formal Proofs*, March 2023. https://isa-afp.org/entries/Expander_Graphs.html, Formal proof development.
- [11] D. Woodruff. Optimal space lower bounds for all frequency moments. In *Proceedings of the Fifteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '04, pages 167–175, USA, 2004. Society for Industrial and Applied Mathematics.