

Measure and Probability Theory

January 18, 2026

Contents

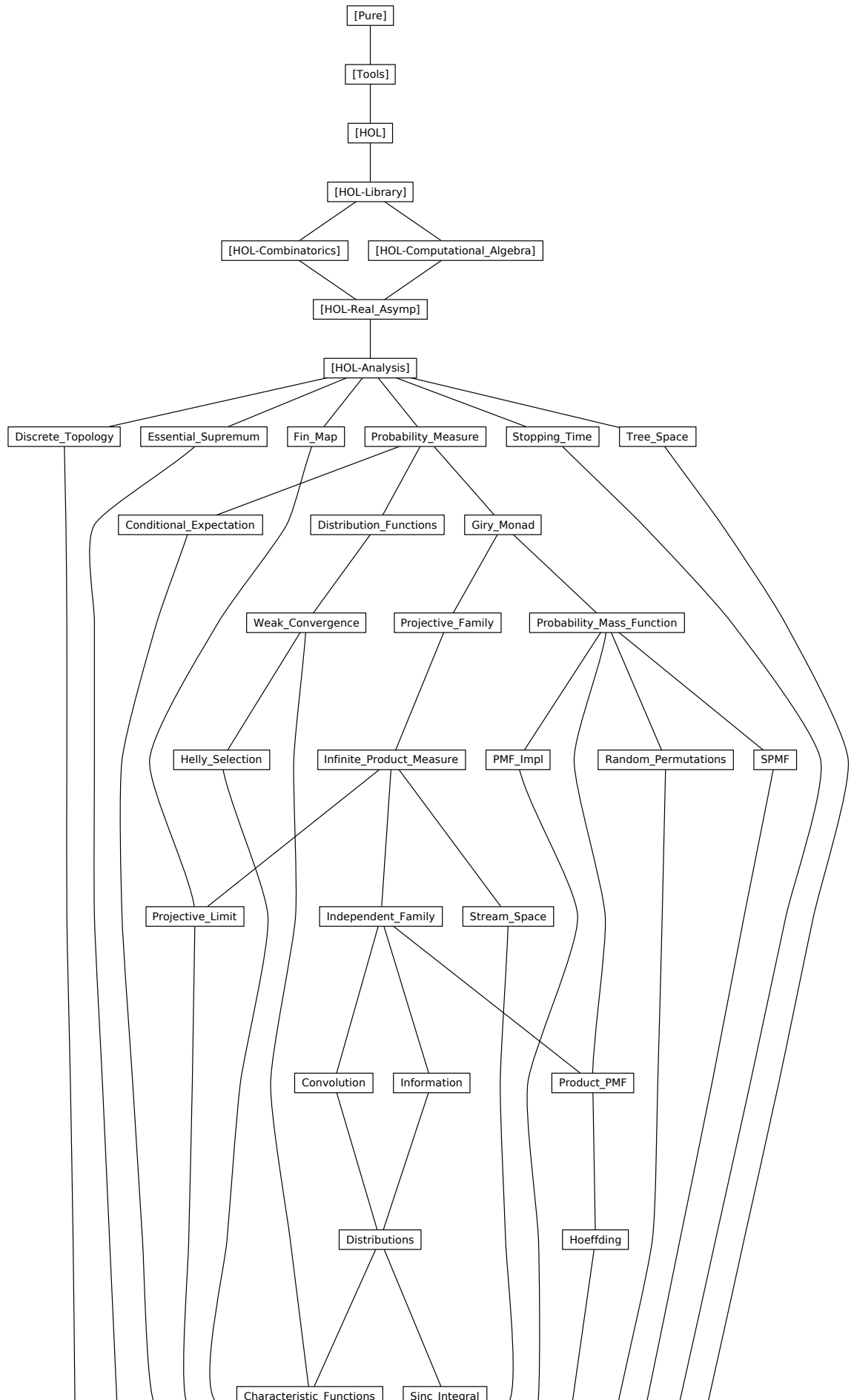
1	Probability measure	3
1.1	Introduce binder for probability	6
1.2	Distributions	9
2	Distribution Functions	16
2.1	Properties of cdf's	17
2.2	Uniqueness	18
3	Weak Convergence of Functions and Distributions	20
4	Weak Convergence of Functions	20
5	Weak Convergence of Distributions	20
6	Skorohod's theorem	20
7	The Giry monad	23
7.1	Sub-probability spaces	23
7.2	Properties of “return”	27
7.3	Join	30
7.4	Giry monad on probability spaces	36
8	Projective Family	39
9	Infinite Product Measure	44
9.1	Sequence space	46
10	Independent families of events, event sets, and random variables	48
11	Convolution Measure	55

12 Information theory	58
12.1 Information theory	58
12.2 Kullback–Leibler divergence	58
12.3 Finite Entropy	60
12.4 Mutual Information	61
12.5 Entropy	63
12.6 Conditional Mutual Information	64
12.7 Conditional Entropy	65
12.8 Equalities	66
13 Properties of Various Distributions	68
13.1 Erlang	69
13.2 Exponential distribution	71
13.3 Uniform distribution	73
13.4 Normal distribution	75
14 Characteristic Functions	80
14.1 Application of the FTC: integrating e^{ix}	80
14.2 The Characteristic Function of a Real Measure.	80
14.3 Independence	81
14.4 Approximations to e^{ix}	81
14.5 Calculation of the Characteristic Function of the Standard Distribution	83
15 Helly’s selection theorem	84
16 Integral of sinc	85
16.1 Various preparatory integrals	85
17 The sinc function, and the sine integral (Si)	86
17.1 The final theorems: boundedness and scalability	87
18 The Levy inversion theorem, and the Levy continuity theo- rem.	88
18.1 The Levy inversion theorem	88
18.2 The Levy continuity theorem	89
19 The Central Limit Theorem	89
20 Probability mass function	91
20.1 PMF as measure	91
20.2 Monad Interpretation	95
20.3 PMFs as function	100
20.4 Conditional Probabilities	103
20.5 Relator	104

20.6	Distributions	108
20.6.1	Bernoulli Distribution	108
20.6.2	Geometric Distribution	109
20.6.3	Uniform Multiset Distribution	110
20.6.4	Uniform Distribution	110
20.6.5	Poisson Distribution	112
20.6.6	Binomial Distribution	112
20.7	Negative Binomial distribution	114
20.8	PMFs from association lists	116
21	Code generation for PMFs	118
21.1	General code generation setup	118
21.2	Code abbreviations for integrals and probabilities	125
22	Finite Maps	127
22.1	Domain and Application	127
22.2	Constructor of Finite Maps	128
22.3	Product set of Finite Maps	128
22.3.1	Basic Properties of Pi'	129
22.4	Topological Space of Finite Maps	129
22.5	Metric Space of Finite Maps	130
22.6	Complete Space of Finite Maps	131
22.7	Second Countable Space of Finite Maps	131
22.8	Polish Space of Finite Maps	132
22.9	Product Measurable Space of Finite Maps	132
22.10	Isomorphism between Functions and Finite Maps	137
23	Projective Limit	139
23.1	Sequences of Finite Maps in Compact Sets	139
23.2	Daniell-Kolmogorov Theorem	140
24	Random Permutations	141
25	Discrete subprobability distribution	143
25.1	Auxiliary material	143
25.1.1	More about extended reals	143
25.1.2	More about <i>'a option</i>	144
25.1.3	A relator for sets that treats sets like predicates	146
25.1.4	Monotonicity rules	147
25.1.5	Bijections	147
25.2	Subprobability mass function	147
25.3	Support	150
25.4	Functorial structure	150
25.5	Monad operations	152

25.5.1	Return	152
25.5.2	Bind	152
25.6	Relator	154
25.7	From ' <i>a pmf</i> ' to ' <i>a spmf</i> '	156
25.8	Weight of a subprobability	157
25.9	From density to spmfs	159
25.10	Ordering on spmfs	159
25.11	CCPO structure for the flat ccpo <i>ord-option</i> (=)	161
25.11.1	Admissibility of <i>rel-spmf</i>	166
25.12	Restrictions on spmfs	167
25.13	Subprobability distributions of sets	169
25.14	Losslessness	171
25.15	Scaling	173
25.16	Conditional spmfs	175
25.17	Product spmf	176
25.18	Assertions	177
25.19	Try	178
25.20	Miscellaneous	180
26	Indexed products of PMFs	180
26.1	Preliminaries	180
26.2	Definition	180
26.3	Dependent product sets with a default	182
26.4	Common PMF operations on products	183
26.5	Merging and splitting PMF products	184
26.6	Additional properties	185
26.7	Applications	186
27	Hoeffding's Lemma and Hoeffding's Inequality	186
27.1	Hoeffding's Lemma	186
27.2	Hoeffding's Inequality	188
27.3	Hoeffding's inequality for i.i.d. bounded random variables	190
27.4	Hoeffding's Inequality for the Binomial distribution	191
27.5	Tail bounds for the negative binomial distribution	193
28	Conditional Expectation	202
28.1	Restricting a measure to a sub-sigma-algebra	203
28.2	Nonnegative conditional expectation	205
28.3	Real conditional expectation	207
29	The essential supremum	212
30	Stopping times	214
30.1	Stopping Time	214

31 Filtration	215
31.1 σ -algebra of a Stopping Time	215



1 Probability measure

theory *Probability-Measure*
imports *HOL-Analysis.Analysis*
begin

locale *prob-space* = *finite-measure* +
assumes *emeasure-space-1*: *emeasure* *M* (*space* *M*) = 1

lemma *prob-spaceI*[*Pure.intro!*]:
assumes *: *emeasure* *M* (*space* *M*) = 1
shows *prob-space* *M*
 ⟨*proof*⟩

lemma *prob-space-imp-sigma-finite*: *prob-space* *M* \implies *sigma-finite-measure* *M*
 ⟨*proof*⟩

abbreviation (**in** *prob-space*) *events* \equiv *sets* *M*
abbreviation (**in** *prob-space*) *prob* \equiv *measure* *M*
abbreviation (**in** *prob-space*) *random-variable* *M'* *X* \equiv *X* \in *measurable* *M* *M'*
abbreviation (**in** *prob-space*) *expectation* \equiv *integral*^{*L*} *M*
abbreviation (**in** *prob-space*) *variance* *X* \equiv *integral*^{*L*} *M* ($\lambda x. (X\ x - \text{expectation}\ X)^2$)

lemma (**in** *prob-space*) *finite-measure* [*simp*]: *finite-measure* *M*
 ⟨*proof*⟩

lemma (**in** *prob-space*) *prob-space-distr*:
assumes *f*: *f* \in *measurable* *M* *M'* **shows** *prob-space* (*distr* *M* *M'* *f*)
 ⟨*proof*⟩

lemma *prob-space-distrD*:
assumes *f*: *f* \in *measurable* *M* *N* **and** *M*: *prob-space* (*distr* *M* *N* *f*) **shows**
prob-space *M*
 ⟨*proof*⟩

lemma (**in** *prob-space*) *prob-space*: *prob* (*space* *M*) = 1
 ⟨*proof*⟩

lemma (**in** *prob-space*) *prob-le-1*[*simp*, *intro*]: *prob* *A* \leq 1
 ⟨*proof*⟩

lemma (**in** *prob-space*) *not-empty*: *space* *M* \neq {}
 ⟨*proof*⟩

lemma (**in** *prob-space*) *emeasure-eq-1-AE*:
S \in *sets* *M* \implies *AE* *x* *in* *M*. *x* \in *S* \implies *emeasure* *M* *S* = 1
 ⟨*proof*⟩

lemma (in *prob-space*) *emeasure-le-1*: $\text{emeasure } M \ S \leq 1$
 ⟨proof⟩

lemma (in *prob-space*) *emeasure-ge-1-iff*: $\text{emeasure } M \ A \geq 1 \longleftrightarrow \text{emeasure } M \ A = 1$
 ⟨proof⟩

lemma (in *prob-space*) *AE-iff-emeasure-eq-1*:
 assumes $[measurable]: \text{Measurable.pred } M \ P$
 shows $(\text{AE } x \text{ in } M. \ P \ x) \longleftrightarrow \text{emeasure } M \ \{x \in \text{space } M. \ P \ x\} = 1$
 ⟨proof⟩

lemma (in *prob-space*) *measure-le-1*: $\text{emeasure } M \ X \leq 1$
 ⟨proof⟩

lemma (in *prob-space*) *measure-ge-1-iff*: $\text{measure } M \ A \geq 1 \longleftrightarrow \text{measure } M \ A = 1$
 ⟨proof⟩

lemma (in *prob-space*) *AE-I-eq-1*:
 assumes $\text{emeasure } M \ \{x \in \text{space } M. \ P \ x\} = 1$ $\{x \in \text{space } M. \ P \ x\} \in \text{sets } M$
 shows $\text{AE } x \text{ in } M. \ P \ x$
 ⟨proof⟩

lemma *prob-space-restrict-space*:
 $S \in \text{sets } M \implies \text{emeasure } M \ S = 1 \implies \text{prob-space } (\text{restrict-space } M \ S)$
 ⟨proof⟩

lemma (in *prob-space*) *prob-compl*:
 assumes $A: A \in \text{events}$
 shows $\text{prob } (\text{space } M - A) = 1 - \text{prob } A$
 ⟨proof⟩

lemma (in *prob-space*) *AE-in-set-eq-1*:
 assumes $A[measurable]: A \in \text{events}$ shows $(\text{AE } x \text{ in } M. \ x \in A) \longleftrightarrow \text{prob } A = 1$
 ⟨proof⟩

lemma (in *prob-space*) *AE-False*: $(\text{AE } x \text{ in } M. \ \text{False}) \longleftrightarrow \text{False}$
 ⟨proof⟩

lemma (in *prob-space*) *AE-prob-1*:
 assumes $\text{prob } A = 1$ shows $\text{AE } x \text{ in } M. \ x \in A$
 ⟨proof⟩

lemma (in *prob-space*) *AE-const[simp]*: $(\text{AE } x \text{ in } M. \ P) \longleftrightarrow P$
 ⟨proof⟩

lemma (in *prob-space*) *ae-filter-bot*: $\text{ae-filter } M \neq \text{bot}$

$\langle proof \rangle$

lemma (in *prob-space*) *AE-contr*:
assumes *ae*: $AE\ \omega\ in\ M.\ P\ \omega\ AE\ \omega\ in\ M.\ \neg\ P\ \omega$
shows *False*
 $\langle proof \rangle$

lemma (in *prob-space*) *integral-ge-const*:
fixes *c* :: *real*
shows $integrable\ M\ f \implies (AE\ x\ in\ M.\ c \leq f\ x) \implies c \leq (\int x.\ f\ x\ \partial M)$
 $\langle proof \rangle$

lemma (in *prob-space*) *integral-le-const*:
fixes *c* :: *real*
shows $integrable\ M\ f \implies (AE\ x\ in\ M.\ f\ x \leq c) \implies (\int x.\ f\ x\ \partial M) \leq c$
 $\langle proof \rangle$

lemma (in *prob-space*) *nn-integral-ge-const*:
 $(AE\ x\ in\ M.\ c \leq f\ x) \implies c \leq (\int^+ x.\ f\ x\ \partial M)$
 $\langle proof \rangle$

lemma (in *prob-space*) *expectation-less*:
fixes *X* :: $- \Rightarrow real$
assumes [*simp*]: $integrable\ M\ X$
assumes *gt*: $AE\ x\ in\ M.\ X\ x < b$
shows $expectation\ X < b$
 $\langle proof \rangle$

lemma (in *prob-space*) *expectation-greater*:
fixes *X* :: $- \Rightarrow real$
assumes [*simp*]: $integrable\ M\ X$
assumes *gt*: $AE\ x\ in\ M.\ a < X\ x$
shows $a < expectation\ X$
 $\langle proof \rangle$

lemma (in *prob-space*) *jensens-inequality*:
fixes *q* :: $real \Rightarrow real$
assumes *X*: $integrable\ M\ X\ AE\ x\ in\ M.\ X\ x \in I$
assumes *I*: $I = \{a <..< b\} \vee I = \{a <..\} \vee I = \{..< b\} \vee I = UNIV$
assumes *q*: $integrable\ M\ (\lambda x.\ q\ (X\ x))\ convex-on\ I\ q$
shows $q\ (expectation\ X) \leq expectation\ (\lambda x.\ q\ (X\ x))$
 $\langle proof \rangle$

lemma (in *prob-space*) *finite-borel-measurable-integrable*:
assumes $f \in borel-measurable\ M$
and *finite* ($f'(space\ M)$)
shows $integrable\ M\ f$
 $\langle proof \rangle$

1.1 Introduce binder for probability

syntax

$-prob :: pttrn \Rightarrow logic \Rightarrow logic \Rightarrow logic \ (\langle 'P'((/- \text{ in } - / -)') \rangle)$

syntax-consts

$-prob == measure$

translations

$\mathcal{P}(x \text{ in } M. P) \Rightarrow CONST \text{ measure } M \ \{x \in CONST \text{ space } M. P\}$

$\langle ML \rangle$

definition

$cond-prob \ M \ P \ Q = \mathcal{P}(\omega \text{ in } M. P \ \omega \wedge Q \ \omega) / \mathcal{P}(\omega \text{ in } M. Q \ \omega)$

syntax

$-conditional-prob :: pttrn \Rightarrow logic \Rightarrow logic \Rightarrow logic \Rightarrow logic \ (\langle 'P'(- \text{ in } -. - \mid / -)') \rangle)$

syntax-consts

$-conditional-prob == cond-prob$

translations

$\mathcal{P}(x \text{ in } M. P \mid Q) \Rightarrow CONST \text{ cond-prob } M \ (\lambda x. P) \ (\lambda x. Q)$

lemma (in prob-space) AE-E-prob:

assumes $ae: AE \ x \text{ in } M. P \ x$

obtains S **where** $S \subseteq \{x \in \text{space } M. P \ x\} \ S \in \text{events} \ prob \ S = 1$

$\langle proof \rangle$

lemma (in prob-space) prob-neg: $\{x \in \text{space } M. P \ x\} \in \text{events} \implies \mathcal{P}(x \text{ in } M. \neg P \ x) = 1 - \mathcal{P}(x \text{ in } M. P \ x)$

$\langle proof \rangle$

lemma (in prob-space) prob-eq-AE:

$(AE \ x \text{ in } M. P \ x \longleftrightarrow Q \ x) \implies \{x \in \text{space } M. P \ x\} \in \text{events} \implies \{x \in \text{space } M. Q \ x\} \in \text{events} \implies \mathcal{P}(x \text{ in } M. P \ x) = \mathcal{P}(x \text{ in } M. Q \ x)$

$\langle proof \rangle$

lemma (in prob-space) prob-eq-0-AE:

assumes $not: AE \ x \text{ in } M. \neg P \ x$ **shows** $\mathcal{P}(x \text{ in } M. P \ x) = 0$

$\langle proof \rangle$

lemma (in prob-space) prob-Collect-eq-0:

$\{x \in \text{space } M. P \ x\} \in \text{sets } M \implies \mathcal{P}(x \text{ in } M. P \ x) = 0 \longleftrightarrow (AE \ x \text{ in } M. \neg P \ x)$

$\langle proof \rangle$

lemma (in prob-space) prob-Collect-eq-1:

$\{x \in \text{space } M. P \ x\} \in \text{sets } M \implies \mathcal{P}(x \text{ in } M. P \ x) = 1 \longleftrightarrow (AE \ x \text{ in } M. P \ x)$

$\langle proof \rangle$

lemma (in prob-space) prob-eq-0:

$A \in \text{sets } M \implies \text{prob } A = 0 \longleftrightarrow (AE \ x \text{ in } M. x \notin A)$

$\langle proof \rangle$

lemma (in *prob-space*) *prob-eq-1*:

$A \in \text{sets } M \implies \text{prob } A = 1 \iff (AE \ x \text{ in } M. \ x \in A)$
 $\langle \text{proof} \rangle$

lemma (in *prob-space*) *prob-sums*:

assumes $P: \bigwedge n. \{x \in \text{space } M. P \ n \ x\} \in \text{events}$
assumes $Q: \{x \in \text{space } M. Q \ x\} \in \text{events}$
assumes $ae: AE \ x \text{ in } M. (\forall n. P \ n \ x \longrightarrow Q \ x) \wedge (Q \ x \longrightarrow (\exists ! n. P \ n \ x))$
shows $(\lambda n. \mathcal{P}(x \text{ in } M. P \ n \ x)) \text{ sums } \mathcal{P}(x \text{ in } M. Q \ x)$
 $\langle \text{proof} \rangle$

lemma (in *prob-space*) *prob-sum*:

assumes $[simp, intro]: \text{finite } I$
assumes $P: \bigwedge n. n \in I \implies \{x \in \text{space } M. P \ n \ x\} \in \text{events}$
assumes $Q: \{x \in \text{space } M. Q \ x\} \in \text{events}$
assumes $ae: AE \ x \text{ in } M. (\forall n \in I. P \ n \ x \longrightarrow Q \ x) \wedge (Q \ x \longrightarrow (\exists ! n \in I. P \ n \ x))$
shows $\mathcal{P}(x \text{ in } M. Q \ x) = (\sum n \in I. \mathcal{P}(x \text{ in } M. P \ n \ x))$
 $\langle \text{proof} \rangle$

lemma (in *prob-space*) *prob-EX-countable*:

assumes $\text{sets}: \bigwedge i. i \in I \implies \{x \in \text{space } M. P \ i \ x\} \in \text{sets } M$ **and** $I: \text{countable } I$
assumes $\text{disj}: AE \ x \text{ in } M. \forall i \in I. \forall j \in I. P \ i \ x \longrightarrow P \ j \ x \longrightarrow i = j$
shows $\mathcal{P}(x \text{ in } M. \exists i \in I. P \ i \ x) = (\int^+ i. \mathcal{P}(x \text{ in } M. P \ i \ x) \ \partial \text{count-space } I)$
 $\langle \text{proof} \rangle$

lemma (in *prob-space*) *cond-prob-eq-AE*:

assumes $P: AE \ x \text{ in } M. Q \ x \longrightarrow P \ x \iff P' \ x \ \{x \in \text{space } M. P \ x\} \in \text{events}$
 $\{x \in \text{space } M. P' \ x\} \in \text{events}$
assumes $Q: AE \ x \text{ in } M. Q \ x \iff Q' \ x \ \{x \in \text{space } M. Q \ x\} \in \text{events} \ \{x \in \text{space } M. Q' \ x\} \in \text{events}$
shows $\text{cond-prob } M \ P \ Q = \text{cond-prob } M \ P' \ Q'$
 $\langle \text{proof} \rangle$

lemma (in *prob-space*) *joint-distribution-Times-le-fst*:

$\text{random-variable } MX \ X \implies \text{random-variable } MY \ Y \implies A \in \text{sets } MX \implies B \in \text{sets } MY$
 $\implies \text{emeasure } (\text{distr } M \ (MX \otimes_M MY) \ (\lambda x. (X \ x, Y \ x))) \ (A \times B) \leq \text{emeasure } (\text{distr } M \ MX \ X) \ A$
 $\langle \text{proof} \rangle$

lemma (in *prob-space*) *joint-distribution-Times-le-snd*:

$\text{random-variable } MX \ X \implies \text{random-variable } MY \ Y \implies A \in \text{sets } MX \implies B \in \text{sets } MY$
 $\implies \text{emeasure } (\text{distr } M \ (MX \otimes_M MY) \ (\lambda x. (X \ x, Y \ x))) \ (A \times B) \leq \text{emeasure } (\text{distr } M \ MY \ Y) \ B$
 $\langle \text{proof} \rangle$

lemma (in *prob-space*) *variance-eq*:

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

assumes [simp]: *integrable* $M\ X$

assumes [simp]: *integrable* $M\ (\lambda x. (X\ x)^2)$

shows *variance* $X = \text{expectation} (\lambda x. (X\ x)^2) - (\text{expectation } X)^2$

<proof>

lemma (in *prob-space*) *variance-positive*: $0 \leq \text{variance } (X :: 'a \Rightarrow \text{real})$

<proof>

lemma (in *prob-space*) *variance-mean-zero*:

expectation $X = 0 \implies \text{variance } X = \text{expectation } (\lambda x. (X\ x)^2)$

<proof>

theorem (in *prob-space*) *Chebyshev-inequality*:

assumes [measurable]: *random-variable borel* f

assumes *integrable* $M\ (\lambda x. f\ x^2)$

defines $\mu \equiv \text{expectation } f$

assumes $a > 0$

shows *prob* $\{x \in \text{space } M. |f\ x - \mu| \geq a\} \leq \text{variance } f / a^2$

unfolding $\mu\text{-def}$

proof (rule *second-moment-method*)

have *integrable*: *integrable* $M\ f$

using *assms by* (blast dest: *square-integrable-imp-integrable*)

show *integrable* $M\ (\lambda x. (f\ x - \text{expectation } f)^2)$

using *assms integrable unfolding power2-eq-square ring-distrib*

by (intro *Bochner-Integration.integrable-diff*) *auto*

qed (use *assms in auto*)

locale *pair-prob-space* = *pair-sigma-finite* $M1\ M2 + M1$: *prob-space* $M1 + M2$:
prob-space $M2$ **for** $M1\ M2$

sublocale *pair-prob-space* $\subseteq P?$: *prob-space* $M1 \otimes_M M2$

<proof>

locale *product-prob-space* = *product-sigma-finite* M **for** $M :: 'i \Rightarrow 'a\ \text{measure} +$

fixes $I :: 'i\ \text{set}$

assumes *prob-space*: $\bigwedge i. \text{prob-space } (M\ i)$

sublocale *product-prob-space* $\subseteq M?$: *prob-space* $M\ i$ **for** i

<proof>

locale *finite-product-prob-space* = *finite-product-sigma-finite* $M\ I + \text{product-prob-space}$
 $M\ I$ **for** $M\ I$

sublocale *finite-product-prob-space* $\subseteq \text{prob-space } \prod_{M\ i \in I. M\ i}$

<proof>

lemma (in *finite-product-prob-space*) *prob-times*:

assumes $X: \bigwedge i. i \in I \implies X\ i \in \text{sets } (M\ i)$
shows $\text{prob } (\prod_{E\ i \in I. X\ i}) = (\prod_{i \in I. M.\text{prob } i\ (X\ i)})$
 $\langle \text{proof} \rangle$

lemma *product-prob-spaceI*:
assumes $\bigwedge i. \text{prob-space } (M\ i)$
shows *product-prob-space* M
 $\langle \text{proof} \rangle$

1.2 Distributions

definition *distributed* $:: 'a\ \text{measure} \Rightarrow 'b\ \text{measure} \Rightarrow ('a \Rightarrow 'b) \Rightarrow ('b \Rightarrow \text{ennreal}) \Rightarrow \text{bool}$

where

$\text{distributed } M\ N\ X\ f \longleftrightarrow$
 $\text{distr } M\ N\ X = \text{density } N\ f \wedge f \in \text{borel-measurable } N \wedge X \in \text{measurable } M\ N$

lemma
assumes *distributed* $M\ N\ X\ f$
shows *distributed-distr-eq-density*: $\text{distr } M\ N\ X = \text{density } N\ f$
and *distributed-measurable*: $X \in \text{measurable } M\ N$
and *distributed-borel-measurable*: $f \in \text{borel-measurable } N$
 $\langle \text{proof} \rangle$

lemma
assumes $D: \text{distributed } M\ N\ X\ f$
shows *distributed-measurable*'[*measurable-dest*]:
 $g \in \text{measurable } L\ M \implies (\lambda x. X\ (g\ x)) \in \text{measurable } L\ N$
and *distributed-borel-measurable*'[*measurable-dest*]:
 $h \in \text{measurable } L\ N \implies (\lambda x. f\ (h\ x)) \in \text{borel-measurable } L$
 $\langle \text{proof} \rangle$

lemma *distributed-real-measurable*:
 $(\bigwedge x. x \in \text{space } N \implies 0 \leq f\ x) \implies \text{distributed } M\ N\ X\ (\lambda x. \text{ennreal } (f\ x)) \implies f \in \text{borel-measurable } N$
 $\langle \text{proof} \rangle$

lemma *distributed-real-measurable'*:
 $(\bigwedge x. x \in \text{space } N \implies 0 \leq f\ x) \implies \text{distributed } M\ N\ X\ (\lambda x. \text{ennreal } (f\ x)) \implies$
 $h \in \text{measurable } L\ N \implies (\lambda x. f\ (h\ x)) \in \text{borel-measurable } L$
 $\langle \text{proof} \rangle$

lemma *joint-distributed-measurable1*:
 $\text{distributed } M\ (S \otimes_M T)\ (\lambda x. (X\ x, Y\ x))\ f \implies h1 \in \text{measurable } N\ M \implies (\lambda x. X\ (h1\ x)) \in \text{measurable } N\ S$
 $\langle \text{proof} \rangle$

lemma *joint-distributed-measurable2*:
 $\text{distributed } M\ (S \otimes_M T)\ (\lambda x. (X\ x, Y\ x))\ f \implies h2 \in \text{measurable } N\ M \implies (\lambda x.$

$Y(h2\ x)) \in \text{measurable } N\ T$
 $\langle \text{proof} \rangle$

lemma *distributed-count-space:*

assumes X : *distributed* M (*count-space* A) $X\ P$ **and** a : $a \in A$ **and** A : *finite* A
shows $P\ a = \text{emeasure } M\ (X - \{a\} \cap \text{space } M)$
 $\langle \text{proof} \rangle$

lemma *distributed-cong-density:*

$(AE\ x\ \text{in } N. f\ x = g\ x) \implies g \in \text{borel-measurable } N \implies f \in \text{borel-measurable } N$
 \implies
 $\text{distributed } M\ N\ X\ f \longleftrightarrow \text{distributed } M\ N\ X\ g$
 $\langle \text{proof} \rangle$

lemma (*in prob-space*) *distributed-imp-emeasure-nonzero:*

assumes X : *distributed* $M\ MX\ X\ Px$
shows $\text{emeasure } MX\ \{x \in \text{space } MX. Px\ x \neq 0\} \neq 0$
 $\langle \text{proof} \rangle$

lemma *subdensity:*

assumes T : $T \in \text{measurable } P\ Q$
assumes f : *distributed* $M\ P\ X\ f$
assumes g : *distributed* $M\ Q\ Y\ g$
assumes Y : $Y = T \circ X$
shows $AE\ x\ \text{in } P. g\ (T\ x) = 0 \longrightarrow f\ x = 0$
 $\langle \text{proof} \rangle$

lemma *subdensity-real:*

fixes $g :: 'a \Rightarrow \text{real}$ **and** $f :: 'b \Rightarrow \text{real}$
assumes T : $T \in \text{measurable } P\ Q$
assumes f : *distributed* $M\ P\ X\ f$
assumes g : *distributed* $M\ Q\ Y\ g$
assumes Y : $Y = T \circ X$
shows $(AE\ x\ \text{in } P. 0 \leq g\ (T\ x)) \implies (AE\ x\ \text{in } P. 0 \leq f\ x) \implies AE\ x\ \text{in } P. g\ (T\ x) = 0 \longrightarrow f\ x = 0$
 $\langle \text{proof} \rangle$

lemma *distributed-emeasure:*

$\text{distributed } M\ N\ X\ f \implies A \in \text{sets } N \implies \text{emeasure } M\ (X - \{A \cap \text{space } M\} = (\int^{+x}. f\ x * \text{indicator } A\ x\ \partial N)$
 $\langle \text{proof} \rangle$

lemma *distributed-nn-integral:*

$\text{distributed } M\ N\ X\ f \implies g \in \text{borel-measurable } N \implies (\int^{+x}. f\ x * g\ x\ \partial N) = (\int^{+x}. g\ (X\ x)\ \partial M)$
 $\langle \text{proof} \rangle$

lemma *distributed-integral:*

$\text{distributed } M\ N\ X\ f \implies g \in \text{borel-measurable } N \implies (\bigwedge x. x \in \text{space } N \implies 0 \leq$

$f\ x) \implies$
 $(\int x. f\ x * g\ x\ \partial N) = (\int x. g\ (X\ x)\ \partial M)$
 $\langle proof \rangle$

lemma *distributed-transform-integral*:

assumes Px : *distributed* $M\ N\ X\ Px \wedge x. x \in \text{space } N \implies 0 \leq Px\ x$
assumes *distributed* $M\ P\ Y\ Py \wedge x. x \in \text{space } P \implies 0 \leq Py\ x$
assumes Y : $Y = T \circ X$ **and** T : $T \in \text{measurable } N\ P$ **and** f : $f \in \text{borel-measurable } P$
shows $(\int x. Py\ x * f\ x\ \partial P) = (\int x. Px\ x * f\ (T\ x)\ \partial N)$
 $\langle proof \rangle$

lemma (*in prob-space*) *distributed-unique*:

assumes Px : *distributed* $M\ S\ X\ Px$
assumes Py : *distributed* $M\ S\ X\ Py$
shows $AE\ x\ \text{in } S. Px\ x = Py\ x$
 $\langle proof \rangle$

lemma (*in prob-space*) *distributed-jointI*:

assumes *sigma-finite-measure* S *sigma-finite-measure* T
assumes $X[\text{measurable}]$: $X \in \text{measurable } M\ S$ **and** $Y[\text{measurable}]$: $Y \in \text{measurable } M\ T$
assumes $[\text{measurable}]$: $f \in \text{borel-measurable } (S \otimes_M T)$ **and** f : $AE\ x\ \text{in } S \otimes_M T. 0 \leq f\ x$
assumes eq : $\bigwedge A\ B. A \in \text{sets } S \implies B \in \text{sets } T \implies$
 $\text{emeasure } M\ \{x \in \text{space } M. X\ x \in A \wedge Y\ x \in B\} = (\int^+ x. (\int^+ y. f\ (x, y) * \text{indicator } B\ y\ \partial T) * \text{indicator } A\ x\ \partial S)$
shows *distributed* $M\ (S \otimes_M T)\ (\lambda x. (X\ x, Y\ x))\ f$
 $\langle proof \rangle$

lemma (*in prob-space*) *distributed-swap*:

assumes *sigma-finite-measure* S *sigma-finite-measure* T
assumes Pxy : *distributed* $M\ (S \otimes_M T)\ (\lambda x. (X\ x, Y\ x))\ Pxy$
shows *distributed* $M\ (T \otimes_M S)\ (\lambda x. (Y\ x, X\ x))\ (\lambda(x, y). Pxy\ (y, x))$
 $\langle proof \rangle$

lemma (*in prob-space*) *distr-marginal1*:

assumes *sigma-finite-measure* S *sigma-finite-measure* T
assumes Pxy : *distributed* $M\ (S \otimes_M T)\ (\lambda x. (X\ x, Y\ x))\ Pxy$
defines $Px \equiv \lambda x. (\int^+ z. Pxy\ (x, z)\ \partial T)$
shows *distributed* $M\ S\ X\ Px$
 $\langle proof \rangle$

lemma (*in prob-space*) *distr-marginal2*:

assumes S : *sigma-finite-measure* S **and** T : *sigma-finite-measure* T
assumes Pxy : *distributed* $M\ (S \otimes_M T)\ (\lambda x. (X\ x, Y\ x))\ Pxy$
shows *distributed* $M\ T\ Y\ (\lambda y. (\int^+ x. Pxy\ (x, y)\ \partial S))$
 $\langle proof \rangle$

lemma (in prob-space) distributed-marginal-eq-joint1:
 assumes T : sigma-finite-measure T
 assumes S : sigma-finite-measure S
 assumes Px : distributed $M S X Px$
 assumes Pxy : distributed $M (S \otimes_M T) (\lambda x. (X x, Y x)) Pxy$
 shows $AE\ x\ in\ S. Px\ x = (\int^+ y. Pxy\ (x, y)\ \partial T)$
 <proof>

lemma (in prob-space) distributed-marginal-eq-joint2:
 assumes T : sigma-finite-measure T
 assumes S : sigma-finite-measure S
 assumes Py : distributed $M T Y Py$
 assumes Pxy : distributed $M (S \otimes_M T) (\lambda x. (X x, Y x)) Pxy$
 shows $AE\ y\ in\ T. Py\ y = (\int^+ x. Pxy\ (x, y)\ \partial S)$
 <proof>

lemma (in prob-space) distributed-joint-indep':
 assumes S : sigma-finite-measure S and T : sigma-finite-measure T
 assumes X [measurable]: distributed $M S X Px$ and Y [measurable]: distributed $M T Y Py$
 assumes indep: $distr\ M\ S\ X \otimes_M distr\ M\ T\ Y = distr\ M\ (S \otimes_M T) (\lambda x. (X x, Y x))$
 shows $distributed\ M\ (S \otimes_M T) (\lambda x. (X x, Y x)) (\lambda(x, y). Px\ x * Py\ y)$
 <proof>

lemma distributed-integrable:
 distributed $M N X f \implies g \in \text{borel-measurable } N \implies (\bigwedge x. x \in \text{space } N \implies 0 \leq f\ x) \implies$
 $integrable\ N\ (\lambda x. f\ x * g\ x) \longleftrightarrow integrable\ M\ (\lambda x. g\ (X\ x))$
 <proof>

lemma distributed-transform-integrable:
 assumes Px : distributed $M N X Px \bigwedge x. x \in \text{space } N \implies 0 \leq Px\ x$
 assumes distributed $M P Y Py \bigwedge x. x \in \text{space } P \implies 0 \leq Py\ x$
 assumes $Y: Y = (\lambda x. T\ (X\ x))$ and $T: T \in \text{measurable } N\ P$ and $f: f \in \text{borel-measurable } P$
 shows $integrable\ P\ (\lambda x. Py\ x * f\ x) \longleftrightarrow integrable\ N\ (\lambda x. Px\ x * f\ (T\ x))$
 <proof>

lemma distributed-integrable-var:
 fixes $X :: 'a \Rightarrow \text{real}$
 shows distributed $M\ \text{lborel } X (\lambda x. \text{ennreal } (f\ x)) \implies (\bigwedge x. 0 \leq f\ x) \implies$
 $integrable\ \text{lborel } (\lambda x. f\ x * x) \implies integrable\ M\ X$
 <proof>

lemma (in prob-space) distributed-variance:
 fixes $f :: \text{real} \Rightarrow \text{real}$
 assumes D : distributed $M\ \text{lborel } X f$ and [simp]: $\bigwedge x. 0 \leq f\ x$
 shows $\text{variance } X = (\int x. x^2 * f\ (x + \text{expectation } X)\ \partial \text{lborel})$

$\langle proof \rangle$

lemma (in *prob-space*) *variance-affine*:

fixes $f :: real \Rightarrow real$

assumes [*arith*]: $b \neq 0$

assumes $D[intro]$: *distributed* M *lborel* X f

assumes [*simp*]: *prob-space* (*density lborel* f)

assumes $I[simp]$: *integrable* M X

assumes $I2[simp]$: *integrable* M $(\lambda x. (X\ x)^2)$

shows *variance* $(\lambda x. a + b * X\ x) = b^2 * \text{variance } X$

$\langle proof \rangle$

definition

simple-distributed M X $f \longleftrightarrow$

$(\forall x. 0 \leq f\ x) \wedge$

distributed M (*count-space* $(X'space\ M)$) X $(\lambda x. \text{ennreal } (f\ x)) \wedge$

finite $(X'space\ M)$

lemma *simple-distributed-nonneg[dest]*: *simple-distributed* M X $f \Longrightarrow 0 \leq f\ x$

$\langle proof \rangle$

lemma *simple-distributed*:

simple-distributed M X $Px \Longrightarrow \text{distributed } M$ (*count-space* $(X'space\ M)$) X Px

$\langle proof \rangle$

lemma *simple-distributed-finite[dest]*: *simple-distributed* M X $P \Longrightarrow \text{finite } (X'space\ M)$

$\langle proof \rangle$

lemma (in *prob-space*) *distributed-simple-function-superset*:

assumes X : *simple-function* M $X \bigwedge x. x \in X'space\ M \Longrightarrow P\ x = \text{measure } M$
 $(X - \{x\} \cap \text{space } M)$

assumes A : $X'space\ M \subseteq A$ *finite* A

defines $S \equiv \text{count-space } A$ **and** $P' \equiv (\lambda x. \text{if } x \in X'space\ M \text{ then } P\ x \text{ else } 0)$

shows *distributed* M S X P'

$\langle proof \rangle$

lemma (in *prob-space*) *simple-distributedI*:

assumes X : *simple-function* M X

$\bigwedge x. 0 \leq P\ x$

$\bigwedge x. x \in X'space\ M \Longrightarrow P\ x = \text{measure } M (X - \{x\} \cap \text{space } M)$

shows *simple-distributed* M X P

$\langle proof \rangle$

lemma *simple-distributed-joint-finite*:

assumes X : *simple-distributed* M $(\lambda x. (X\ x, Y\ x))\ Px$

shows *finite* $(X'space\ M)$ *finite* $(Y'space\ M)$

$\langle proof \rangle$

lemma *simple-distributed-joint2-finite:*

assumes X : *simple-distributed* M $(\lambda x. (X\ x, Y\ x, Z\ x))\ Px$

shows *finite* $(X\ 'space\ M)$ *finite* $(Y\ 'space\ M)$ *finite* $(Z\ 'space\ M)$

$\langle proof \rangle$

lemma *simple-distributed-simple-function:*

simple-distributed $M\ X\ Px \implies$ *simple-function* $M\ X$

$\langle proof \rangle$

lemma *simple-distributed-measure:*

simple-distributed $M\ X\ P \implies a \in X\ 'space\ M \implies P\ a = \text{measure } M\ (X - \{a\} \cap \text{space } M)$

$\langle proof \rangle$

lemma (*in prob-space*) *simple-distributed-joint:*

assumes X : *simple-distributed* M $(\lambda x. (X\ x, Y\ x))\ Px$

defines $S \equiv \text{count-space } (X\ 'space\ M) \otimes_M \text{count-space } (Y\ 'space\ M)$

defines $P \equiv (\lambda x. \text{if } x \in (\lambda x. (X\ x, Y\ x))\ 'space\ M \text{ then } Px\ x \text{ else } 0)$

shows *distributed* $M\ S\ (\lambda x. (X\ x, Y\ x))\ P$

$\langle proof \rangle$

lemma (*in prob-space*) *simple-distributed-joint2:*

assumes X : *simple-distributed* M $(\lambda x. (X\ x, Y\ x, Z\ x))\ Px$

defines $S \equiv \text{count-space } (X\ 'space\ M) \otimes_M \text{count-space } (Y\ 'space\ M) \otimes_M \text{count-space } (Z\ 'space\ M)$

defines $P \equiv (\lambda x. \text{if } x \in (\lambda x. (X\ x, Y\ x, Z\ x))\ 'space\ M \text{ then } Px\ x \text{ else } 0)$

shows *distributed* $M\ S\ (\lambda x. (X\ x, Y\ x, Z\ x))\ P$

$\langle proof \rangle$

lemma (*in prob-space*) *simple-distributed-sum-space:*

assumes X : *simple-distributed* $M\ X\ f$

shows *sum* $f\ (X\ 'space\ M) = 1$

$\langle proof \rangle$

lemma (*in prob-space*) *distributed-marginal-eq-joint-simple:*

assumes Px : *simple-function* $M\ X$

assumes P_y : *simple-distributed* $M\ Y\ P_y$

assumes P_{xy} : *simple-distributed* $M\ (\lambda x. (X\ x, Y\ x))\ P_{xy}$

assumes y : $y \in Y\ 'space\ M$

shows $P_y\ y = (\sum_{x \in X\ 'space\ M. \text{if } (x, y) \in (\lambda x. (X\ x, Y\ x))\ 'space\ M \text{ then } P_{xy}\ (x, y) \text{ else } 0)$

$\langle proof \rangle$

lemma *distributedI-real:*

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

assumes gen : *sets* $M1 = \text{sigma-sets } (\text{space } M1)\ E$ **and** *Int-stable* E

and A : *range* $A \subseteq E\ (\bigcup i::\text{nat. } A\ i) = \text{space } M1\ \bigwedge i. \text{emeasure } (\text{distr } M\ M1\ X)$

$(A\ i) \neq \infty$

and X : $X \in \text{measurable } M\ M1$

and f : $f \in \text{borel-measurable } M1 \text{ } AE \text{ } x \text{ in } M1. 0 \leq f \text{ } x$
and eq : $\bigwedge A. A \in E \implies \text{emeasure } M (X - 'A \cap \text{space } M) = (\int^+ x. f \text{ } x * \text{indicator } A \text{ } x \partial M1)$
shows $\text{distributed } M \text{ } M1 \text{ } X \text{ } f$
 $\langle \text{proof} \rangle$

lemma $\text{distributedI-borel-atMost}$:

fixes $f :: \text{real} \Rightarrow \text{real}$
assumes $[\text{measurable}]$: $X \in \text{borel-measurable } M$
and $[\text{measurable}]$: $f \in \text{borel-measurable borel}$ **and** $f[\text{simp}]$: $AE \text{ } x \text{ in } \text{lborel}. 0 \leq f \text{ } x$
and $g\text{-eq}$: $\bigwedge a. (\int^+ x. f \text{ } x * \text{indicator } \{..a\} \text{ } x \partial \text{lborel}) = \text{ennreal } (g \text{ } a)$
and $M\text{-eq}$: $\bigwedge a. \text{emeasure } M \{x \in \text{space } M. X \text{ } x \leq a\} = \text{ennreal } (g \text{ } a)$
shows $\text{distributed } M \text{ } \text{lborel } X \text{ } f$
 $\langle \text{proof} \rangle$

lemma $(\text{in prob-space}) \text{uniform-distributed-params}$:

assumes X : $\text{distributed } M \text{ } MX \text{ } X (\lambda x. \text{indicator } A \text{ } x / \text{measure } MX \text{ } A)$
shows $A \in \text{sets } MX \text{ measure } MX \text{ } A \neq 0$
 $\langle \text{proof} \rangle$

lemma $\text{prob-space-uniform-measure}$:

assumes A : $\text{emeasure } M \text{ } A \neq 0 \text{ emeasure } M \text{ } A \neq \infty$
shows $\text{prob-space } (\text{uniform-measure } M \text{ } A)$
 $\langle \text{proof} \rangle$

lemma $\text{prob-space-uniform-count-measure}$: $\text{finite } A \implies A \neq \{\} \implies \text{prob-space } (\text{uniform-count-measure } A)$

$\langle \text{proof} \rangle$

lemma $(\text{in prob-space}) \text{measure-uniform-measure-eq-cond-prob}$:

assumes $[\text{measurable}]$: $\text{Measurable.pred } M \text{ } P \text{ Measurable.pred } M \text{ } Q$
shows $\mathcal{P}(x \text{ in } \text{uniform-measure } M \{x \in \text{space } M. Q \text{ } x\}. P \text{ } x) = \mathcal{P}(x \text{ in } M. P \text{ } x \mid Q \text{ } x)$
 $\langle \text{proof} \rangle$

lemma $\text{prob-space-point-measure}$:

$\text{finite } S \implies (\bigwedge s. s \in S \implies 0 \leq p \text{ } s) \implies (\sum s \in S. p \text{ } s) = 1 \implies \text{prob-space } (\text{point-measure } S \text{ } p)$
 $\langle \text{proof} \rangle$

lemma $(\text{in prob-space}) \text{distr-pair-fst}$: $\text{distr } (N \otimes_M M) \text{ } N \text{ } \text{fst} = N$

$\langle \text{proof} \rangle$

lemma $(\text{in product-prob-space}) \text{distr-reorder}$:

assumes $\text{inj-on } t \text{ } J \text{ } t \in J \rightarrow K \text{ finite } K$
shows $\text{distr } (PiM \text{ } K \text{ } M) (PiM \text{ } J (\lambda x. M \text{ } (t \text{ } x))) (\lambda \omega. \lambda n \in J. \omega \text{ } (t \text{ } n)) = PiM \text{ } J (\lambda x. M \text{ } (t \text{ } x))$
 $\langle \text{proof} \rangle$

lemma (in *product-prob-space*) *distr-restrict*:

$J \subseteq K \implies \text{finite } K \implies (\Pi_M i \in J. M i) = \text{distr } (\Pi_M i \in K. M i) (\Pi_M i \in J. M i)$
 $(\lambda f. \text{restrict } f J)$
 $\langle \text{proof} \rangle$

lemma (in *product-prob-space*) *emeasure-prod-emb[simp]*:

assumes $L: J \subseteq L$ *finite* L **and** $X: X \in \text{sets } (Pi_M J M)$

shows $\text{emeasure } (Pi_M L M) (\text{prod-emb } L M J X) = \text{emeasure } (Pi_M J M) X$

$\langle \text{proof} \rangle$

lemma *emeasure-distr-restrict*:

assumes $I \subseteq K$ **and** $Q[\text{measurable-cong}]$: $\text{sets } Q = \text{sets } (PiM K M)$ **and**
 $A[\text{measurable}]$: $A \in \text{sets } (PiM I M)$

shows $\text{emeasure } (\text{distr } Q (PiM I M) (\lambda \omega. \text{restrict } \omega I)) A = \text{emeasure } Q$
 $(\text{prod-emb } K M I A)$

$\langle \text{proof} \rangle$

lemma (in *prob-space*) *prob-space-completion*: *prob-space (completion M)*

$\langle \text{proof} \rangle$

lemma *distr-PiM-finite-prob-space*:

assumes *fin*: *finite* I

assumes *product-prob-space* M

assumes *product-prob-space* M'

assumes $[\text{measurable}]$: $\bigwedge i. i \in I \implies f \in \text{measurable } (M i) (M' i)$

shows $\text{distr } (PiM I M) (PiM I M') (\text{compose } I f) = PiM I (\lambda i. \text{distr } (M i)$
 $(M' i) f)$

$\langle \text{proof} \rangle$

end

2 Distribution Functions

Shows that the cumulative distribution function (cdf) of a distribution (a measure on the reals) is nondecreasing and right continuous, which tends to 0 and 1 in either direction.

Conversely, every such function is the cdf of a unique distribution. This direction defines the measure in the obvious way on half-open intervals, and then applies the Caratheodory extension theorem.

theory *Distribution-Functions*

imports *Probability-Measure*

begin

lemma *UN-Ioc-eq-UNIV*: $(\bigcup n. \{ -\text{real } n <.. \text{real } n \}) = \text{UNIV}$

$\langle \text{proof} \rangle$

2.1 Properties of cdf's

definition

$cdf :: real \text{ measure} \Rightarrow real \Rightarrow real$

where

$cdf \ M \equiv \lambda x. \text{ measure } M \ \{..x\}$

lemma *cdf-def2*: $cdf \ M \ x = \text{ measure } M \ \{..x\}$

<proof>

locale *finite-borel-measure* = *finite-measure* *M* **for** *M* :: *real measure* +

assumes *M-is-borel*: *sets M* = *sets borel*

begin

lemma *sets-M[intro]*: $a \in \text{sets borel} \implies a \in \text{sets } M$

<proof>

lemma *cdf-diff-eq*:

assumes $x < y$

shows $cdf \ M \ y - cdf \ M \ x = \text{ measure } M \ \{x<..y\}$

<proof>

lemma *cdf-nondecreasing*: $x \leq y \implies cdf \ M \ x \leq cdf \ M \ y$

<proof>

lemma *borel-UNIV*: $\text{space } M = \text{UNIV}$

<proof>

lemma *cdf-nonneg*: $cdf \ M \ x \geq 0$

<proof>

lemma *cdf-bounded*: $cdf \ M \ x \leq \text{ measure } M \ (\text{space } M)$

<proof>

lemma *cdf-lim-inf*:

$((\lambda i. cdf \ M \ (\text{real } i)) \longrightarrow \text{ measure } M \ (\text{space } M))$

<proof>

lemma *cdf-lim-at-top*: $(cdf \ M \longrightarrow \text{ measure } M \ (\text{space } M)) \text{ at-top}$

<proof>

lemma *cdf-lim-neg-inf*: $((\lambda i. cdf \ M \ (- \text{ real } i)) \longrightarrow 0)$

<proof>

lemma *cdf-lim-at-bot*: $(cdf \ M \longrightarrow 0) \text{ at-bot}$

<proof>

lemma *cdf-is-right-cont*: $\text{continuous } (\text{at-right } a) \ (cdf \ M)$

<proof>

lemma *cdf-at-left*: $(cdf\ M \longrightarrow measure\ M\ \{..<a\})\ (at\text{-}left\ a)$
 $\langle proof \rangle$

lemma *isCont-cdf*: $isCont\ (cdf\ M)\ x \longleftrightarrow measure\ M\ \{x\} = 0$
 $\langle proof \rangle$

lemma *countable-atoms*: $countable\ \{x.\ measure\ M\ \{x\} > 0\}$
 $\langle proof \rangle$

end

locale *real-distribution* = *prob-space* M **for** $M :: real\ measure +$
assumes *events-eq-borel* [*simp*, *measurable-cong*]: $sets\ M = sets\ borel$
begin

lemma *finite-borel-measure-M*: *finite-borel-measure* M
 $\langle proof \rangle$

sublocale *finite-borel-measure* M
 $\langle proof \rangle$

lemma *space-eq-univ* [*simp*]: $space\ M = UNIV$
 $\langle proof \rangle$

lemma *cdf-bounded-prob*: $\bigwedge x.\ cdf\ M\ x \leq 1$
 $\langle proof \rangle$

lemma *cdf-lim-inf-prob*: $(\lambda i.\ cdf\ M\ (real\ i)) \longrightarrow 1$
 $\langle proof \rangle$

lemma *cdf-lim-at-top-prob*: $(cdf\ M \longrightarrow 1)\ at\text{-}top$
 $\langle proof \rangle$

lemma *measurable-finite-borel* [*simp*]:
 $f \in borel\text{-}measurable\ borel \implies f \in borel\text{-}measurable\ M$
 $\langle proof \rangle$

end

lemma (**in** *prob-space*) *real-distribution-distr* [*intro*, *simp*]:
random-variable $borel\ X \implies real\text{-}distribution\ (distr\ M\ borel\ X)$
 $\langle proof \rangle$

2.2 Uniqueness

lemma (**in** *finite-borel-measure*) *emeasure-Ioc*:
assumes $a \leq b$ **shows** $emeasure\ M\ \{a <.. b\} = cdf\ M\ b - cdf\ M\ a$
 $\langle proof \rangle$

lemma *cdf-unique'*:

fixes $M1\ M2$

assumes *finite-borel-measure* $M1$ **and** *finite-borel-measure* $M2$

assumes $\text{cdf } M1 = \text{cdf } M2$

shows $M1 = M2$

<proof>

lemma *cdf-unique*:

real-distribution $M1 \implies \text{real-distribution } M2 \implies \text{cdf } M1 = \text{cdf } M2 \implies M1 = M2$

<proof>

lemma

fixes $F :: \text{real} \Rightarrow \text{real}$

assumes *nondecF* : $\bigwedge x\ y. x \leq y \implies F\ x \leq F\ y$

and *right-cont-F* : $\bigwedge a. \text{continuous (at-right } a) F$

and *lim-F-at-bot* : $(F \longrightarrow 0) \text{ at-bot}$

and *lim-F-at-top* : $(F \longrightarrow m) \text{ at-top}$

and $m: 0 \leq m$

shows *interval-measure-UNIV*: *emeasure (interval-measure F) UNIV = m*

and *finite-borel-measure-interval-measure*: *finite-borel-measure (interval-measure F)*

<proof>

lemma *real-distribution-interval-measure*:

fixes $F :: \text{real} \Rightarrow \text{real}$

assumes *nondecF* : $\bigwedge x\ y. x \leq y \implies F\ x \leq F\ y$ **and**

right-cont-F : $\bigwedge a. \text{continuous (at-right } a) F$ **and**

lim-F-at-bot : $(F \longrightarrow 0) \text{ at-bot}$ **and**

lim-F-at-top : $(F \longrightarrow 1) \text{ at-top}$

shows *real-distribution (interval-measure F)*

<proof>

lemma

fixes $F :: \text{real} \Rightarrow \text{real}$

assumes *nondecF* : $\bigwedge x\ y. x \leq y \implies F\ x \leq F\ y$ **and**

right-cont-F : $\bigwedge a. \text{continuous (at-right } a) F$ **and**

lim-F-at-bot : $(F \longrightarrow 0) \text{ at-bot}$

shows *emeasure-interval-measure-Iic*: *emeasure (interval-measure F) \{.. x\} = F x*

and *measure-interval-measure-Iic*: *measure (interval-measure F) \{.. x\} = F x*

<proof>

lemma *cdf-interval-measure*:

$(\bigwedge x\ y. x \leq y \implies F\ x \leq F\ y) \implies (\bigwedge a. \text{continuous (at-right } a) F) \implies (F \longrightarrow 0) \text{ at-bot} \implies \text{cdf (interval-measure F)} = F$

<proof>

end

3 Weak Convergence of Functions and Distributions

Properties of weak convergence of functions and measures, including the portmanteau theorem.

```
theory Weak-Convergence
  imports Distribution-Functions
begin
```

4 Weak Convergence of Functions

definition

$$weak_conv :: (nat \Rightarrow (real \Rightarrow real)) \Rightarrow (real \Rightarrow real) \Rightarrow bool$$

where

$$weak_conv\ F\text{-seq}\ F \equiv \forall x. isCont\ F\ x \longrightarrow (\lambda n. F\text{-seq}\ n\ x) \longrightarrow F\ x$$

5 Weak Convergence of Distributions

definition

$$weak_conv_m :: (nat \Rightarrow real\ measure) \Rightarrow real\ measure \Rightarrow bool$$

where

$$weak_conv_m\ M\text{-seq}\ M \equiv weak_conv\ (\lambda n. cdf\ (M\text{-seq}\ n))\ (cdf\ M)$$

6 Skorohod’s theorem

locale *right-continuous-mono* =

fixes $f :: real \Rightarrow real$ **and** $a\ b :: real$

assumes *cont*: $\bigwedge x. continuous\ (at\text{-right}\ x)\ f$

assumes *mono*: *mono* f

assumes *bot*: $(f \longrightarrow a)\ at\text{-bot}$

assumes *top*: $(f \longrightarrow b)\ at\text{-top}$

begin

abbreviation $I :: real \Rightarrow real$ **where**

$$I\ \omega \equiv Inf\ \{x. \omega \leq f\ x\}$$

lemma *pseudoinverse*: **assumes** $a < \omega < b$ **shows** $\omega \leq f\ x \longleftrightarrow I\ \omega \leq x$
 $\langle proof \rangle$

lemma *pseudoinverse'*: $\forall \omega \in \{a < .. < b\}. \forall x. \omega \leq f\ x \longleftrightarrow I\ \omega \leq x$
 $\langle proof \rangle$

lemma *mono-I*: *mono-on* $\{a < .. < b\}\ I$
 $\langle proof \rangle$

end

locale *cdf-distribution* = *real-distribution*
begin

abbreviation $C \equiv \text{cdf } M$

sublocale *right-continuous-mono* $C \ 0 \ 1$
 $\langle \text{proof} \rangle$

lemma *measurable-C*[*measurable*]: $C \in \text{borel-measurable borel}$
 $\langle \text{proof} \rangle$

lemma *measurable-CI*[*measurable*]: $I \in \text{borel-measurable } (\text{restrict-space borel } \{0 < .. < 1 :: \text{real}\})$
 $\langle \text{proof} \rangle$

lemma *emeasure-distr-I*: $\text{emeasure } (\text{distr } (\text{restrict-space lborel } \{0 < .. < 1 :: \text{real}\}) \text{ borel } I) \text{ UNIV} = 1$
 $\langle \text{proof} \rangle$

lemma *distr-I-eq-M*: $\text{distr } (\text{restrict-space lborel } \{0 < .. < 1 :: \text{real}\}) \text{ borel } I = M$ (**is**
 $?I = -$)
 $\langle \text{proof} \rangle$

end

context
fixes $\mu :: \text{nat} \Rightarrow \text{real measure}$
and $M :: \text{real measure}$
assumes $\mu: \bigwedge n. \text{real-distribution } (\mu \ n)$
assumes $M: \text{real-distribution } M$
assumes $\mu\text{-to-}M: \text{weak-conv-m } \mu \ M$
begin

theorem *Skorohod*:

$\exists (\Omega :: \text{real measure}) (\text{Y-seq} :: \text{nat} \Rightarrow \text{real} \Rightarrow \text{real}) (Y :: \text{real} \Rightarrow \text{real}).$
 $\text{prob-space } \Omega \wedge$
 $(\forall n. \text{Y-seq } n \in \text{measurable } \Omega \text{ borel}) \wedge$
 $(\forall n. \text{distr } \Omega \text{ borel } (\text{Y-seq } n) = \mu \ n) \wedge$
 $Y \in \text{measurable } \Omega \text{ lborel} \wedge$
 $\text{distr } \Omega \text{ borel } Y = M \wedge$
 $(\forall x \in \text{space } \Omega. (\lambda n. \text{Y-seq } n \ x) \longrightarrow Y \ x)$
 $\langle \text{proof} \rangle$

The Portmanteau theorem, that is, the equivalence of various definitions of weak convergence.

theorem *weak-conv-imp-bdd-ae-continuous-conv*:

fixes
 $f :: \text{real} \Rightarrow 'a :: \{\text{banach, second-countable-topology}\}$

assumes

discont-null: $M (\{x. \neg \text{isCont } f\ x\}) = 0$ **and**

f-bdd: $\bigwedge x. \text{norm } (f\ x) \leq B$ **and**

[*measurable*]: $f \in \text{borel-measurable borel}$

shows

$(\lambda n. \text{integral}^L (\mu\ n)\ f) \longrightarrow \text{integral}^L M\ f$

<proof>

theorem *weak-conv-imp-integral-bdd-continuous-conv*:

fixes $f :: \text{real} \Rightarrow 'a :: \{\text{banach, second-countable-topology}\}$

assumes

$\bigwedge x. \text{isCont } f\ x$ **and**

$\bigwedge x. \text{norm } (f\ x) \leq B$

shows

$(\lambda n. \text{integral}^L (\mu\ n)\ f) \longrightarrow \text{integral}^L M\ f$

<proof>

theorem *weak-conv-imp-continuity-set-conv*:

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

assumes [*measurable*]: $A \in \text{sets borel}$ **and** $M (\text{frontier } A) = 0$

shows $(\lambda n. \text{measure } (\mu\ n)\ A) \longrightarrow \text{measure } M\ A$

<proof>

end

definition

cts-step :: $\text{real} \Rightarrow \text{real} \Rightarrow \text{real} \Rightarrow \text{real}$

where

cts-step $a\ b\ x \equiv \text{if } x \leq a \text{ then } 1 \text{ else if } x \geq b \text{ then } 0 \text{ else } (b - x) / (b - a)$

lemma *cts-step-uniformly-continuous*:

assumes [*arith*]: $a < b$

shows *uniformly-continuous-on UNIV* (*cts-step* $a\ b$)

<proof>

lemma (*in real-distribution*) *integrable-cts-step*: $a < b \implies \text{integrable } M\ (\text{cts-step } a\ b)$

<proof>

lemma (*in real-distribution*) *cdf-cts-step*:

assumes [*arith*]: $x < y$

shows $\text{cdf } M\ x \leq \text{integral}^L M\ (\text{cts-step } x\ y)$ **and** $\text{integral}^L M\ (\text{cts-step } x\ y) \leq \text{cdf } M\ y$

<proof>

context

fixes *M-seq* :: $\text{nat} \Rightarrow \text{real measure}$

and $M :: \text{real measure}$

assumes *distr-M-seq* [*simp*]: $\bigwedge n. \text{real-distribution } (M\text{-seq } n)$

assumes *distr-M* [*simp*]: *real-distribution M*
begin

theorem *continuity-set-conv-imp-weak-conv*:

fixes *f* :: *real* \Rightarrow *real*
assumes *: $\bigwedge A. A \in \text{sets borel} \implies M (\text{frontier } A) = 0 \implies (\lambda n. (\text{measure } (M\text{-seq } n) A)) \longrightarrow \text{measure } M A$
shows *weak-conv-m M-seq M*
 $\langle \text{proof} \rangle$

theorem *integral-cts-step-conv-imp-weak-conv*:

assumes *integral-conv*: $\bigwedge x y. x < y \implies (\lambda n. \text{integral}^L (M\text{-seq } n) (\text{cts-step } x y)) \longrightarrow \text{integral}^L M (\text{cts-step } x y)$
shows *weak-conv-m M-seq M*
 $\langle \text{proof} \rangle$

theorem *integral-bdd-continuous-conv-imp-weak-conv*:

assumes
 $\bigwedge f. (\bigwedge x. \text{isCont } f x) \implies (\bigwedge x. \text{abs } (f x) \leq 1) \implies (\lambda n. \text{integral}^L (M\text{-seq } n) f) \longrightarrow \text{integral}^L M f$
shows
weak-conv-m M-seq M
 $\langle \text{proof} \rangle$

end

end

7 The Giry monad

theory *Giry-Monad*

imports *Probability-Measure HOL-Library.Monad-Syntax*
begin

7.1 Sub-probability spaces

locale *subprob-space* = *finite-measure* +
assumes *emeasure-space-le-1*: *emeasure M (space M) ≤ 1*
assumes *subprob-not-empty*: *space M $\neq \{\}$*

lemma *subprob-spaceI*[*Pure.intro!*]:

assumes *: *emeasure M (space M) ≤ 1*
assumes *space M $\neq \{\}$*
shows *subprob-space M*
 $\langle \text{proof} \rangle$

lemma (in *subprob-space*) *emeasure-subprob-space-less-top*: *emeasure M A $\neq \text{top}$*
 $\langle \text{proof} \rangle$

lemma *prob-space-imp-subprob-space*:

prob-space $M \implies$ *subprob-space* M

\langle *proof* \rangle

lemma *subprob-space-imp-sigma-finite*: *subprob-space* $M \implies$ *sigma-finite-measure* M

\langle *proof* \rangle

sublocale *prob-space* \subseteq *subprob-space*

\langle *proof* \rangle

lemma *subprob-space-sigma* [*simp*]: $\Omega \neq \{\}$ \implies *subprob-space* (*sigma* Ω X)

\langle *proof* \rangle

lemma *subprob-space-null-measure*: *space* $M \neq \{\}$ \implies *subprob-space* (*null-measure* M)

\langle *proof* \rangle

lemma (*in subprob-space*) *subprob-space-distr*:

assumes $f: f \in$ *measurable* M M' **and** *space* $M' \neq \{\}$ **shows** *subprob-space* (*distr* M M' f)

\langle *proof* \rangle

lemma (*in subprob-space*) *subprob-emeasure-le-1*: *emeasure* M $X \leq 1$

\langle *proof* \rangle

lemma (*in subprob-space*) *subprob-measure-le-1*: *measure* M $X \leq 1$

\langle *proof* \rangle

lemma (*in subprob-space*) *nn-integral-le-const*:

assumes $0 \leq c$ *AE* x *in* M . f $x \leq c$

shows $(\int^+ x. f$ x $\partial M) \leq c$

\langle *proof* \rangle

lemma *emeasure-density-distr-interval*:

fixes $h ::$ *real* \Rightarrow *real* **and** $g ::$ *real* \Rightarrow *real* **and** $g' ::$ *real* \Rightarrow *real*

assumes [*simp*]: $a \leq b$

assumes $Mf[measurable]$: $f \in$ *borel-measurable borel*

assumes $Mg[measurable]$: $g \in$ *borel-measurable borel*

assumes $Mg'[measurable]$: $g' \in$ *borel-measurable borel*

assumes $Mh[measurable]$: $h \in$ *borel-measurable borel*

assumes *prob*: *subprob-space* (*density lborel* f)

assumes *nonnegf*: $\bigwedge x. f$ $x \geq 0$

assumes *derivg*: $\bigwedge x. x \in \{a..b\} \implies (g$ *has-real-derivative* g' $x)$ (*at* x)

assumes *contg'*: *continuous-on* $\{a..b\}$ g'

assumes *mono*: *strict-mono-on* $\{a..b\}$ g **and** *inv*: $\bigwedge x. h$ $x \in \{a..b\} \implies g$ (h x)

$= x$

assumes *range*: $\{a..b\} \subseteq$ *range* h

shows *emeasure* (*distr* (*density lborel* f) *lborel* h) $\{a..b\} =$

$\text{emeasure } (\text{density } \text{lborel } (\lambda x. f (g x) * g' x)) \{a..b\}$
 $\langle \text{proof} \rangle$

locale *pair-subprob-space* =
pair-sigma-finite *M1 M2* + *M1*: *subprob-space* *M1* + *M2*: *subprob-space* *M2* **for**
M1 M2

sublocale *pair-subprob-space* \subseteq *P?*: *subprob-space* *M1* \otimes_M *M2*
 $\langle \text{proof} \rangle$

lemma *subprob-space-null-measure-iff*:
subprob-space (*null-measure* *M*) \longleftrightarrow *space* *M* $\neq \{\}$
 $\langle \text{proof} \rangle$

lemma *subprob-space-restrict-space*:
assumes *M*: *subprob-space* *M*
and *A*: *A* \cap *space* *M* \in *sets* *M* *A* \cap *space* *M* $\neq \{\}$
shows *subprob-space* (*restrict-space* *M* *A*)
 $\langle \text{proof} \rangle$

definition *subprob-algebra* :: 'a *measure* \Rightarrow 'a *measure measure* **where**
subprob-algebra *K* =
 $(\text{SUP } A \in \text{sets } K. \text{vimage-algebra } \{M. \text{subprob-space } M \wedge \text{sets } M = \text{sets } K\})$
 $(\lambda M. \text{emeasure } M A) \text{ borel}$

lemma *space-subprob-algebra*: *space* (*subprob-algebra* *A*) = $\{M. \text{subprob-space } M \wedge \text{sets } M = \text{sets } A\}$
 $\langle \text{proof} \rangle$

lemma *subprob-algebra-cong*: *sets* *M* = *sets* *N* \implies *subprob-algebra* *M* = *subprob-algebra* *N*
 $\langle \text{proof} \rangle$

lemma *measurable-emeasure-subprob-algebra*[*measurable*]:
 $a \in \text{sets } A \implies (\lambda M. \text{emeasure } M a) \in \text{borel-measurable } (\text{subprob-algebra } A)$
 $\langle \text{proof} \rangle$

lemma *measurable-measure-subprob-algebra*[*measurable*]:
 $a \in \text{sets } A \implies (\lambda M. \text{measure } M a) \in \text{borel-measurable } (\text{subprob-algebra } A)$
 $\langle \text{proof} \rangle$

lemma *subprob-measurableD*:
assumes *N*: *N* \in *measurable* *M* (*subprob-algebra* *S*) **and** *x*: *x* \in *space* *M*
shows *space* (*N* *x*) = *space* *S*
and *sets* (*N* *x*) = *sets* *S*
and *measurable* (*N* *x*) *K* = *measurable* *S* *K*
and *measurable* *K* (*N* *x*) = *measurable* *K* *S*
 $\langle \text{proof} \rangle$

$\langle ML \rangle$

context

fixes $K\ M\ N$ **assumes** $K: K \in \text{measurable } M \text{ (subprob-algebra } N)$
begin

lemma *subprob-space-kernel*: $a \in \text{space } M \implies \text{subprob-space } (K\ a)$
 $\langle \text{proof} \rangle$

lemma *sets-kernel*: $a \in \text{space } M \implies \text{sets } (K\ a) = \text{sets } N$
 $\langle \text{proof} \rangle$

lemma *measurable-emeasure-kernel*[*measurable*]:
 $A \in \text{sets } N \implies (\lambda a. \text{emeasure } (K\ a)\ A) \in \text{borel-measurable } M$
 $\langle \text{proof} \rangle$

end

lemma *measurable-subprob-algebra*:
 $(\bigwedge a. a \in \text{space } M \implies \text{subprob-space } (K\ a)) \implies$
 $(\bigwedge a. a \in \text{space } M \implies \text{sets } (K\ a) = \text{sets } N) \implies$
 $(\bigwedge A. A \in \text{sets } N \implies (\lambda a. \text{emeasure } (K\ a)\ A) \in \text{borel-measurable } M) \implies$
 $K \in \text{measurable } M \text{ (subprob-algebra } N)$
 $\langle \text{proof} \rangle$

lemma *measurable-submarkov*:
 $K \in \text{measurable } M \text{ (subprob-algebra } M) \longleftrightarrow$
 $(\forall x \in \text{space } M. \text{subprob-space } (K\ x) \wedge \text{sets } (K\ x) = \text{sets } M) \wedge$
 $(\forall A \in \text{sets } M. (\lambda x. \text{emeasure } (K\ x)\ A) \in \text{measurable } M \text{ borel})$
 $\langle \text{proof} \rangle$

lemma *measurable-subprob-algebra-generated*:
assumes *eq*: $\text{sets } N = \text{sigma-sets } \Omega\ G$ **and** *Int-stable* $G\ G \subseteq \text{Pow } \Omega$
assumes *subsp*: $\bigwedge a. a \in \text{space } M \implies \text{subprob-space } (K\ a)$
assumes *sets*: $\bigwedge a. a \in \text{space } M \implies \text{sets } (K\ a) = \text{sets } N$
assumes $\bigwedge A. A \in G \implies (\lambda a. \text{emeasure } (K\ a)\ A) \in \text{borel-measurable } M$
assumes $\Omega: (\lambda a. \text{emeasure } (K\ a)\ \Omega) \in \text{borel-measurable } M$
shows $K \in \text{measurable } M \text{ (subprob-algebra } N)$
 $\langle \text{proof} \rangle$

lemma *space-subprob-algebra-empty-iff*:
 $\text{space } (\text{subprob-algebra } N) = \{\} \longleftrightarrow \text{space } N = \{\}$
 $\langle \text{proof} \rangle$

lemma *nn-integral-measurable-subprob-algebra*[*measurable*]:
assumes *f*: $f \in \text{borel-measurable } N$
shows $(\lambda M. \text{integral}^N\ M\ f) \in \text{borel-measurable } (\text{subprob-algebra } N)$ (**is** - $\in ?B$)
 $\langle \text{proof} \rangle$

lemma *measurable-distr*:

assumes $[measurable]: f \in measurable\ M\ N$
shows $(\lambda M'.\ distr\ M'\ N\ f) \in measurable\ (subprob-algebra\ M)\ (subprob-algebra\ N)$
 $\langle proof \rangle$

lemma *emeasure-space-subprob-algebra* $[measurable]$:

$(\lambda a.\ emeasure\ a\ (space\ a)) \in borel-measurable\ (subprob-algebra\ N)$
 $\langle proof \rangle$

lemma *integrable-measurable-subprob-algebra* $[measurable]$:

fixes $f :: 'a \Rightarrow 'b :: \{banach, second-countable-topology\}$
assumes $[measurable]: f \in borel-measurable\ N$
shows $Measurable.pred\ (subprob-algebra\ N)\ (\lambda M.\ integrable\ M\ f)$
 $\langle proof \rangle$

lemma *integral-measurable-subprob-algebra* $[measurable]$:

fixes $f :: 'a \Rightarrow 'b :: \{banach, second-countable-topology\}$
assumes $f [measurable]: f \in borel-measurable\ N$
shows $(\lambda M.\ integral^L\ M\ f) \in subprob-algebra\ N \rightarrow_M borel$
 $\langle proof \rangle$

lemma *measurable-pair-measure*:

assumes $f: f \in measurable\ M\ (subprob-algebra\ N)$
assumes $g: g \in measurable\ M\ (subprob-algebra\ L)$
shows $(\lambda x.\ f\ x \otimes_M g\ x) \in measurable\ M\ (subprob-algebra\ (N \otimes_M L))$
 $\langle proof \rangle$

lemma *restrict-space-measurable*:

assumes $X: X \neq \{\} \ X \in sets\ K$
assumes $N: N \in measurable\ M\ (subprob-algebra\ K)$
shows $(\lambda x.\ restrict-space\ (N\ x)\ X) \in measurable\ M\ (subprob-algebra\ (restrict-space\ K\ X))$
 $\langle proof \rangle$

7.2 Properties of “return”

definition *return* $:: 'a\ measure \Rightarrow 'a \Rightarrow 'a\ measure$ **where**

$return\ R\ x = measure-of\ (space\ R)\ (sets\ R)\ (\lambda A.\ indicator\ A\ x)$

lemma *space-return* $[simp]$: $space\ (return\ M\ x) = space\ M$

$\langle proof \rangle$

lemma *sets-return* $[simp]$: $sets\ (return\ M\ x) = sets\ M$

$\langle proof \rangle$

lemma *measurable-return1* $[simp]$: $measurable\ (return\ N\ x)\ L = measurable\ N\ L$

$\langle proof \rangle$

lemma *measurable-return2*[simp]: *measurable* L (*return* N x) = *measurable* L N
 ⟨*proof*⟩

lemma *return-sets-cong*: *sets* M = *sets* N \implies *return* M = *return* N
 ⟨*proof*⟩

lemma *return-cong*: *sets* A = *sets* B \implies *return* A x = *return* B x
 ⟨*proof*⟩

lemma *emeasure-return*[simp]:
 assumes $A \in \text{sets } M$
 shows *emeasure* (*return* M x) A = *indicator* A x
 ⟨*proof*⟩

lemma *prob-space-return*: $x \in \text{space } M \implies \text{prob-space } (\text{return } M \ x)$
 ⟨*proof*⟩

lemma *subprob-space-return*: $x \in \text{space } M \implies \text{subprob-space } (\text{return } M \ x)$
 ⟨*proof*⟩

lemma *subprob-space-return-ne*:
 assumes $\text{space } M \neq \{\}$ shows *subprob-space* (*return* M x)
 ⟨*proof*⟩

lemma *measure-return*: assumes X : $X \in \text{sets } M$ shows *measure* (*return* M x) X
 = *indicator* X x
 ⟨*proof*⟩

lemma *AE-return*:
 assumes [simp]: $x \in \text{space } M$ and [measurable]: *Measurable.pred* M P
 shows (*AE* y in *return* M x . P y) $\longleftrightarrow P$ x
 ⟨*proof*⟩

lemma *nn-integral-return*:
 assumes $x \in \text{space } M$ $g \in \text{borel-measurable } M$
 shows $(\int^+ a. g \ a \ \partial \text{return } M \ x) = g \ x$
 ⟨*proof*⟩

lemma *integral-return*:
 fixes $g :: - \Rightarrow 'a :: \{\text{banach, second-countable-topology}\}$
 assumes $x \in \text{space } M$ $g \in \text{borel-measurable } M$
 shows $(\int a. g \ a \ \partial \text{return } M \ x) = g \ x$
 ⟨*proof*⟩

lemma *return-measurable*[measurable]: *return* $N \in \text{measurable } N$ (*subprob-algebra* N)
 ⟨*proof*⟩

lemma *distr-return*:

assumes $f \in \text{measurable } M \ N$ **and** $x \in \text{space } M$

shows $\text{distr } (\text{return } M \ x) \ N \ f = \text{return } N \ (f \ x)$

<proof>

lemma *return-restrict-space*:

$\Omega \in \text{sets } M \implies \text{return } (\text{restrict-space } M \ \Omega) \ x = \text{restrict-space } (\text{return } M \ x) \ \Omega$

<proof>

lemma *measurable-distr2*:

assumes $f[\text{measurable}]$: $\text{case-prod } f \in \text{measurable } (L \otimes_M M) \ N$

assumes $g[\text{measurable}]$: $g \in \text{measurable } L \ (\text{subprob-algebra } M)$

shows $(\lambda x. \text{distr } (g \ x) \ N \ (f \ x)) \in \text{measurable } L \ (\text{subprob-algebra } N)$

<proof>

lemma *nn-integral-measurable-subprob-algebra2*:

assumes $f[\text{measurable}]$: $(\lambda(x, y). f \ x \ y) \in \text{borel-measurable } (M \otimes_M N)$

assumes $N[\text{measurable}]$: $L \in \text{measurable } M \ (\text{subprob-algebra } N)$

shows $(\lambda x. \text{integral}^N (L \ x) (f \ x)) \in \text{borel-measurable } M$

<proof>

lemma *emeasure-measurable-subprob-algebra2*:

assumes $A[\text{measurable}]$: $(\text{SIGMA } x:\text{space } M. A \ x) \in \text{sets } (M \otimes_M N)$

assumes $L[\text{measurable}]$: $L \in \text{measurable } M \ (\text{subprob-algebra } N)$

shows $(\lambda x. \text{emeasure } (L \ x) (A \ x)) \in \text{borel-measurable } M$

<proof>

lemma *measure-measurable-subprob-algebra2*:

assumes $A[\text{measurable}]$: $(\text{SIGMA } x:\text{space } M. A \ x) \in \text{sets } (M \otimes_M N)$

assumes $L[\text{measurable}]$: $L \in \text{measurable } M \ (\text{subprob-algebra } N)$

shows $(\lambda x. \text{measure } (L \ x) (A \ x)) \in \text{borel-measurable } M$

<proof>

definition *select-sets* $M = (\text{SOME } N. \text{sets } M = \text{sets } (\text{subprob-algebra } N))$

lemma *select-sets1*:

$\text{sets } M = \text{sets } (\text{subprob-algebra } N) \implies \text{sets } M = \text{sets } (\text{subprob-algebra } (\text{select-sets } M))$

<proof>

lemma *sets-select-sets[simp]*:

assumes *sets*: $\text{sets } M = \text{sets } (\text{subprob-algebra } N)$

shows $\text{sets } (\text{select-sets } M) = \text{sets } N$

<proof>

lemma *space-select-sets[simp]*:

$\text{sets } M = \text{sets } (\text{subprob-algebra } N) \implies \text{space } (\text{select-sets } M) = \text{space } N$

<proof>

7.3 Join

definition $join :: 'a \text{ measure measure} \Rightarrow 'a \text{ measure}$ **where**

$join\ M = \text{measure-of}\ (\text{space}\ (\text{select-sets}\ M))\ (\text{sets}\ (\text{select-sets}\ M))\ (\lambda B. \int^+ M'. \text{emeasure}\ M'\ B\ \partial M)$

lemma

shows $\text{space-join}[simp]: \text{space}\ (join\ M) = \text{space}\ (\text{select-sets}\ M)$

and $\text{sets-join}[simp]: \text{sets}\ (join\ M) = \text{sets}\ (\text{select-sets}\ M)$

$\langle \text{proof} \rangle$

lemma $\text{emeasure-join}:$

assumes $M[\text{simp}, \text{measurable-cong}]: \text{sets}\ M = \text{sets}\ (\text{subprob-algebra}\ N)$ **and** $A:$
 $A \in \text{sets}\ N$

shows $\text{emeasure}\ (join\ M)\ A = (\int^+ M'. \text{emeasure}\ M'\ A\ \partial M)$

$\langle \text{proof} \rangle$

lemma $\text{measurable-join}:$

$join \in \text{measurable}\ (\text{subprob-algebra}\ (\text{subprob-algebra}\ N))\ (\text{subprob-algebra}\ N)$

$\langle \text{proof} \rangle$

lemma $\text{nn-integral-join}:$

assumes $f: f \in \text{borel-measurable}\ N$

and $M[\text{measurable-cong}]: \text{sets}\ M = \text{sets}\ (\text{subprob-algebra}\ N)$

shows $(\int^+ x. f\ x\ \partial join\ M) = (\int^+ M'. \int^+ x. f\ x\ \partial M'\ \partial M)$

$\langle \text{proof} \rangle$

lemma $\text{measurable-join1}:$

$\llbracket f \in \text{measurable}\ N\ K; \text{sets}\ M = \text{sets}\ (\text{subprob-algebra}\ N) \rrbracket$

$\implies f \in \text{measurable}\ (join\ M)\ K$

$\langle \text{proof} \rangle$

lemma

fixes $f :: - \Rightarrow \text{real}$

assumes $f\text{-measurable}\ [\text{measurable}]: f \in \text{borel-measurable}\ N$

and $f\text{-bounded}: \bigwedge x. x \in \text{space}\ N \implies |f\ x| \leq B$

and $M\ [\text{measurable-cong}]: \text{sets}\ M = \text{sets}\ (\text{subprob-algebra}\ N)$

and $\text{fin}: \text{finite-measure}\ M$

and $M\text{-bounded}: \text{AE}\ M'\ \text{in}\ M. \text{emeasure}\ M'\ (\text{space}\ M') \leq \text{ennreal}\ B'$

shows $\text{integrable-join}: \text{integrable}\ (join\ M)\ f\ (\text{is}\ ?\text{integrable})$

and $\text{integral-join}: \text{integral}^L\ (join\ M)\ f = \int M'. \text{integral}^L\ M'\ f\ \partial M\ (\text{is}\ ?\text{integral})$

$\langle \text{proof} \rangle$

lemma $\text{join-assoc}:$

assumes $M[\text{measurable-cong}]: \text{sets}\ M = \text{sets}\ (\text{subprob-algebra}\ (\text{subprob-algebra}\ N))$

shows $join\ (\text{distr}\ M\ (\text{subprob-algebra}\ N)\ join) = join\ (join\ M)$

$\langle \text{proof} \rangle$

lemma $\text{join-return}:$

assumes $sets\ M = sets\ N$ **and** $subprob\text{-}space\ M$
shows $join\ (return\ (subprob\text{-}algebra\ N)\ M) = M$
 $\langle proof \rangle$

lemma $join\text{-}return'$:
assumes $sets\ N = sets\ M$
shows $join\ (distr\ M\ (subprob\text{-}algebra\ N)\ (return\ N)) = M$
 $\langle proof \rangle$

lemma $join\text{-}distr\text{-}distr$:
fixes $f :: 'a \Rightarrow 'b$ **and** $M :: 'a\ measure\ measure$ **and** $N :: 'b\ measure$
assumes $sets\ M = sets\ (subprob\text{-}algebra\ R)$ **and** $f \in measurable\ R\ N$
shows $join\ (distr\ M\ (subprob\text{-}algebra\ N)\ (\lambda M. distr\ M\ N\ f)) = distr\ (join\ M)$
 $N\ f\ (is\ ?r = ?l)$
 $\langle proof \rangle$

definition $bind :: 'a\ measure \Rightarrow ('a \Rightarrow 'b\ measure) \Rightarrow 'b\ measure$ **where**
 $bind\ M\ f = (if\ space\ M = \{\} \text{ then } count\text{-}space\ \{\} \text{ else }$
 $join\ (distr\ M\ (subprob\text{-}algebra\ (f\ (SOME\ x. x \in space\ M))))\ f))$

adhoc-overloading $Monad\text{-}Syntax.bind \rightleftharpoons bind$

lemma $bind\text{-}empty$:
 $space\ M = \{\} \implies bind\ M\ f = count\text{-}space\ \{\}$
 $\langle proof \rangle$

lemma $bind\text{-}nonempty$:
 $space\ M \neq \{\} \implies bind\ M\ f = join\ (distr\ M\ (subprob\text{-}algebra\ (f\ (SOME\ x. x \in space\ M))))\ f)$
 $\langle proof \rangle$

lemma $sets\text{-}bind\text{-}empty$: $sets\ M = \{\} \implies sets\ (bind\ M\ f) = \{\{\}\}$
 $\langle proof \rangle$

lemma $space\text{-}bind\text{-}empty$: $space\ M = \{\} \implies space\ (bind\ M\ f) = \{\}$
 $\langle proof \rangle$

lemma $sets\text{-}bind[simp, measurable\text{-}cong]$:
assumes $f: \bigwedge x. x \in space\ M \implies sets\ (f\ x) = sets\ N$ **and** $M: space\ M \neq \{\}$
shows $sets\ (bind\ M\ f) = sets\ N$
 $\langle proof \rangle$

lemma $space\text{-}bind[simp]$:
assumes $\bigwedge x. x \in space\ M \implies sets\ (f\ x) = sets\ N$ **and** $space\ M \neq \{\}$
shows $space\ (bind\ M\ f) = space\ N$
 $\langle proof \rangle$

lemma $bind\text{-}cong\text{-}All$:
assumes $\forall x \in space\ M. f\ x = g\ x$

shows $\text{bind } M f = \text{bind } M g$
 $\langle \text{proof} \rangle$

lemma *bind-cong*:

$M = N \implies (\bigwedge x. x \in \text{space } M \implies f x = g x) \implies \text{bind } M f = \text{bind } N g$
 $\langle \text{proof} \rangle$

lemma *bind-nonempty'*:

assumes $f \in \text{measurable } M \text{ (subprob-algebra } N) \ x \in \text{space } M$
shows $\text{bind } M f = \text{join (distr } M \text{ (subprob-algebra } N) f)$
 $\langle \text{proof} \rangle$

lemma *bind-nonempty''*:

assumes $f \in \text{measurable } M \text{ (subprob-algebra } N) \ \text{space } M \neq \{\}$
shows $\text{bind } M f = \text{join (distr } M \text{ (subprob-algebra } N) f)$
 $\langle \text{proof} \rangle$

lemma *emeasure-bind*:

$\llbracket \text{space } M \neq \{\}; f \in \text{measurable } M \text{ (subprob-algebra } N); X \in \text{sets } N \rrbracket$
 $\implies \text{emeasure } (M \ggg f) X = \int^+ x. \text{emeasure } (f x) X \ \partial M$
 $\langle \text{proof} \rangle$

lemma *nn-integral-bind*:

assumes $f: f \in \text{borel-measurable } B$
assumes $N: N \in \text{measurable } M \text{ (subprob-algebra } B)$
shows $(\int^+ x. f x \ \partial(M \ggg N)) = (\int^+ x. \int^+ y. f y \ \partial N x \ \partial M)$
 $\langle \text{proof} \rangle$

lemma *AE-bind*:

assumes $N[\text{measurable}]: N \in \text{measurable } M \text{ (subprob-algebra } B)$
assumes $P[\text{measurable}]: \text{Measurable.pred } B P$
shows $(AE \ x \ \text{in } M \ggg N. P x) \longleftrightarrow (AE \ x \ \text{in } M. AE \ y \ \text{in } N x. P y)$
 $\langle \text{proof} \rangle$

lemma *measurable-bind'*:

assumes $M1: f \in \text{measurable } M \text{ (subprob-algebra } N) \ \text{and}$
 $M2: \text{case-prod } g \in \text{measurable } (M \otimes_M N) \text{ (subprob-algebra } R)$
shows $(\lambda x. \text{bind } (f x) (g x)) \in \text{measurable } M \text{ (subprob-algebra } R)$
 $\langle \text{proof} \rangle$

lemma *measurable-bind[measurable (raw)]*:

assumes $M1: f \in \text{measurable } M \text{ (subprob-algebra } N) \ \text{and}$
 $M2: (\lambda x. g \ (\text{fst } x) \ (\text{snd } x)) \in \text{measurable } (M \otimes_M N) \text{ (subprob-algebra } R)$
shows $(\lambda x. \text{bind } (f x) (g x)) \in \text{measurable } M \text{ (subprob-algebra } R)$
 $\langle \text{proof} \rangle$

lemma *measurable-bind2*:

assumes $f \in \text{measurable } M \text{ (subprob-algebra } N) \ \text{and } g \in \text{measurable } N \text{ (subprob-algebra } R)$

shows $(\lambda x. \text{bind } (f x) g) \in \text{measurable } M \text{ (subprob-algebra } R)$
 $\langle \text{proof} \rangle$

lemma *subprob-space-bind*:
assumes *subprob-space* M $f \in \text{measurable } M \text{ (subprob-algebra } N)$
shows *subprob-space* $(M \gg f)$
 $\langle \text{proof} \rangle$

lemma
fixes $f :: - \Rightarrow \text{real}$
assumes *f-measurable* [*measurable*]: $f \in \text{borel-measurable } K$
and *f-bounded*: $\bigwedge x. x \in \text{space } K \implies |f x| \leq B$
and N [*measurable*]: $N \in \text{measurable } M \text{ (subprob-algebra } K)$
and *fin*: *finite-measure* M
and *M-bounded*: $\text{AE } x \text{ in } M. \text{emeasure } (N x) (\text{space } (N x)) \leq \text{ennreal } B'$
shows *integrable-bind*: *integrable* $(\text{bind } M N) f$ (**is** *?integrable*)
and *integral-bind*: $\text{integral}^L (\text{bind } M N) f = \int x. \text{integral}^L (N x) f \partial M$ (**is** *?integral*)
 $\langle \text{proof} \rangle$

lemma (**in** *prob-space*) *prob-space-bind*:
assumes *ae*: $\text{AE } x \text{ in } M. \text{prob-space } (N x)$
and N [*measurable*]: $N \in \text{measurable } M \text{ (subprob-algebra } S)$
shows *prob-space* $(M \gg N)$
 $\langle \text{proof} \rangle$

lemma (**in** *subprob-space*) *bind-in-space*:
 $A \in \text{measurable } M \text{ (subprob-algebra } N) \implies (M \gg A) \in \text{space } (\text{subprob-algebra } N)$
 $\langle \text{proof} \rangle$

lemma (**in** *subprob-space*) *measure-bind*:
assumes $f: f \in \text{measurable } M \text{ (subprob-algebra } N)$ **and** $X: X \in \text{sets } N$
shows *measure* $(M \gg f) X = \int x. \text{measure } (f x) X \partial M$
 $\langle \text{proof} \rangle$

lemma *emeasure-bind-const*:
 $\text{space } M \neq \{\}$ $\implies X \in \text{sets } N \implies \text{subprob-space } N \implies$
 $\text{emeasure } (M \gg (\lambda x. N)) X = \text{emeasure } N X * \text{emeasure } M (\text{space } M)$
 $\langle \text{proof} \rangle$

lemma *emeasure-bind-const'*:
assumes *subprob-space* M *subprob-space* N
shows $\text{emeasure } (M \gg (\lambda x. N)) X = \text{emeasure } N X * \text{emeasure } M (\text{space } M)$
 $\langle \text{proof} \rangle$

lemma *emeasure-bind-const-prob-space*:
assumes *prob-space* M *subprob-space* N
shows $\text{emeasure } (M \gg (\lambda x. N)) X = \text{emeasure } N X$

$\langle \text{proof} \rangle$

lemma *bind-return*:

assumes $f \in \text{measurable } M \text{ (subprob-algebra } N) \text{ and } x \in \text{space } M$

shows $\text{bind } (\text{return } M \ x) \ f = f \ x$

$\langle \text{proof} \rangle$

lemma *bind-return'*:

shows $\text{bind } M \ (\text{return } M) = M$

$\langle \text{proof} \rangle$

lemma *distr-bind*:

assumes $N: N \in \text{measurable } M \text{ (subprob-algebra } K) \text{ space } M \neq \{\}$

assumes $f: f \in \text{measurable } K \ R$

shows $\text{distr } (M \gg N) \ R \ f = (M \gg (\lambda x. \text{distr } (N \ x) \ R \ f))$

$\langle \text{proof} \rangle$

lemma *bind-distr*:

assumes $f[\text{measurable}]: f \in \text{measurable } M \ X$

assumes $N[\text{measurable}]: N \in \text{measurable } X \text{ (subprob-algebra } K) \text{ and space } M \neq \{\}$

shows $(\text{distr } M \ X \ f \gg N) = (M \gg (\lambda x. N \ (f \ x)))$

$\langle \text{proof} \rangle$

lemma *bind-count-space-singleton*:

assumes $\text{subprob-space } (f \ x)$

shows $\text{count-space } \{x\} \gg f = f \ x$

$\langle \text{proof} \rangle$

lemma *restrict-space-bind*:

assumes $N: N \in \text{measurable } M \text{ (subprob-algebra } K)$

assumes $\text{space } M \neq \{\}$

assumes $X[\text{simp}]: X \in \text{sets } K \ X \neq \{\}$

shows $\text{restrict-space } (\text{bind } M \ N) \ X = \text{bind } M \ (\lambda x. \text{restrict-space } (N \ x) \ X)$

$\langle \text{proof} \rangle$

lemma *bind-restrict-space*:

assumes $A: A \cap \text{space } M \neq \{\} \ A \cap \text{space } M \in \text{sets } M$

and $f: f \in \text{measurable } (\text{restrict-space } M \ A) \text{ (subprob-algebra } N)$

shows $\text{restrict-space } M \ A \gg f = M \gg (\lambda x. \text{if } x \in A \text{ then } f \ x \text{ else null-measure } (f \ (\text{SOME } x. x \in A \wedge x \in \text{space } M)))$

(**is** ?lhs = ?rhs **is** - = $M \gg ?f$)

$\langle \text{proof} \rangle$

lemma *bind-const'*: $\llbracket \text{prob-space } M; \text{subprob-space } N \rrbracket \implies M \gg (\lambda x. N) = N$

$\langle \text{proof} \rangle$

lemma *bind-return-distr*:

assumes $\text{space } M \neq \{\} \ f \in \text{measurable } M \ N$

shows $\text{bind } M \text{ (return } N \circ f) = \text{distr } M \ N \ f$
 $\langle \text{proof} \rangle$

lemma *bind-return-distr'*:

space $M \neq \{\}$ $\implies f \in \text{measurable } M \ N \implies \text{bind } M \ (\lambda x. \text{return } N \ (f \ x)) = \text{distr } M \ N \ f$
 $\langle \text{proof} \rangle$

lemma *bind-assoc*:

fixes $f :: 'a \Rightarrow 'b \text{ measure}$ **and** $g :: 'b \Rightarrow 'c \text{ measure}$
assumes $M1: f \in \text{measurable } M \ (\text{subprob-algebra } N)$ **and** $M2: g \in \text{measurable } N \ (\text{subprob-algebra } R)$
shows $\text{bind } (\text{bind } M \ f) \ g = \text{bind } M \ (\lambda x. \text{bind } (f \ x) \ g)$
 $\langle \text{proof} \rangle$

lemma *double-bind-assoc*:

assumes $Mg: g \in \text{measurable } N \ (\text{subprob-algebra } N')$
assumes $Mf: f \in \text{measurable } M \ (\text{subprob-algebra } M')$
assumes $Mh: \text{case-prod } h \in \text{measurable } (M \otimes_M M') \ N$
shows $\text{do } \{x \leftarrow M; y \leftarrow f \ x; g \ (h \ x \ y)\} = \text{do } \{x \leftarrow M; y \leftarrow f \ x; \text{return } N \ (h \ x \ y)\} \ggg g$
 $\langle \text{proof} \rangle$

lemma (**in** *prob-space*) *M-in-subprob[measurable (raw)]*: $M \in \text{space } (\text{subprob-algebra } M)$
 $\langle \text{proof} \rangle$

lemma (**in** *pair-prob-space*) *pair-measure-eq-bind*:

$(M1 \otimes_M M2) = (M1 \ggg (\lambda x. M2 \ggg (\lambda y. \text{return } (M1 \otimes_M M2) \ (x, y))))$
 $\langle \text{proof} \rangle$

lemma (**in** *pair-prob-space*) *bind-rotate*:

assumes $C[\text{measurable}]: (\lambda(x, y). C \ x \ y) \in \text{measurable } (M1 \otimes_M M2) \ (\text{subprob-algebra } N)$
shows $(M1 \ggg (\lambda x. M2 \ggg (\lambda y. C \ x \ y))) = (M2 \ggg (\lambda y. M1 \ggg (\lambda x. C \ x \ y)))$
 $\langle \text{proof} \rangle$

lemma *bind-return''*: $\text{sets } M = \text{sets } N \implies M \ggg \text{return } N = M$
 $\langle \text{proof} \rangle$

lemma (**in** *prob-space*) *distr-const[simp]*:

$c \in \text{space } N \implies \text{distr } M \ N \ (\lambda x. c) = \text{return } N \ c$
 $\langle \text{proof} \rangle$

lemma *return-count-space-eq-density*:

$\text{return } (\text{count-space } M) \ x = \text{density } (\text{count-space } M) \ (\text{indicator } \{x\})$
 $\langle \text{proof} \rangle$

lemma *null-measure-in-space-subprob-algebra* [simp]:
 $\text{null-measure } M \in \text{space } (\text{subprob-algebra } M) \longleftrightarrow \text{space } M \neq \{\}$
 ⟨proof⟩

7.4 Giry monad on probability spaces

definition *prob-algebra* :: 'a measure \Rightarrow 'a measure measure **where**
 $\text{prob-algebra } K = \text{restrict-space } (\text{subprob-algebra } K) \{M. \text{prob-space } M\}$

lemma *space-prob-algebra*: $\text{space } (\text{prob-algebra } M) = \{N. \text{sets } N = \text{sets } M \wedge \text{prob-space } N\}$
 ⟨proof⟩

lemma *measurable-measure-prob-algebra*[measurable]:
 $a \in \text{sets } A \implies (\lambda M. \text{Sigma-Algebra.measure } M a) \in \text{prob-algebra } A \rightarrow_M \text{borel}$
 ⟨proof⟩

lemma *measurable-prob-algebraD*:
 $f \in N \rightarrow_M \text{prob-algebra } M \implies f \in N \rightarrow_M \text{subprob-algebra } M$
 ⟨proof⟩

lemma *measure-measurable-prob-algebra2*:
 $\text{Sigma } (\text{space } M) A \in \text{sets } (M \otimes_M N) \implies L \in M \rightarrow_M \text{prob-algebra } N \implies$
 $(\lambda x. \text{Sigma-Algebra.measure } (L x) (A x)) \in \text{borel-measurable } M$
 ⟨proof⟩

lemma *measurable-prob-algebraI*:
 $(\bigwedge x. x \in \text{space } N \implies \text{prob-space } (f x)) \implies f \in N \rightarrow_M \text{subprob-algebra } M \implies$
 $f \in N \rightarrow_M \text{prob-algebra } M$
 ⟨proof⟩

lemma *measurable-distr-prob-space*:
assumes $f: f \in M \rightarrow_M N$
shows $(\lambda M'. \text{distr } M' N f) \in \text{prob-algebra } M \rightarrow_M \text{prob-algebra } N$
 ⟨proof⟩

lemma *measurable-return-prob-space*[measurable]: $\text{return } N \in N \rightarrow_M \text{prob-algebra } N$
 ⟨proof⟩

lemma *measurable-distr-prob-space2*[measurable (raw)]:
assumes $f: g \in L \rightarrow_M \text{prob-algebra } M \ (\lambda(x, y). f x y) \in L \otimes_M M \rightarrow_M N$
shows $(\lambda x. \text{distr } (g x) N (f x)) \in L \rightarrow_M \text{prob-algebra } N$
 ⟨proof⟩

lemma *measurable-bind-prob-space*:
assumes $f: f \in M \rightarrow_M \text{prob-algebra } N$ **and** $g: g \in N \rightarrow_M \text{prob-algebra } R$
shows $(\lambda x. \text{bind } (f x) g) \in M \rightarrow_M \text{prob-algebra } R$
 ⟨proof⟩

lemma *measurable-bind-prob-space2*[*measurable (raw)*]:

assumes $f: f \in M \rightarrow_M \text{prob-algebra } N$ **and** $g: (\lambda(x, y). g \ x \ y) \in (M \otimes_M N) \rightarrow_M \text{prob-algebra } R$
shows $(\lambda x. \text{bind } (f \ x) \ (g \ x)) \in M \rightarrow_M \text{prob-algebra } R$
<proof>

lemma *measurable-prob-algebra-generated*:

assumes *eq*: $\text{sets } N = \text{sigma-sets } \Omega \ G$ **and** *Int-stable* $G \ G \subseteq \text{Pow } \Omega$
assumes *subsp*: $\bigwedge a. a \in \text{space } M \implies \text{prob-space } (K \ a)$
assumes *sets*: $\bigwedge a. a \in \text{space } M \implies \text{sets } (K \ a) = \text{sets } N$
assumes $\bigwedge A. A \in G \implies (\lambda a. \text{emeasure } (K \ a) \ A) \in \text{borel-measurable } M$
shows $K \in \text{measurable } M \ (\text{prob-algebra } N)$
<proof>

lemma *in-space-prob-algebra*:

$x \in \text{space } (\text{prob-algebra } M) \implies \text{emeasure } x \ (\text{space } M) = 1$
<proof>

lemma *prob-space-pair*:

assumes $\text{prob-space } M \ \text{prob-space } N$ **shows** $\text{prob-space } (M \otimes_M N)$
<proof>

lemma *measurable-pair-prob*[*measurable*]:

$f \in M \rightarrow_M \text{prob-algebra } N \implies g \in M \rightarrow_M \text{prob-algebra } L \implies (\lambda x. f \ x \otimes_M g \ x) \in M \rightarrow_M \text{prob-algebra } (N \otimes_M L)$
<proof>

lemma *emeasure-bind-prob-algebra*:

assumes $A: A \in \text{space } (\text{prob-algebra } N)$
assumes $B: B \in N \rightarrow_M \text{prob-algebra } L$
assumes $X: X \in \text{sets } L$
shows $\text{emeasure } (\text{bind } A \ B) \ X = (\int^+ x. \text{emeasure } (B \ x) \ X \ \partial A)$
<proof>

lemma *prob-space-bind'*:

assumes $A: A \in \text{space } (\text{prob-algebra } M)$ **and** $B: B \in M \rightarrow_M \text{prob-algebra } N$
shows $\text{prob-space } (A \gg B)$
<proof>

lemma *sets-bind'*:

assumes $A: A \in \text{space } (\text{prob-algebra } M)$ **and** $B: B \in M \rightarrow_M \text{prob-algebra } N$
shows $\text{sets } (A \gg B) = \text{sets } N$
<proof>

lemma *bind-cong-AE'*:

assumes $M: M \in \text{space } (\text{prob-algebra } L)$
and $f: f \in L \rightarrow_M \text{prob-algebra } N$ **and** $g: g \in L \rightarrow_M \text{prob-algebra } N$

and $ae: AE\ x\ in\ M. f\ x = g\ x$
shows $bind\ M\ f = bind\ M\ g$
 $\langle proof \rangle$

lemma *density-discrete:*

$countable\ A \implies sets\ N = Set.Pow\ A \implies (\bigwedge x. f\ x \geq 0) \implies (\bigwedge x. x \in A \implies f\ x = emeasure\ N\ \{x\}) \implies$
 $density\ (count-space\ A)\ f = N$
 $\langle proof \rangle$

lemma *distr-density-discrete:*

fixes f'
assumes $countable\ A$
assumes $f' \in borel-measurable\ M$
assumes $g \in measurable\ M\ (count-space\ A)$
defines $f \equiv \lambda x. \int^+ t. (if\ g\ t = x\ then\ 1\ else\ 0) * f'\ t\ \partial M$
assumes $\bigwedge x. x \in space\ M \implies g\ x \in A$
shows $density\ (count-space\ A)\ (\lambda x. f\ x) = distr\ (density\ M\ f')\ (count-space\ A)\ g$
 $\langle proof \rangle$

lemma *bind-cong-AE:*

assumes $M = N$
assumes $f: f \in measurable\ N\ (subprob-algebra\ B)$
assumes $g: g \in measurable\ N\ (subprob-algebra\ B)$
assumes $ae: AE\ x\ in\ N. f\ x = g\ x$
shows $bind\ M\ f = bind\ N\ g$
 $\langle proof \rangle$

lemma *bind-cong-simp:* $M = N \implies (\bigwedge x. x \in space\ M = simp \implies f\ x = g\ x) \implies$
 $bind\ M\ f = bind\ N\ g$
 $\langle proof \rangle$

lemma *sets-bind-measurable:*

assumes $f: f \in measurable\ M\ (subprob-algebra\ B)$
assumes $M: space\ M \neq \{\}$
shows $sets\ (M \gg f) = sets\ B$
 $\langle proof \rangle$

lemma *space-bind-measurable:*

assumes $f: f \in measurable\ M\ (subprob-algebra\ B)$
assumes $M: space\ M \neq \{\}$
shows $space\ (M \gg f) = space\ B$
 $\langle proof \rangle$

lemma *bind-distr-return:*

$f \in M \rightarrow_M N \implies g \in N \rightarrow_M L \implies space\ M \neq \{\} \implies$
 $distr\ M\ N\ f \gg (\lambda x. return\ L\ (g\ x)) = distr\ M\ L\ (\lambda x. g\ (f\ x))$
 $\langle proof \rangle$

lemma (in *prob-space*) *AE-eq-constD*:
 assumes *AE x in M. x = y*
 shows *M = return M y y ∈ space M*
 ⟨*proof*⟩
 end

8 Projective Family

theory *Projective-Family*
imports *Giry-Monad*
begin

lemma *vimage-restrict-preserve-mono*:
 assumes *J: J ⊆ I*
 and *sets: A ⊆ (Π_E i∈J. S i) B ⊆ (Π_E i∈J. S i) and ne: (Π_E i∈I. S i) ≠ {}*
 and *eq: (λx. restrict x J) -‘ A ∩ (Π_E i∈I. S i) ⊆ (λx. restrict x J) -‘ B ∩ (Π_E i∈I. S i)*
 shows *A ⊆ B*
 ⟨*proof*⟩

locale *projective-family* =
 fixes *I :: 'i set* and *P :: 'i set ⇒ ('i ⇒ 'a) measure* and *M :: 'i ⇒ 'a measure*
 assumes *P: ⋀ J H. J ⊆ H ⇒ finite H ⇒ H ⊆ I ⇒ P J = distr (P H) (PiM J M) (λf. restrict f J)*
 assumes *prob-space-P: ⋀ J. finite J ⇒ J ⊆ I ⇒ prob-space (P J)*
begin

lemma *sets-P: finite J ⇒ J ⊆ I ⇒ sets (P J) = sets (PiM J M)*
 ⟨*proof*⟩

lemma *space-P: finite J ⇒ J ⊆ I ⇒ space (P J) = space (PiM J M)*
 ⟨*proof*⟩

lemma *not-empty-M: i ∈ I ⇒ space (M i) ≠ {}*
 ⟨*proof*⟩

lemma *not-empty: space (PiM I M) ≠ {}*
 ⟨*proof*⟩

abbreviation

emb L K ≡ prod-emb L M K

lemma *emb-preserve-mono*:
 assumes *J ⊆ L L ⊆ I* and *sets: X ∈ sets (PiM J M) Y ∈ sets (PiM J M)*
 assumes *emb L J X ⊆ emb L J Y*
 shows *X ⊆ Y*

$\langle proof \rangle$

lemma *emb-injective*:

assumes $L: J \subseteq L \subseteq I$ **and** $X: X \in sets (Pi_M J M)$ **and** $Y: Y \in sets (Pi_M J M)$

shows $emb L J X = emb L J Y \implies X = Y$

$\langle proof \rangle$

lemma *emeasure-P*: $J \subseteq K \implies finite K \implies K \subseteq I \implies X \in sets (Pi_M J M) \implies P K (emb K J X) = P J X$

$\langle proof \rangle$

inductive-set *generator* :: $('i \Rightarrow 'a)$ *set set* **where**

$finite J \implies J \subseteq I \implies X \in sets (Pi_M J M) \implies emb I J X \in generator$

lemma *algebra-generator*: *algebra* (*space* ($Pi_M I M$)) *generator*

$\langle proof \rangle$

interpretation *generator*: *algebra* (*space* ($Pi_M I M$)) *generator*

$\langle proof \rangle$

lemma *sets-PiM-generator*: $sets (Pi_M I M) = sigma-sets (space (Pi_M I M)) generator$

$\langle proof \rangle$

definition *mu-G* (μG) **where**

$\mu G A = (THE x. \forall J \subseteq I. finite J \longrightarrow (\forall X \in sets (Pi_M J M). A = emb I J X \longrightarrow x = emeasure (P J) X))$

definition *lim* :: $('i \Rightarrow 'a)$ *measure* **where**

$lim = extend-measure (space (Pi_M I M)) generator (\lambda x. x) \mu G$

lemma *space-lim[simp]*: $space lim = space (Pi_M I M)$

$\langle proof \rangle$

lemma *sets-lim[simp, measurable]*: $sets lim = sets (Pi_M I M)$

$\langle proof \rangle$

lemma *mu-G-spec*:

assumes $J: finite J \subseteq I$ $X \in sets (Pi_M J M)$

shows $\mu G (emb I J X) = emeasure (P J) X$

$\langle proof \rangle$

lemma *positive-mu-G*: *positive generator* μG

$\langle proof \rangle$

lemma *additive-mu-G*: *additive generator* μG

$\langle proof \rangle$

lemma *emeasure-lim*:

assumes JX : *finite* J $J \subseteq I$ $X \in \text{sets } (PiM\ J\ M)$
assumes *cont*: $\bigwedge J\ X. (\bigwedge i. J\ i \subseteq I) \implies \text{incseq } J \implies (\bigwedge i. \text{finite } (J\ i)) \implies (\bigwedge i. X\ i \in \text{sets } (PiM\ (J\ i)\ M)) \implies$
 $\text{decseq } (\lambda i. \text{emb } I\ (J\ i)\ (X\ i)) \implies 0 < (\text{INF } i. P\ (J\ i)\ (X\ i)) \implies (\bigcap i. \text{emb } I\ (J\ i)\ (X\ i)) \neq \{\}$
shows *emeasure lim* $(\text{emb } I\ J\ X) = P\ J\ X$
 $\langle \text{proof} \rangle$

end

sublocale *product-prob-space* \subseteq *projective-family* $I\ \lambda J. PiM\ J\ M\ M$
 $\langle \text{proof} \rangle$

Proof due to Ionescu Tulcea.

locale *Ionescu-Tulcea* =

fixes $P :: \text{nat} \Rightarrow (\text{nat} \Rightarrow 'a) \Rightarrow 'a\ \text{measure}$ **and** $M :: \text{nat} \Rightarrow 'a\ \text{measure}$
assumes $P[\text{measurable}]$: $\bigwedge i. P\ i \in \text{measurable } (PiM\ \{0..<i\}\ M)\ (\text{subprob-algebra } (M\ i))$
assumes *prob-space-P*: $\bigwedge i\ x. x \in \text{space } (PiM\ \{0..<i\}\ M) \implies \text{prob-space } (P\ i\ x)$
begin

lemma *non-empty[simp]*: $\text{space } (M\ i) \neq \{\}$
 $\langle \text{proof} \rangle$

lemma *space-PiM-not-empty[simp]*: $\text{space } (PiM\ \text{UNIV}\ M) \neq \{\}$
 $\langle \text{proof} \rangle$

lemma *space-P*: $x \in \text{space } (PiM\ \{0..<n\}\ M) \implies \text{space } (P\ n\ x) = \text{space } (M\ n)$
 $\langle \text{proof} \rangle$

lemma *sets-P[measurable-cong]*: $x \in \text{space } (PiM\ \{0..<n\}\ M) \implies \text{sets } (P\ n\ x) = \text{sets } (M\ n)$
 $\langle \text{proof} \rangle$

definition $eP :: \text{nat} \Rightarrow (\text{nat} \Rightarrow 'a) \Rightarrow (\text{nat} \Rightarrow 'a)\ \text{measure}$ **where**
 $eP\ n\ \omega = \text{distr } (P\ n\ \omega)\ (PiM\ \{0..<Suc\ n\}\ M)\ (\text{fun-upd } \omega\ n)$

lemma *measurable-eP[measurable]*:
 $eP\ n \in \text{measurable } (PiM\ \{0..<n\}\ M)\ (\text{subprob-algebra } (PiM\ \{0..<Suc\ n\}\ M))$
 $\langle \text{proof} \rangle$

lemma *space-eP*:
 $x \in \text{space } (PiM\ \{0..<n\}\ M) \implies \text{space } (eP\ n\ x) = \text{space } (PiM\ \{0..<Suc\ n\}\ M)$
 $\langle \text{proof} \rangle$

lemma *sets-eP[measurable]*:
 $x \in \text{space } (PiM\ \{0..<n\}\ M) \implies \text{sets } (eP\ n\ x) = \text{sets } (PiM\ \{0..<Suc\ n\}\ M)$

$\langle \text{proof} \rangle$

lemma *prob-space-eP*: $x \in \text{space } (PiM \{0..<n\} M) \implies \text{prob-space } (eP \ n \ x)$
 $\langle \text{proof} \rangle$

lemma *nn-integral-eP*:

$\omega \in \text{space } (PiM \{0..<n\} M) \implies f \in \text{borel-measurable } (PiM \{0..<Suc \ n\} M)$
 \implies
 $(\int^{+x}. f \ x \ \partial eP \ n \ \omega) = (\int^{+x}. f \ (\text{fun-upd } \omega \ n \ x) \ \partial P \ n \ \omega)$
 $\langle \text{proof} \rangle$

lemma *emeasure-eP*:

assumes $\omega[\text{simp}]$: $\omega \in \text{space } (PiM \{0..<n\} M)$ **and** $A[\text{measurable}]$: $A \in \text{sets } (PiM \{0..<Suc \ n\} M)$
shows $eP \ n \ \omega \ A = P \ n \ \omega \ ((\lambda x. \text{fun-upd } \omega \ n \ x) -' A \cap \text{space } (M \ n))$
 $\langle \text{proof} \rangle$

primrec $C :: \text{nat} \Rightarrow \text{nat} \Rightarrow (\text{nat} \Rightarrow 'a) \Rightarrow (\text{nat} \Rightarrow 'a) \text{ measure where}$

$C \ n \ 0 \ \omega = \text{return } (PiM \{0..<n\} M) \ \omega$
 $| \ C \ n \ (Suc \ m) \ \omega = C \ n \ m \ \omega \gg eP \ (n + m)$

lemma *measurable-C[measurable]*:

$C \ n \ m \in \text{measurable } (PiM \{0..<n\} M) \ (\text{subprob-algebra } (PiM \{0..<n + m\} M))$
 $\langle \text{proof} \rangle$

lemma *space-C*:

$x \in \text{space } (PiM \{0..<n\} M) \implies \text{space } (C \ n \ m \ x) = \text{space } (PiM \{0..<n + m\} M)$
 $\langle \text{proof} \rangle$

lemma *sets-C[measurable-cong]*:

$x \in \text{space } (PiM \{0..<n\} M) \implies \text{sets } (C \ n \ m \ x) = \text{sets } (PiM \{0..<n + m\} M)$
 $\langle \text{proof} \rangle$

lemma *prob-space-C*: $x \in \text{space } (PiM \{0..<n\} M) \implies \text{prob-space } (C \ n \ m \ x)$
 $\langle \text{proof} \rangle$

lemma *split-C*:

assumes ω : $\omega \in \text{space } (PiM \{0..<n\} M)$ **shows** $(C \ n \ m \ \omega \gg C \ (n + m) \ l) = C \ n \ (m + l) \ \omega$
 $\langle \text{proof} \rangle$

lemma *nn-integral-C*:

assumes $m \leq m'$ **and** $f[\text{measurable}]$: $f \in \text{borel-measurable } (PiM \{0..<n+m\} M)$
and nonneg : $\bigwedge x. x \in \text{space } (PiM \{0..<n+m\} M) \implies 0 \leq f \ x$
and x : $x \in \text{space } (PiM \{0..<n\} M)$
shows $(\int^{+x}. f \ x \ \partial C \ n \ m \ x) = (\int^{+x}. f \ (\text{restrict } x \ \{0..<n+m\}) \ \partial C \ n \ m' \ x)$

$\langle \text{proof} \rangle$

lemma *emeasure-C*:

assumes $m \leq m'$ **and** $A[\text{measurable}]$: $A \in \text{sets } (PiM \ \{0..<n+m\} \ M)$ **and** $[simp]$:
 $x \in \text{space } (PiM \ \{0..<n\} \ M)$

shows $\text{emeasure } (C \ n \ m' \ x) \ (\text{prod-emb } \{0..<n + m'\} \ M \ \{0..<n+m\} \ A) =$
 $\text{emeasure } (C \ n \ m \ x) \ A$

$\langle \text{proof} \rangle$

lemma *distr-C*:

assumes $m \leq m'$ **and** $[simp]$: $x \in \text{space } (PiM \ \{0..<n\} \ M)$

shows $C \ n \ m \ x = \text{distr } (C \ n \ m' \ x) \ (PiM \ \{0..<n+m\} \ M) \ (\lambda x. \text{restrict } x \ \{0..<n+m\})$

$\langle \text{proof} \rangle$

definition *up-to* :: $\text{nat set} \Rightarrow \text{nat}$ **where**

$\text{up-to } J = (\text{LEAST } n. \forall i \geq n. i \notin J)$

lemma *up-to-less*: $\text{finite } J \Longrightarrow i \in J \Longrightarrow i < \text{up-to } J$

$\langle \text{proof} \rangle$

lemma *up-to-iff*: $\text{finite } J \Longrightarrow \text{up-to } J \leq n \longleftrightarrow (\forall i \in J. i < n)$

$\langle \text{proof} \rangle$

lemma *up-to-iff-Ico*: $\text{finite } J \Longrightarrow \text{up-to } J \leq n \longleftrightarrow J \subseteq \{0..<n\}$

$\langle \text{proof} \rangle$

lemma *up-to*: $\text{finite } J \Longrightarrow J \subseteq \{0..< \text{up-to } J\}$

$\langle \text{proof} \rangle$

lemma *up-to-mono*: $J \subseteq H \Longrightarrow \text{finite } H \Longrightarrow \text{up-to } J \leq \text{up-to } H$

$\langle \text{proof} \rangle$

definition *CI* :: $\text{nat set} \Rightarrow (\text{nat} \Rightarrow 'a) \text{ measure}$ **where**

$\text{CI } J = \text{distr } (C \ 0 \ (\text{up-to } J) \ (\lambda x. \text{undefined})) \ (PiM \ J \ M) \ (\lambda f. \text{restrict } f \ J)$

sublocale *PF*: *projective-family UNIV CI*

$\langle \text{proof} \rangle$

lemma *emeasure-CI'*:

$\text{finite } J \Longrightarrow X \in \text{sets } (PiM \ J \ M) \Longrightarrow \text{CI } J \ X = C \ 0 \ (\text{up-to } J) \ (\lambda-. \text{undefined})$
 $(PF.\text{emb } \{0..<\text{up-to } J\} \ J \ X)$

$\langle \text{proof} \rangle$

lemma *emeasure-CI*:

$J \subseteq \{0..<n\} \Longrightarrow X \in \text{sets } (PiM \ J \ M) \Longrightarrow \text{CI } J \ X = C \ 0 \ n \ (\lambda-. \text{undefined})$
 $(PF.\text{emb } \{0..<n\} \ J \ X)$

$\langle \text{proof} \rangle$

lemma *lim*:

assumes J : *finite* J **and** X : $X \in \text{sets } (PiM\ J\ M)$

shows $\text{emeasure } PF.\text{lim } (PF.\text{emb } UNIV\ J\ X) = \text{emeasure } (CI\ J)\ X$

$\langle \text{proof} \rangle$

lemma *distr-lim*: **assumes** $J[\text{simp}]$: *finite* J **shows** $\text{distr } PF.\text{lim } (PiM\ J\ M)\ (\lambda x. \text{restrict } x\ J) = CI\ J$

$\langle \text{proof} \rangle$

end

lemma (*in product-prob-space*) *emeasure-lim-emb*:

assumes $*$: *finite* $J\ J \subseteq I\ X \in \text{sets } (PiM\ J\ M)$

shows $\text{emeasure } \text{lim } (\text{emb } I\ J\ X) = \text{emeasure } (Pi_M\ J\ M)\ X$

$\langle \text{proof} \rangle$

end

9 Infinite Product Measure

theory *Infinite-Product-Measure*

imports *Probability-Measure Projective-Family*

begin

lemma (*in product-prob-space*) *distr-PiM-restrict-finite*:

assumes *finite* $J\ J \subseteq I$

shows $\text{distr } (PiM\ I\ M)\ (PiM\ J\ M)\ (\lambda x. \text{restrict } x\ J) = PiM\ J\ M$

$\langle \text{proof} \rangle$

lemma (*in product-prob-space*) *emeasure-PiM-emb'*:

$J \subseteq I \implies \text{finite } J \implies X \in \text{sets } (PiM\ J\ M) \implies \text{emeasure } (Pi_M\ I\ M)\ (\text{emb } I\ J\ X) = PiM\ J\ M\ X$

$\langle \text{proof} \rangle$

lemma (*in product-prob-space*) *emeasure-PiM-emb*:

$J \subseteq I \implies \text{finite } J \implies (\bigwedge i. i \in J \implies X\ i \in \text{sets } (M\ i)) \implies$

$\text{emeasure } (Pi_M\ I\ M)\ (\text{emb } I\ J\ (Pi_E\ J\ X)) = (\prod_{i \in J. \text{emeasure } (M\ i)\ (X\ i)})$

$\langle \text{proof} \rangle$

sublocale *product-prob-space* $\subseteq P?$: *prob-space* $Pi_M\ I\ M$

$\langle \text{proof} \rangle$

lemma *prob-space-PiM*:

assumes M : $\bigwedge i. i \in I \implies \text{prob-space } (M\ i)$ **shows** *prob-space* $(PiM\ I\ M)$

$\langle \text{proof} \rangle$

lemma (*in product-prob-space*) *emeasure-PiM-Collect*:

assumes X : $J \subseteq I$ *finite* $J\ \bigwedge i. i \in J \implies X\ i \in \text{sets } (M\ i)$

shows $\text{emeasure } (Pi_M\ I\ M)\ \{x \in \text{space } (Pi_M\ I\ M). \forall i \in J. x\ i \in X\ i\} = (\prod_{i \in J. \text{emeasure } (M\ i)\ (X\ i)})$

emeasure ($M\ i$) ($X\ i$)
 $\langle proof \rangle$

lemma (in *product-prob-space*) *emeasure-PiM-Collect-single*:

assumes $X: i \in I \ A \in \text{sets } (M\ i)$

shows $\text{emeasure } (Pi_M\ I\ M) \ \{x \in \text{space } (Pi_M\ I\ M). \ x\ i \in A\} = \text{emeasure } (M\ i) \ A$

$\langle proof \rangle$

lemma (in *product-prob-space*) *measure-PiM-emb*:

assumes $J \subseteq I$ *finite* $J \ \bigwedge i. i \in J \implies X\ i \in \text{sets } (M\ i)$

shows $\text{measure } (PiM\ I\ M) \ (\text{emb } I\ J \ (Pi_E\ J\ X)) = (\prod_{i \in J. \text{measure } (M\ i) \ (X\ i)})$

$\langle proof \rangle$

lemma *sets-Collect-single'*:

$i \in I \implies \{x \in \text{space } (M\ i). \ P\ x\} \in \text{sets } (M\ i) \implies \{x \in \text{space } (PiM\ I\ M). \ P\ (x\ i)\} \in \text{sets } (PiM\ I\ M)$

$\langle proof \rangle$

lemma (in *finite-product-prob-space*) *finite-measure-PiM-emb*:

$(\bigwedge i. i \in I \implies A\ i \in \text{sets } (M\ i)) \implies \text{measure } (PiM\ I\ M) \ (Pi_E\ I\ A) = (\prod_{i \in I. \text{measure } (M\ i) \ (A\ i)})$

$\langle proof \rangle$

lemma (in *product-prob-space*) *PiM-component*:

assumes $i \in I$

shows $\text{distr } (PiM\ I\ M) \ (M\ i) \ (\lambda \omega. \ \omega\ i) = M\ i$

$\langle proof \rangle$

lemma (in *product-prob-space*) *PiM-eq*:

assumes $M': \text{sets } M' = \text{sets } (PiM\ I\ M)$

assumes $\text{eq}: \bigwedge J\ F. \text{finite } J \implies J \subseteq I \implies (\bigwedge j. j \in J \implies F\ j \in \text{sets } (M\ j)) \implies \text{emeasure } M' \ (\text{prod-emb } I\ M\ J \ (\Pi_E\ j \in J. \ F\ j)) = (\prod_{j \in J. \text{emeasure } (M\ j) \ (F\ j)})$

shows $M' = (PiM\ I\ M)$

$\langle proof \rangle$

lemma (in *product-prob-space*) *AE-component*: $i \in I \implies AE\ x \text{ in } M\ i. \ P\ x \implies AE\ x \text{ in } PiM\ I\ M. \ P\ (x\ i)$

$\langle proof \rangle$

lemma *emeasure-PiM-emb*:

assumes $M: \bigwedge i. i \in I \implies \text{prob-space } (M\ i)$

assumes $J: J \subseteq I$ *finite* J **and** $A: \bigwedge i. i \in J \implies A\ i \in \text{sets } (M\ i)$

shows $\text{emeasure } (Pi_M\ I\ M) \ (\text{prod-emb } I\ M\ J \ (Pi_E\ J\ A)) = (\prod_{i \in J. \text{emeasure } (M\ i) \ (A\ i)})$

$\langle proof \rangle$

lemma *distr-pair-PiM-eq-PiM*:

fixes $i' :: 'i$ **and** $I :: 'i$ **set and** $M :: 'i \Rightarrow 'a$ **measure**
assumes $M: \bigwedge i. i \in I \implies \text{prob-space } (M\ i) \text{ prob-space } (M\ i')$
shows $\text{distr } (M\ i' \otimes_M (\Pi_M\ i \in I. M\ i)) (\Pi_M\ i \in \text{insert } i' I. M\ i) (\lambda(x, X). X(i' := x)) =$
 $(\Pi_M\ i \in \text{insert } i' I. M\ i) \text{ (is ?L = -)}$
 $\langle \text{proof} \rangle$

lemma *distr-PiM-reindex*:
assumes $M: \bigwedge i. i \in K \implies \text{prob-space } (M\ i)$
assumes $f: \text{inj-on } f\ I\ f \in I \rightarrow K$
shows $\text{distr } (Pi_M\ K\ M) (\Pi_M\ i \in I. M\ (f\ i)) (\lambda\omega. \lambda n \in I. \omega\ (f\ n)) = (\Pi_M\ i \in I. M\ (f\ i))$
 $\text{(is distr ?K ?I ?t = ?I)}$
 $\langle \text{proof} \rangle$

lemma *distr-PiM-component*:
assumes $M: \bigwedge i. i \in I \implies \text{prob-space } (M\ i)$
assumes $i \in I$
shows $\text{distr } (Pi_M\ I\ M) (M\ i) (\lambda\omega. \omega\ i) = M\ i$
 $\langle \text{proof} \rangle$

lemma *AE-PiM-component*:
 $(\bigwedge i. i \in I \implies \text{prob-space } (M\ i)) \implies i \in I \implies \text{AE } x \text{ in } M\ i. P\ x \implies \text{AE } x \text{ in } Pi_M\ I\ M. P\ (x\ i)$
 $\langle \text{proof} \rangle$

lemma *decseq-emb-PiE*:
 $\text{incseq } J \implies \text{decseq } (\lambda i. \text{prod-emb } I\ M\ (J\ i) (\Pi_E\ j \in J\ i. X\ j))$
 $\langle \text{proof} \rangle$

9.1 Sequence space

definition *comb-seq* $:: \text{nat} \Rightarrow (\text{nat} \Rightarrow 'a) \Rightarrow (\text{nat} \Rightarrow 'a) \Rightarrow (\text{nat} \Rightarrow 'a)$ **where**
 $\text{comb-seq } i\ \omega\ \omega'\ j = (\text{if } j < i \text{ then } \omega\ j \text{ else } \omega'\ (j - i))$

lemma *split-comb-seq*: $P\ (\text{comb-seq } i\ \omega\ \omega'\ j) \longleftrightarrow (j < i \longrightarrow P\ (\omega\ j)) \wedge (\forall k. j = i + k \longrightarrow P\ (\omega'\ k))$
 $\langle \text{proof} \rangle$

lemma *split-comb-seq-asm*: $P\ (\text{comb-seq } i\ \omega\ \omega'\ j) \longleftrightarrow \neg ((j < i \wedge \neg P\ (\omega\ j)) \vee (\exists k. j = i + k \wedge \neg P\ (\omega'\ k)))$
 $\langle \text{proof} \rangle$

lemma *measurable-comb-seq*:
 $(\lambda(\omega, \omega'). \text{comb-seq } i\ \omega\ \omega') \in \text{measurable } ((\Pi_M\ i \in UNIV. M) \otimes_M (\Pi_M\ i \in UNIV. M)) (\Pi_M\ i \in UNIV. M)$
 $\langle \text{proof} \rangle$

lemma *measurable-comb-seq'[measurable (raw)]*:

assumes $f: f \in \text{measurable } N \ (\Pi_M \ i \in \text{UNIV}. M)$ **and** $g: g \in \text{measurable } N \ (\Pi_M \ i \in \text{UNIV}. M)$

shows $(\lambda x. \text{comb-seq } i \ (f \ x) \ (g \ x)) \in \text{measurable } N \ (\Pi_M \ i \in \text{UNIV}. M)$
 $\langle \text{proof} \rangle$

lemma *comb-seq-0*: $\text{comb-seq } 0 \ \omega \ \omega' = \omega'$
 $\langle \text{proof} \rangle$

lemma *comb-seq-Suc*: $\text{comb-seq } (\text{Suc } n) \ \omega \ \omega' = \text{comb-seq } n \ \omega \ (\text{case-nat } (\omega \ n) \ \omega')$
 $\langle \text{proof} \rangle$

lemma *comb-seq-Suc-0[simp]*: $\text{comb-seq } (\text{Suc } 0) \ \omega = \text{case-nat } (\omega \ 0)$
 $\langle \text{proof} \rangle$

lemma *comb-seq-less*: $i < n \implies \text{comb-seq } n \ \omega \ \omega' \ i = \omega \ i$
 $\langle \text{proof} \rangle$

lemma *comb-seq-add*: $\text{comb-seq } n \ \omega \ \omega' \ (i + n) = \omega' \ i$
 $\langle \text{proof} \rangle$

lemma *case-nat-comb-seq*: $\text{case-nat } s' \ (\text{comb-seq } n \ \omega \ \omega') \ (i + n) = \text{case-nat } (\text{case-nat } s' \ \omega \ n) \ \omega' \ i$
 $\langle \text{proof} \rangle$

lemma *case-nat-comb-seq'*:
 $\text{case-nat } s \ (\text{comb-seq } i \ \omega \ \omega') = \text{comb-seq } (\text{Suc } i) \ (\text{case-nat } s \ \omega) \ \omega'$
 $\langle \text{proof} \rangle$

locale *sequence-space* = *product-prob-space* $\lambda i. M \ \text{UNIV} :: \text{nat set}$ **for** M
begin

abbreviation $S \equiv \Pi_M \ i \in \text{UNIV} :: \text{nat set}. M$

lemma *infprod-in-sets[intro]*:
fixes $E :: \text{nat} \Rightarrow 'a \text{ set}$ **assumes** $E: \bigwedge i. E \ i \in \text{sets } M$
shows $\Pi i \ \text{UNIV} \ E \in \text{sets } S$
 $\langle \text{proof} \rangle$

lemma *measure-PiM-countable*:
fixes $E :: \text{nat} \Rightarrow 'a \text{ set}$ **assumes** $E: \bigwedge i. E \ i \in \text{sets } M$
shows $(\lambda n. \prod_{i \leq n}. \text{measure } M \ (E \ i)) \longrightarrow \text{measure } S \ (\Pi i \ \text{UNIV} \ E)$
 $\langle \text{proof} \rangle$

lemma *nat-eq-diff-eq*:
fixes $a \ b \ c :: \text{nat}$
shows $c \leq b \implies a = b - c \longleftrightarrow a + c = b$
 $\langle \text{proof} \rangle$

lemma *PiM-comb-seq*:

$distr (S \otimes_M S) S (\lambda(\omega, \omega'). comb-seq i \omega \omega') = S (is ?D = -)$
 $\langle proof \rangle$

lemma *PiM-iter*:

$distr (M \otimes_M S) S (\lambda(s, \omega). case-nat s \omega) = S (is ?D = -)$
 $\langle proof \rangle$

end

lemma *PiM-return*:

assumes *finite I*
assumes [*measurable*]: $\bigwedge i. i \in I \implies \{a\ i\} \in sets\ (M\ i)$
shows $PiM\ I\ (\lambda i. return\ (M\ i)\ (a\ i)) = return\ (PiM\ I\ M)\ (restrict\ a\ I)$
 $\langle proof \rangle$

lemma *distr-PiM-finite-prob-space'*:

assumes *fin: finite I*
assumes $\bigwedge i. i \in I \implies prob-space\ (M\ i)$
assumes $\bigwedge i. i \in I \implies prob-space\ (M'\ i)$
assumes [*measurable*]: $\bigwedge i. i \in I \implies f \in measurable\ (M\ i)\ (M'\ i)$
shows $distr\ (PiM\ I\ M)\ (PiM\ I\ M')\ (compose\ I\ f) = PiM\ I\ (\lambda i. distr\ (M\ i)\ (M'\ i)\ f)$
 $\langle proof \rangle$

end

10 Independent families of events, event sets, and random variables

theory *Independent-Family*

imports *Infinite-Product-Measure*

begin

definition (*in prob-space*)

$indep-sets\ F\ I \longleftrightarrow (\forall i \in I. F\ i \subseteq events) \wedge$
 $(\forall J \subseteq I. J \neq \{\} \longrightarrow finite\ J \longrightarrow (\forall A \in Pi\ J\ F. prob\ (\bigcap_{j \in J}. A\ j) = (\prod_{j \in J}. prob\ (A\ j))))$

definition (*in prob-space*)

$indep-set\ A\ B \longleftrightarrow indep-sets\ (case-bool\ A\ B)\ UNIV$

definition (*in prob-space*)

$indep-events-def-alt: indep-events\ A\ I \longleftrightarrow indep-sets\ (\lambda i. \{A\ i\})\ I$

lemma (*in prob-space*) *indep-events-def*:

$indep-events\ A\ I \longleftrightarrow (A\ I \subseteq events) \wedge$
 $(\forall J \subseteq I. J \neq \{\} \longrightarrow finite\ J \longrightarrow prob\ (\bigcap_{j \in J}. A\ j) = (\prod_{j \in J}. prob\ (A\ j)))$
 $\langle proof \rangle$

lemma (in *prob-space*) *indep-eventsI*:

$(\bigwedge i. i \in I \implies F\ i \in \text{sets } M) \implies (\bigwedge J. J \subseteq I \implies \text{finite } J \implies J \neq \{\} \implies \text{prob} (\bigcap_{i \in J}. F\ i) = (\prod_{i \in J}. \text{prob } (F\ i))) \implies \text{indep-events } F\ I$
 ⟨proof⟩

definition (in *prob-space*)

indep-event $A\ B \longleftrightarrow \text{indep-events } (\text{case-bool } A\ B)\ \text{UNIV}$

lemma (in *prob-space*) *indep-sets-cong*:

$I = J \implies (\bigwedge i. i \in I \implies F\ i = G\ i) \implies \text{indep-sets } F\ I \longleftrightarrow \text{indep-sets } G\ J$
 ⟨proof⟩

lemma (in *prob-space*) *indep-events-finite-index-events*:

$\text{indep-events } F\ I \longleftrightarrow (\forall J \subseteq I. J \neq \{\} \longrightarrow \text{finite } J \longrightarrow \text{indep-events } F\ J)$
 ⟨proof⟩

lemma (in *prob-space*) *indep-sets-finite-index-sets*:

$\text{indep-sets } F\ I \longleftrightarrow (\forall J \subseteq I. J \neq \{\} \longrightarrow \text{finite } J \longrightarrow \text{indep-sets } F\ J)$
 ⟨proof⟩

lemma (in *prob-space*) *indep-sets-mono-index*:

$J \subseteq I \implies \text{indep-sets } F\ I \implies \text{indep-sets } F\ J$
 ⟨proof⟩

lemma (in *prob-space*) *indep-sets-mono-sets*:

assumes *indep*: $\text{indep-sets } F\ I$
assumes *mono*: $\bigwedge i. i \in I \implies G\ i \subseteq F\ i$
shows $\text{indep-sets } G\ I$
 ⟨proof⟩

lemma (in *prob-space*) *indep-sets-mono*:

assumes *indep*: $\text{indep-sets } F\ I$
assumes *mono*: $J \subseteq I \bigwedge i. i \in J \implies G\ i \subseteq F\ i$
shows $\text{indep-sets } G\ J$
 ⟨proof⟩

lemma (in *prob-space*) *indep-setsI*:

assumes $\bigwedge i. i \in I \implies F\ i \subseteq \text{events}$
and $\bigwedge A\ J. J \neq \{\} \implies J \subseteq I \implies \text{finite } J \implies (\forall j \in J. A\ j \in F\ j) \implies \text{prob} (\bigcap_{j \in J}. A\ j) = (\prod_{j \in J}. \text{prob } (A\ j))$
shows $\text{indep-sets } F\ I$
 ⟨proof⟩

lemma (in *prob-space*) *indep-setsD*:

assumes $\text{indep-sets } F\ I$ **and** $J \subseteq I\ J \neq \{\} \text{ finite } J\ \forall j \in J. A\ j \in F\ j$
shows $\text{prob } (\bigcap_{j \in J}. A\ j) = (\prod_{j \in J}. \text{prob } (A\ j))$
 ⟨proof⟩

lemma (in *prob-space*) *indep-setI*:
 assumes *ev*: $A \subseteq \text{events}$ $B \subseteq \text{events}$
 and *indep*: $\bigwedge a b. a \in A \implies b \in B \implies \text{prob } (a \cap b) = \text{prob } a * \text{prob } b$
 shows *indep-set* $A B$
 ⟨*proof*⟩

lemma (in *prob-space*) *indep-setD*:
 assumes *indep*: *indep-set* $A B$ and *ev*: $a \in A \ b \in B$
 shows $\text{prob } (a \cap b) = \text{prob } a * \text{prob } b$
 ⟨*proof*⟩

lemma (in *prob-space*)
 assumes *indep*: *indep-set* $A B$
 shows *indep-setD-ev1*: $A \subseteq \text{events}$
 and *indep-setD-ev2*: $B \subseteq \text{events}$
 ⟨*proof*⟩

lemma (in *prob-space*) *indep-sets-Dynkin*:
 assumes *indep*: *indep-sets* $F I$
 shows *indep-sets* $(\lambda i. \text{Dynkin } (\text{space } M) (F i)) I$
 (is *indep-sets ?F I*)
 ⟨*proof*⟩

lemma (in *prob-space*) *indep-sets-sigma*:
 assumes *indep*: *indep-sets* $F I$
 assumes *stable*: $\bigwedge i. i \in I \implies \text{Int-stable } (F i)$
 shows *indep-sets* $(\lambda i. \text{sigma-sets } (\text{space } M) (F i)) I$
 ⟨*proof*⟩

lemma (in *prob-space*) *indep-sets-sigma-sets-iff*:
 assumes $\bigwedge i. i \in I \implies \text{Int-stable } (F i)$
 shows *indep-sets* $(\lambda i. \text{sigma-sets } (\text{space } M) (F i)) I \longleftrightarrow \text{indep-sets } F I$
 ⟨*proof*⟩

definition (in *prob-space*)
indep-vars-def2: *indep-vars* $M' X I \longleftrightarrow$
 $(\forall i \in I. \text{random-variable } (M' i) (X i)) \wedge$
indep-sets $(\lambda i. \{ X i - ' A \cap \text{space } M \mid A. A \in \text{sets } (M' i) \}) I$

definition (in *prob-space*)
indep-var $Ma A Mb B \longleftrightarrow \text{indep-vars } (\text{case-bool } Ma Mb) (\text{case-bool } A B) \text{ UNIV}$

lemma (in *prob-space*) *indep-vars-def*:
indep-vars $M' X I \longleftrightarrow$
 $(\forall i \in I. \text{random-variable } (M' i) (X i)) \wedge$
indep-sets $(\lambda i. \text{sigma-sets } (\text{space } M) \{ X i - ' A \cap \text{space } M \mid A. A \in \text{sets } (M' i) \}) I$
 ⟨*proof*⟩

lemma (in prob-space) indep-var-eq:

indep-var $S\ X\ T\ Y \longleftrightarrow$
 $(\text{random-variable } S\ X \wedge \text{random-variable } T\ Y) \wedge$
indep-set
 $(\text{sigma-sets } (\text{space } M) \{ X - 'A \cap \text{space } M \mid A. A \in \text{sets } S \})$
 $(\text{sigma-sets } (\text{space } M) \{ Y - 'A \cap \text{space } M \mid A. A \in \text{sets } T \})$
 ⟨proof⟩

lemma (in prob-space) indep-sets2-eq:

indep-set $A\ B \longleftrightarrow A \subseteq \text{events} \wedge B \subseteq \text{events} \wedge (\forall a \in A. \forall b \in B. \text{prob } (a \cap b) =$
 $\text{prob } a * \text{prob } b)$
 ⟨proof⟩

lemma (in prob-space) indep-set-sigma-sets:

assumes *indep-set* $A\ B$
assumes A : Int-stable A **and** B : Int-stable B
shows *indep-set* $(\text{sigma-sets } (\text{space } M) A) (\text{sigma-sets } (\text{space } M) B)$
 ⟨proof⟩

lemma (in prob-space) indep-eventsI-indep-vars:

assumes *indep*: *indep-vars* $N\ X\ I$
assumes P : $\bigwedge i. i \in I \implies \{x \in \text{space } (N\ i). P\ i\ x\} \in \text{sets } (N\ i)$
shows *indep-events* $(\lambda i. \{x \in \text{space } M. P\ i\ (X\ i\ x)\})\ I$
 ⟨proof⟩

lemma (in prob-space) indep-sets-collect-sigma:

fixes $I :: 'j \Rightarrow 'i\ \text{set}$ **and** $J :: 'j\ \text{set}$ **and** $E :: 'i \Rightarrow 'a\ \text{set set}$
assumes *indep*: *indep-sets* $E\ (\bigcup j \in J. I\ j)$
assumes Int-stable: $\bigwedge i\ j. j \in J \implies i \in I\ j \implies \text{Int-stable } (E\ i)$
assumes disjoint: disjoint-family-on $I\ J$
shows *indep-sets* $(\lambda j. \text{sigma-sets } (\text{space } M) (\bigcup i \in I\ j. E\ i))\ J$
 ⟨proof⟩

lemma (in prob-space) indep-vars-restrict:

assumes ind: *indep-vars* $M'\ X\ I$ **and** K : $\bigwedge j. j \in L \implies K\ j \subseteq I$ **and** J :
 disjoint-family-on $K\ L$
shows *indep-vars* $(\lambda j. \text{PiM } (K\ j)\ M') (\lambda j\ \omega. \text{restrict } (\lambda i. X\ i\ \omega) (K\ j))\ L$
 ⟨proof⟩

lemma (in prob-space) indep-var-restrict:

assumes ind: *indep-vars* $M'\ X\ I$ **and** AB : $A \cap B = \{\}$ $A \subseteq I\ B \subseteq I$
shows *indep-var* $(\text{PiM } A\ M') (\lambda \omega. \text{restrict } (\lambda i. X\ i\ \omega) A) (\text{PiM } B\ M') (\lambda \omega.$
 $\text{restrict } (\lambda i. X\ i\ \omega) B)$
 ⟨proof⟩

lemma (in prob-space) indep-vars-subset:

assumes *indep-vars* $M'\ X\ I\ J \subseteq I$
shows *indep-vars* $M'\ X\ J$
 ⟨proof⟩

lemma (in *prob-space*) *indep-vars-cong*:

$I = J \implies (\bigwedge i. i \in I \implies X\ i = Y\ i) \implies (\bigwedge i. i \in I \implies M'\ i = N'\ i) \implies$
indep-vars $M'\ X\ I \longleftrightarrow \text{indep-vars } N'\ Y\ J$
 ⟨proof⟩

definition (in *prob-space*) *tail-events* **where**

tail-events $A = (\bigcap n. \text{sigma-sets } (\text{space } M) (\bigcup (A\ ' \{n..\})))$

lemma (in *prob-space*) *tail-events-sets*:

assumes $A: \bigwedge i::\text{nat}. A\ i \subseteq \text{events}$
shows *tail-events* $A \subseteq \text{events}$
 ⟨proof⟩

lemma (in *prob-space*) *sigma-algebra-tail-events*:

assumes $\bigwedge i::\text{nat}. \text{sigma-algebra } (\text{space } M) (A\ i)$
shows *sigma-algebra* $(\text{space } M) (\text{tail-events } A)$
 ⟨proof⟩

lemma (in *prob-space*) *kolmogorov-0-1-law*:

fixes $A :: \text{nat} \Rightarrow 'a \text{ set set}$
assumes $\bigwedge i::\text{nat}. \text{sigma-algebra } (\text{space } M) (A\ i)$
assumes *indep*: *indep-sets* $A\ \text{UNIV}$
and $X: X \in \text{tail-events } A$
shows $\text{prob } X = 0 \vee \text{prob } X = 1$
 ⟨proof⟩

lemma (in *prob-space*) *borel-0-1-law*:

fixes $F :: \text{nat} \Rightarrow 'a \text{ set}$
assumes $F2: \text{indep-events } F\ \text{UNIV}$
shows $\text{prob } (\bigcap n. \bigcup m \in \{n..\}. F\ m) = 0 \vee \text{prob } (\bigcap n. \bigcup m \in \{n..\}. F\ m) = 1$
 ⟨proof⟩

lemma (in *prob-space*) *borel-0-1-law-AE*:

fixes $P :: \text{nat} \Rightarrow 'a \Rightarrow \text{bool}$
assumes *indep-events* $(\lambda m. \{x \in \text{space } M. P\ m\ x\})\ \text{UNIV}$ (**is** *indep-events* $?P\ -$)
shows $(AE\ x\ \text{in } M. \text{infinite } \{m. P\ m\ x\}) \vee (AE\ x\ \text{in } M. \text{finite } \{m. P\ m\ x\})$
 ⟨proof⟩

lemma (in *prob-space*) *indep-sets-finite*:

assumes $I: I \neq \{\}$ *finite* I
and $F: \bigwedge i. i \in I \implies F\ i \subseteq \text{events} \bigwedge i. i \in I \implies \text{space } M \in F\ i$
shows *indep-sets* $F\ I \longleftrightarrow (\forall A \in \text{Pi } I\ F. \text{prob } (\bigcap j \in I. A\ j) = (\prod j \in I. \text{prob } (A\ j)))$
 ⟨proof⟩

lemma (in *prob-space*) *indep-vars-finite*:

fixes $I :: 'i \text{ set}$
assumes $I: I \neq \{\}$ *finite* I

and M' : $\bigwedge i. i \in I \implies \text{sets } (M' i) = \text{sigma-sets } (\text{space } (M' i)) (E i)$
and rv : $\bigwedge i. i \in I \implies \text{random-variable } (M' i) (X i)$
and Int-stable : $\bigwedge i. i \in I \implies \text{Int-stable } (E i)$
and space : $\bigwedge i. i \in I \implies \text{space } (M' i) \in E i$ **and** closed : $\bigwedge i. i \in I \implies E i \subseteq \text{Pow } (\text{space } (M' i))$
shows $\text{indep-vars } M' X I \longleftrightarrow$
 $(\forall A \in (\prod_{i \in I}. E i). \text{prob } (\bigcap_{j \in I}. X j - 'A j \cap \text{space } M) = (\prod_{j \in I}. \text{prob } (X j - 'A j \cap \text{space } M)))$
 $\langle \text{proof} \rangle$

lemma (**in** prob-space) $\text{indep-vars-compose}$:
assumes $\text{indep-vars } M' X I$
assumes rv : $\bigwedge i. i \in I \implies Y i \in \text{measurable } (M' i) (N i)$
shows $\text{indep-vars } N (\lambda i. Y i \circ X i) I$
 $\langle \text{proof} \rangle$

lemma (**in** prob-space) $\text{indep-vars-compose2}$:
assumes $\text{indep-vars } M' X I$
assumes rv : $\bigwedge i. i \in I \implies Y i \in \text{measurable } (M' i) (N i)$
shows $\text{indep-vars } N (\lambda i x. Y i (X i x)) I$
 $\langle \text{proof} \rangle$

lemma (**in** prob-space) indep-var-compose :
assumes $\text{indep-var } M1 X1 M2 X2 Y1 \in \text{measurable } M1 N1 Y2 \in \text{measurable } M2 N2$
shows $\text{indep-var } N1 (Y1 \circ X1) N2 (Y2 \circ X2)$
 $\langle \text{proof} \rangle$

lemma (**in** prob-space) indep-vars-Min :
fixes $X :: 'i \Rightarrow 'a \Rightarrow \text{real}$
assumes I : $\text{finite } I \ i \notin I$ **and** indep : $\text{indep-vars } (\lambda -. \text{borel}) X (\text{insert } i I)$
shows $\text{indep-var borel } (X i) \text{ borel } (\lambda \omega. \text{Min } ((\lambda i. X i \omega) 'I))$
 $\langle \text{proof} \rangle$

lemma (**in** prob-space) indep-vars-sum :
fixes $X :: 'i \Rightarrow 'a \Rightarrow \text{real}$
assumes I : $\text{finite } I \ i \notin I$ **and** indep : $\text{indep-vars } (\lambda -. \text{borel}) X (\text{insert } i I)$
shows $\text{indep-var borel } (X i) \text{ borel } (\lambda \omega. \sum_{i \in I}. X i \omega)$
 $\langle \text{proof} \rangle$

lemma (**in** prob-space) indep-vars-prod :
fixes $X :: 'i \Rightarrow 'a \Rightarrow \text{real}$
assumes I : $\text{finite } I \ i \notin I$ **and** indep : $\text{indep-vars } (\lambda -. \text{borel}) X (\text{insert } i I)$
shows $\text{indep-var borel } (X i) \text{ borel } (\lambda \omega. \prod_{i \in I}. X i \omega)$
 $\langle \text{proof} \rangle$

lemma (**in** prob-space) $\text{indep-varsD-finite}$:
assumes X : $\text{indep-vars } M' X I$
assumes I : $I \neq \{\}$ $\text{finite } I \ \bigwedge i. i \in I \implies A i \in \text{sets } (M' i)$

shows $\text{prob} (\bigcap_{i \in I}. X\ i - 'A\ i \cap \text{space } M) = (\prod_{i \in I}. \text{prob} (X\ i - 'A\ i \cap \text{space } M))$
 $\langle \text{proof} \rangle$

lemma (in *prob-space*) *indep-varsD*:
assumes X : *indep-vars* M' $X\ I$
assumes I : $J \neq \{\}$ *finite* $J\ J \subseteq I \wedge i. i \in J \implies A\ i \in \text{sets } (M'\ i)$
shows $\text{prob} (\bigcap_{i \in J}. X\ i - 'A\ i \cap \text{space } M) = (\prod_{i \in J}. \text{prob} (X\ i - 'A\ i \cap \text{space } M))$
 $\langle \text{proof} \rangle$

lemma (in *prob-space*) *indep-vars-iff-distr-eq-PiM*:
fixes $I :: 'i\ \text{set}$ **and** $X :: 'i \Rightarrow 'a \Rightarrow 'b$
assumes $I \neq \{\}$
assumes rv : $\bigwedge i. \text{random-variable } (M'\ i) (X\ i)$
shows *indep-vars* M' $X\ I \longleftrightarrow$
 $\text{distr } M (\prod_M i \in I. M'\ i) (\lambda x. \lambda i \in I. X\ i\ x) = (\prod_M i \in I. \text{distr } M (M'\ i) (X\ i))$
 $\langle \text{proof} \rangle$

lemma (in *prob-space*) *indep-vars-iff-distr-eq-PiM'*:
fixes $I :: 'i\ \text{set}$ **and** $X :: 'i \Rightarrow 'a \Rightarrow 'b$
assumes $I \neq \{\}$
assumes rv : $\bigwedge i. i \in I \implies \text{random-variable } (M'\ i) (X\ i)$
shows *indep-vars* M' $X\ I \longleftrightarrow$
 $\text{distr } M (\prod_M i \in I. M'\ i) (\lambda x. \lambda i \in I. X\ i\ x) = (\prod_M i \in I. \text{distr } M (M'\ i) (X\ i))$
 $\langle \text{proof} \rangle$

lemma (in *prob-space*) *indep-varD*:
assumes *indep*: *indep-var* $Ma\ A\ Mb\ B$
assumes *sets*: $Xa \in \text{sets } Ma\ Xb \in \text{sets } Mb$
shows $\text{prob} ((\lambda x. (A\ x, B\ x)) - '(Xa \times Xb) \cap \text{space } M) =$
 $\text{prob } (A - 'Xa \cap \text{space } M) * \text{prob } (B - 'Xb \cap \text{space } M)$
 $\langle \text{proof} \rangle$

lemma (in *prob-space*) *prob-indep-random-variable*:
assumes *ind[simp]*: *indep-var* $N\ X\ N\ Y$
assumes *[simp]*: $A \in \text{sets } N\ B \in \text{sets } N$
shows $\mathcal{P}(x\ \text{in } M. X\ x \in A \wedge Y\ x \in B) = \mathcal{P}(x\ \text{in } M. X\ x \in A) * \mathcal{P}(x\ \text{in } M. Y\ x \in B)$
 $\langle \text{proof} \rangle$

lemma (in *prob-space*)
assumes *indep-var* $S\ X\ T\ Y$
shows *indep-var-rv1*: *random-variable* $S\ X$
and *indep-var-rv2*: *random-variable* $T\ Y$
 $\langle \text{proof} \rangle$

lemma (in *prob-space*) *indep-var-distribution-eq*:

$indep-var\ S\ X\ T\ Y \longleftrightarrow random-variable\ S\ X \wedge random-variable\ T\ Y \wedge$
 $distr\ M\ S\ X \otimes_M distr\ M\ T\ Y = distr\ M\ (S \otimes_M T)\ (\lambda x. (X\ x, Y\ x))\ (is -$
 $\longleftrightarrow - \wedge - \wedge ?S \otimes_M ?T = ?J)$
 $\langle proof \rangle$

lemma (in prob-space) distributed-joint-indep:

assumes S : sigma-finite-measure S **and** T : sigma-finite-measure T
assumes X : distributed $M\ S\ X\ Px$ **and** Y : distributed $M\ T\ Y\ Py$
assumes indep: indep-var $S\ X\ T\ Y$
shows distributed $M\ (S \otimes_M T)\ (\lambda x. (X\ x, Y\ x))\ (\lambda(x, y). Px\ x * Py\ y)$
 $\langle proof \rangle$

lemma (in prob-space) indep-vars-nn-integral:

assumes I : finite I indep-vars $(\lambda-. borel)\ X\ I \bigwedge i. i \in I \implies 0 \leq X\ i\ \omega$
shows $(\int^+ \omega. (\prod_{i \in I}. X\ i\ \omega)\ \partial M) = (\prod_{i \in I}. \int^+ \omega. X\ i\ \omega\ \partial M)$
 $\langle proof \rangle$

lemma (in prob-space)

fixes $X :: 'i \Rightarrow 'a \Rightarrow 'b :: \{real-normed-field, banach, second-countable-topology\}$
assumes I : finite I indep-vars $(\lambda-. borel)\ X\ I \bigwedge i. i \in I \implies integrable\ M\ (X\ i)$
shows indep-vars-lebesgue-integral: $(\int \omega. (\prod_{i \in I}. X\ i\ \omega)\ \partial M) = (\prod_{i \in I}. \int \omega. X\ i\ \omega\ \partial M)$ (is ?eq)
and indep-vars-integrable: integrable $M\ (\lambda \omega. (\prod_{i \in I}. X\ i\ \omega))$ (is ?int)
 $\langle proof \rangle$

lemma (in prob-space)

fixes $X1\ X2 :: 'a \Rightarrow 'b :: \{real-normed-field, banach, second-countable-topology\}$
assumes indep-var borel $X1$ borel $X2$ integrable $M\ X1$ integrable $M\ X2$
shows indep-var-lebesgue-integral: $(\int \omega. X1\ \omega * X2\ \omega\ \partial M) = (\int \omega. X1\ \omega\ \partial M)$
 $* (\int \omega. X2\ \omega\ \partial M)$ (is ?eq)
and indep-var-integrable: integrable $M\ (\lambda \omega. X1\ \omega * X2\ \omega)$ (is ?int)
 $\langle proof \rangle$

end

11 Convolution Measure

theory Convolution

imports Independent-Family

begin

lemma (in finite-measure) sigma-finite-measure: sigma-finite-measure M
 $\langle proof \rangle$

definition convolution :: $('a :: ordered-euclidean-space)\ measure \Rightarrow 'a\ measure \Rightarrow$
 $'a\ measure$ (infix \star) 50) **where**
convolution $M\ N = distr\ (M \otimes_M N)\ borel\ (\lambda(x, y). x + y)$

lemma

shows $\text{space-convolution}[simp]: \text{space } (\text{convolution } M \ N) = \text{space borel}$
and $\text{sets-convolution}[simp]: \text{sets } (\text{convolution } M \ N) = \text{sets borel}$
and $\text{measurable-convolution1}[simp]: \text{measurable } A \ (\text{convolution } M \ N) = \text{measurable } A \ \text{borel}$
and $\text{measurable-convolution2}[simp]: \text{measurable } (\text{convolution } M \ N) \ B = \text{measurable borel } B$
 $\langle \text{proof} \rangle$

lemma $\text{nn-integral-convolution}$:

assumes $\text{finite-measure } M \ \text{finite-measure } N$
assumes $[measurable-cong]: \text{sets } N = \text{sets borel sets } M = \text{sets borel}$
assumes $[measurable]: f \in \text{borel-measurable borel}$
shows $(\int^{+x}. f \ x \ \partial \text{convolution } M \ N) = (\int^{+x}. \int^{+y}. f \ (x + y) \ \partial N \ \partial M)$
 $\langle \text{proof} \rangle$

lemma $\text{convolution-emeasure}$:

assumes $A \in \text{sets borel finite-measure } M \ \text{finite-measure } N$
assumes $[simp]: \text{sets } N = \text{sets borel sets } M = \text{sets borel}$
assumes $[simp]: \text{space } M = \text{space } N \ \text{space } N = \text{space borel}$
shows $\text{emeasure } (M \star N) \ A = \int^{+x}. (\text{emeasure } N \ \{a. a + x \in A\}) \ \partial M$
 $\langle \text{proof} \rangle$

lemma $\text{convolution-emeasure}'$:

assumes $[simp]: A \in \text{sets borel}$
assumes $[simp]: \text{finite-measure } M \ \text{finite-measure } N$
assumes $[simp]: \text{sets } N = \text{sets borel sets } M = \text{sets borel}$
shows $\text{emeasure } (M \star N) \ A = \int^{+x}. \int^{+y}. (\text{indicator } A \ (x + y)) \ \partial N \ \partial M$
 $\langle \text{proof} \rangle$

lemma $\text{convolution-finite}$:

assumes $[simp]: \text{finite-measure } M \ \text{finite-measure } N$
assumes $[measurable-cong]: \text{sets } N = \text{sets borel sets } M = \text{sets borel}$
shows $\text{finite-measure } (M \star N)$
 $\langle \text{proof} \rangle$

lemma $\text{convolution-emeasure-3}$:

assumes $[simp, measurable]: A \in \text{sets borel}$
assumes $[simp]: \text{finite-measure } M \ \text{finite-measure } N \ \text{finite-measure } L$
assumes $[simp]: \text{sets } N = \text{sets borel sets } M = \text{sets borel sets } L = \text{sets borel}$
shows $\text{emeasure } (L \star (M \star N)) \ A = \int^{+x}. \int^{+y}. \int^{+z}. \text{indicator } A \ (x + y + z) \ \partial N \ \partial M \ \partial L$
 $\langle \text{proof} \rangle$

lemma $\text{convolution-emeasure-3}'$:

assumes $[simp, measurable]: A \in \text{sets borel}$
assumes $[simp]: \text{finite-measure } M \ \text{finite-measure } N \ \text{finite-measure } L$
assumes $[measurable-cong, simp]: \text{sets } N = \text{sets borel sets } M = \text{sets borel sets } L = \text{sets borel}$
shows $\text{emeasure } ((L \star M) \star N) \ A = \int^{+x}. \int^{+y}. \int^{+z}. \text{indicator } A \ (x + y + z)$

$\partial N \ \partial M \ \partial L$
 $\langle \text{proof} \rangle$

lemma *convolution-commutative:*

assumes $[simp]$: *finite-measure* M *finite-measure* N
assumes $[measurable-cong, simp]$: *sets* $N = \text{sets borel}$ *sets* $M = \text{sets borel}$
shows $(M \star N) = (N \star M)$
 $\langle \text{proof} \rangle$

lemma *convolution-associative:*

assumes $[simp]$: *finite-measure* M *finite-measure* N *finite-measure* L
assumes $[simp]$: *sets* $N = \text{sets borel}$ *sets* $M = \text{sets borel}$ *sets* $L = \text{sets borel}$
shows $(L \star (M \star N)) = ((L \star M) \star N)$
 $\langle \text{proof} \rangle$

lemma (*in prob-space*) *sum-indep-random-variable:*

assumes *ind: indep-var borel* X *borel* Y
assumes $[simp, measurable]$: *random-variable borel* X
assumes $[simp, measurable]$: *random-variable borel* Y
shows *distr* M *borel* $(\lambda x. X \ x + Y \ x) = \text{convolution} \ (\text{distr } M \ \text{borel } X) \ (\text{distr } M \ \text{borel } Y)$
 $\langle \text{proof} \rangle$

lemma (*in prob-space*) *sum-indep-random-variable-lborel:*

assumes *ind: indep-var borel* X *borel* Y
assumes $[simp, measurable]$: *random-variable lborel* X
assumes $[simp, measurable]$: *random-variable lborel* Y
shows *distr* M *lborel* $(\lambda x. X \ x + Y \ x) = \text{convolution} \ (\text{distr } M \ \text{lborel } X) \ (\text{distr } M \ \text{lborel } Y)$
 $\langle \text{proof} \rangle$

lemma *convolution-density:*

fixes $f \ g :: \text{real} \Rightarrow \text{ennreal}$
assumes $[measurable]$: $f \in \text{borel-measurable borel}$ $g \in \text{borel-measurable borel}$
assumes $[simp]$: *finite-measure* $(\text{density lborel } f)$ *finite-measure* $(\text{density lborel } g)$
shows *density lborel* $f \star \text{density lborel } g = \text{density lborel } (\lambda x. \int^+ y. f \ (x - y) * g \ y \ \partial \text{lborel})$
 $(\text{is } ?l = ?r)$
 $\langle \text{proof} \rangle$

lemma (*in prob-space*) *distributed-finite-measure-density:*

distributed $M \ N \ X \ f \Longrightarrow \text{finite-measure} \ (\text{density } N \ f)$
 $\langle \text{proof} \rangle$

lemma (*in prob-space*) *distributed-convolution:*

fixes $f :: \text{real} \Rightarrow -$
fixes $g :: \text{real} \Rightarrow -$
assumes *indep: indep-var borel* X *borel* Y

```

assumes  $X$ : distributed  $M$  lborel  $X$   $f$ 
assumes  $Y$ : distributed  $M$  lborel  $Y$   $g$ 
shows distributed  $M$  lborel  $(\lambda x. X\ x + Y\ x) (\lambda x. \int^+ y. f\ (x - y) * g\ y\ \partial \text{lborel})$ 
   $\langle \text{proof} \rangle$ 

lemma prob-space-convolution-density:
  fixes  $f$ :: real  $\Rightarrow$  -
  fixes  $g$ :: real  $\Rightarrow$  -
  assumes [measurable]:  $f \in \text{borel-measurable borel}$ 
  assumes [measurable]:  $g \in \text{borel-measurable borel}$ 
  assumes  $gt-0[\text{simp}]$ :  $\bigwedge x. 0 \leq f\ x \ \bigwedge x. 0 \leq g\ x$ 
  assumes prob-space (density lborel  $f$ ) (is prob-space  $?F$ )
  assumes prob-space (density lborel  $g$ ) (is prob-space  $?G$ )
  shows prob-space (density lborel  $(\lambda x. \int^+ y. f\ (x - y) * g\ y\ \partial \text{lborel})$ ) (is prob-space  $?D$ )
   $\langle \text{proof} \rangle$ 

end

```

12 Information theory

```

theory Information
imports
  Independent-Family
begin

```

12.1 Information theory

```

locale information-space = prob-space +
  fixes  $b$  :: real assumes  $b\text{-gt-1}$ :  $1 < b$ 

```

Introduce some simplification rules for logarithm of base b .

```

lemmas log-simps = log-mult log-inverse log-divide

```

12.2 Kullback–Leibler divergence

The Kullback–Leibler divergence is also known as relative entropy or Kullback–Leibler distance.

definition

$$\text{entropy-density } b\ M\ N = \log b \circ \text{enn2real} \circ \text{RN-deriv } M\ N$$

definition

$$\text{KL-divergence } b\ M\ N = \text{integral}^L\ N\ (\text{entropy-density } b\ M\ N)$$

```

lemma measurable-entropy-density[measurable]: entropy-density  $b\ M\ N \in \text{borel-measurable } M$ 
   $\langle \text{proof} \rangle$ 

```

lemma (in *sigma-finite-measure*) *KL-density*:

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

assumes $1 < b$

assumes $f[\text{measurable}]$: $f \in \text{borel-measurable } M$ **and** nn : $\text{AE } x \text{ in } M. 0 \leq f x$

shows *KL-divergence* $b \ M \ (\text{density } M f) = (\int x. f x * \log b (f x) \ \partial M)$

<proof>

lemma (in *sigma-finite-measure*) *KL-density-density*:

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

assumes $1 < b$

assumes f : $f \in \text{borel-measurable } M$ $\text{AE } x \text{ in } M. 0 \leq f x$

assumes g : $g \in \text{borel-measurable } M$ $\text{AE } x \text{ in } M. 0 \leq g x$

assumes ac : $\text{AE } x \text{ in } M. f x = 0 \longrightarrow g x = 0$

shows *KL-divergence* $b \ (\text{density } M f) \ (\text{density } M g) = (\int x. g x * \log b (g x / f x) \ \partial M)$

<proof>

lemma (in *information-space*) *KL-gt-0*:

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

assumes *prob-space* (*density* $M \ D$)

assumes D : $D \in \text{borel-measurable } M$ $\text{AE } x \text{ in } M. 0 \leq D x$

assumes *int*: *integrable* $M \ (\lambda x. D x * \log b (D x))$

assumes A : *density* $M \ D \neq M$

shows $0 < \text{KL-divergence } b \ M \ (\text{density } M \ D)$

<proof>

lemma (in *sigma-finite-measure*) *KL-same-eq-0*: *KL-divergence* $b \ M \ M = 0$

<proof>

lemma (in *information-space*) *KL-eq-0-iff-eq*:

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

assumes *prob-space* (*density* $M \ D$)

assumes D : $D \in \text{borel-measurable } M$ $\text{AE } x \text{ in } M. 0 \leq D x$

assumes *int*: *integrable* $M \ (\lambda x. D x * \log b (D x))$

shows *KL-divergence* $b \ M \ (\text{density } M \ D) = 0 \longleftrightarrow \text{density } M \ D = M$

<proof>

lemma (in *information-space*) *KL-eq-0-iff-eq-ac*:

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

assumes *prob-space* N

assumes ac : *absolutely-continuous* $M \ N$ *sets* $N = \text{sets } M$

assumes *int*: *integrable* $N \ (\text{entropy-density } b \ M \ N)$

shows *KL-divergence* $b \ M \ N = 0 \longleftrightarrow N = M$

<proof>

lemma (in *information-space*) *KL-nonneg*:

assumes *prob-space* (*density* $M \ D$)

assumes D : $D \in \text{borel-measurable } M$ $\text{AE } x \text{ in } M. 0 \leq D x$

assumes *int*: *integrable* $M \ (\lambda x. D x * \log b (D x))$

shows $0 \leq \text{KL-divergence } b \ M \ (\text{density } M \ D)$
 $\langle \text{proof} \rangle$

lemma (in *sigma-finite-measure*) *KL-density-density-nonneg*:
fixes $f \ g :: 'a \Rightarrow \text{real}$
assumes $1 < b$
assumes f : $f \in \text{borel-measurable } M \ AE \ x \ \text{in } M. \ 0 \leq f \ x \ \text{prob-space } (\text{density } M \ f)$
assumes g : $g \in \text{borel-measurable } M \ AE \ x \ \text{in } M. \ 0 \leq g \ x \ \text{prob-space } (\text{density } M \ g)$
assumes ac : $AE \ x \ \text{in } M. \ f \ x = 0 \longrightarrow g \ x = 0$
assumes int : $\text{integrable } M \ (\lambda x. \ g \ x * \log b \ (g \ x / f \ x))$
shows $0 \leq \text{KL-divergence } b \ (\text{density } M \ f) \ (\text{density } M \ g)$
 $\langle \text{proof} \rangle$

12.3 Finite Entropy

definition (in *information-space*) *finite-entropy* :: $'b \ \text{measure} \Rightarrow ('a \Rightarrow 'b) \Rightarrow ('b \Rightarrow \text{real}) \Rightarrow \text{bool}$

where

$\text{finite-entropy } S \ X \ f \longleftrightarrow$
 $\text{distributed } M \ S \ X \ f \wedge$
 $\text{integrable } S \ (\lambda x. \ f \ x * \log b \ (f \ x)) \wedge$
 $(\forall x \in \text{space } S. \ 0 \leq f \ x)$

lemma (in *information-space*) *finite-entropy-simple-function*:
assumes X : *simple-function* $M \ X$
shows $\text{finite-entropy } (\text{count-space } (X' \text{space } M)) \ X \ (\lambda a. \ \text{measure } M \ \{x \in \text{space } M. \ X \ x = a\})$
 $\langle \text{proof} \rangle$

lemma *ac-fst*:
assumes *sigma-finite-measure* T
shows *absolutely-continuous* $S \ (\text{distr } (S \otimes_M T) \ S \ \text{fst})$
 $\langle \text{proof} \rangle$

lemma *ac-snd*:
assumes *sigma-finite-measure* T
shows *absolutely-continuous* $T \ (\text{distr } (S \otimes_M T) \ T \ \text{snd})$
 $\langle \text{proof} \rangle$

lemma (in *information-space*) *finite-entropy-integrable*:
 $\text{finite-entropy } S \ X \ Px \Longrightarrow \text{integrable } S \ (\lambda x. \ Px \ x * \log b \ (Px \ x))$
 $\langle \text{proof} \rangle$

lemma (in *information-space*) *finite-entropy-distributed*:
 $\text{finite-entropy } S \ X \ Px \Longrightarrow \text{distributed } M \ S \ X \ Px$
 $\langle \text{proof} \rangle$

lemma (in *information-space*) *finite-entropy-nn*:
finite-entropy $S \ X \ Px \implies x \in \text{space } S \implies 0 \leq Px \ x$
 ⟨proof⟩

lemma (in *information-space*) *finite-entropy-measurable*:
finite-entropy $S \ X \ Px \implies Px \in S \rightarrow_M \text{borel}$
 ⟨proof⟩

lemma (in *information-space*) *subdensity-finite-entropy*:
 fixes $g :: 'b \Rightarrow \text{real}$ and $f :: 'c \Rightarrow \text{real}$
 assumes $T: T \in \text{measurable } P \ Q$
 assumes $f: \text{finite-entropy } P \ X \ f$
 assumes $g: \text{finite-entropy } Q \ Y \ g$
 assumes $Y: Y = T \circ X$
 shows $AE \ x \text{ in } P. \ g \ (T \ x) = 0 \longrightarrow f \ x = 0$
 ⟨proof⟩

lemma (in *information-space*) *finite-entropy-integrable-transform*:
finite-entropy $S \ X \ Px \implies \text{distributed } M \ T \ Y \ Py \implies (\bigwedge x. x \in \text{space } T \implies 0 \leq Py \ x) \implies$
 $X = (\lambda x. f \ (Y \ x)) \implies f \in \text{measurable } T \ S \implies \text{integrable } T \ (\lambda x. Py \ x * \log b \ (Px \ (f \ x)))$
 ⟨proof⟩

12.4 Mutual Information

definition (in *prob-space*)
mutual-information $b \ S \ T \ X \ Y =$
 $KL\text{-divergence } b \ (\text{distr } M \ S \ X \otimes_M \text{distr } M \ T \ Y) \ (\text{distr } M \ (S \otimes_M T) \ (\lambda x. (X \ x, Y \ x)))$

lemma (in *information-space*) *mutual-information-indep-vars*:
 fixes $S \ T \ X \ Y$
 defines $P \equiv \text{distr } M \ S \ X \otimes_M \text{distr } M \ T \ Y$
 defines $Q \equiv \text{distr } M \ (S \otimes_M T) \ (\lambda x. (X \ x, Y \ x))$
 shows $\text{indep-var } S \ X \ T \ Y \longleftrightarrow$
 $(\text{random-variable } S \ X \wedge \text{random-variable } T \ Y \wedge$
 $\text{absolutely-continuous } P \ Q \wedge \text{integrable } Q \ (\text{entropy-density } b \ P \ Q) \wedge$
 $\text{mutual-information } b \ S \ T \ X \ Y = 0)$
 ⟨proof⟩

abbreviation (in *information-space*)
mutual-information-Pow $(\langle \mathcal{I}'(- ; -) \rangle)$ **where**
 $\mathcal{I}(X ; Y) \equiv \text{mutual-information } b \ (\text{count-space } (X' \text{space } M)) \ (\text{count-space } (Y' \text{space } M)) \ X \ Y$

lemma (in *information-space*)
 fixes $Pxy :: 'b \times 'c \Rightarrow \text{real}$ and $Px :: 'b \Rightarrow \text{real}$ and $Py :: 'c \Rightarrow \text{real}$
 assumes $S: \text{sigma-finite-measure } S$ and $T: \text{sigma-finite-measure } T$

assumes Fx : *finite-entropy* $S \ X \ Px$ **and** Fy : *finite-entropy* $T \ Y \ Py$
assumes Fxy : *finite-entropy* $(S \otimes_M T) (\lambda x. (X \ x, Y \ x)) \ Pxy$
defines $f \equiv \lambda x. Pxy \ x * \log b (Pxy \ x / (Px \ (fst \ x) * Py \ (snd \ x)))$
shows *mutual-information-distr'*: *mutual-information* $b \ S \ T \ X \ Y = \text{integral}^L (S \otimes_M T) f$ (**is** $?M = ?R$)
and *mutual-information-nonneg'*: $0 \leq \text{mutual-information } b \ S \ T \ X \ Y$
 $\langle \text{proof} \rangle$

lemma (*in information-space*)
fixes $Pxy :: 'b \times 'c \Rightarrow \text{real}$ **and** $Px :: 'b \Rightarrow \text{real}$ **and** $Py :: 'c \Rightarrow \text{real}$
assumes *sigma-finite-measure* S *sigma-finite-measure* T
assumes Px : *distributed* $M \ S \ X \ Px$ **and** Px -nn: $\bigwedge x. x \in \text{space } S \Rightarrow 0 \leq Px \ x$
and Py : *distributed* $M \ T \ Y \ Py$ **and** Py -nn: $\bigwedge y. y \in \text{space } T \Rightarrow 0 \leq Py \ y$
and Pxy : *distributed* $M \ (S \otimes_M T) (\lambda x. (X \ x, Y \ x)) \ Pxy$
and Pxy -nn: $\bigwedge x \ y. x \in \text{space } S \Rightarrow y \in \text{space } T \Rightarrow 0 \leq Pxy \ (x, y)$
defines $f \equiv \lambda x. Pxy \ x * \log b (Pxy \ x / (Px \ (fst \ x) * Py \ (snd \ x)))$
shows *mutual-information-distr*: *mutual-information* $b \ S \ T \ X \ Y = \text{integral}^L (S \otimes_M T) f$ (**is** $?M = ?R$)
and *mutual-information-nonneg*: *integrable* $(S \otimes_M T) f \Rightarrow 0 \leq \text{mutual-information } b \ S \ T \ X \ Y$
 $\langle \text{proof} \rangle$

lemma (*in information-space*)
fixes $Pxy :: 'b \times 'c \Rightarrow \text{real}$ **and** $Px :: 'b \Rightarrow \text{real}$ **and** $Py :: 'c \Rightarrow \text{real}$
assumes *sigma-finite-measure* S *sigma-finite-measure* T
assumes Px [*measurable*]: *distributed* $M \ S \ X \ Px$ **and** Px -nn: $\bigwedge x. x \in \text{space } S \Rightarrow 0 \leq Px \ x$
and Py [*measurable*]: *distributed* $M \ T \ Y \ Py$ **and** Py -nn: $\bigwedge x. x \in \text{space } T \Rightarrow 0 \leq Py \ x$
and Pxy [*measurable*]: *distributed* $M \ (S \otimes_M T) (\lambda x. (X \ x, Y \ x)) \ Pxy$
and Pxy -nn: $\bigwedge x. x \in \text{space } (S \otimes_M T) \Rightarrow 0 \leq Pxy \ x$
assumes ae : $AE \ x \text{ in } S. AE \ y \text{ in } T. Pxy \ (x, y) = Px \ x * Py \ y$
shows *mutual-information-eq-0*: *mutual-information* $b \ S \ T \ X \ Y = 0$
 $\langle \text{proof} \rangle$

lemma (*in information-space*) *mutual-information-simple-distributed*:
assumes X : *simple-distributed* $M \ X \ Px$ **and** Y : *simple-distributed* $M \ Y \ Py$
assumes XY : *simple-distributed* $M \ (\lambda x. (X \ x, Y \ x)) \ Pxy$
shows $\mathcal{I}(X ; Y) = (\sum (x, y) \in (\lambda x. (X \ x, Y \ x))' \text{space } M. Pxy \ (x, y) * \log b (Pxy \ (x, y) / (Px \ x * Py \ y)))$
 $\langle \text{proof} \rangle$

lemma (*in information-space*)
fixes $Pxy :: 'b \times 'c \Rightarrow \text{real}$ **and** $Px :: 'b \Rightarrow \text{real}$ **and** $Py :: 'c \Rightarrow \text{real}$
assumes Px : *simple-distributed* $M \ X \ Px$ **and** Py : *simple-distributed* $M \ Y \ Py$
assumes Pxy : *simple-distributed* $M \ (\lambda x. (X \ x, Y \ x)) \ Pxy$
assumes ae : $\forall x \in \text{space } M. Pxy \ (X \ x, Y \ x) = Px \ (X \ x) * Py \ (Y \ x)$
shows *mutual-information-eq-0-simple*: $\mathcal{I}(X ; Y) = 0$
 $\langle \text{proof} \rangle$

12.5 Entropy

definition (in *prob-space*) *entropy* :: *real* \Rightarrow '*b* *measure* \Rightarrow ('*a* \Rightarrow '*b*) \Rightarrow *real* **where**
entropy *b* *S* *X* = - *KL-divergence* *b* *S* (*distr* *M* *S* *X*)

abbreviation (in *information-space*)

entropy-Pow ($\langle \mathcal{H}'(-) \rangle$) **where**

$\mathcal{H}(X) \equiv \text{entropy } b \text{ (count-space } (X' \text{space } M)) \text{ } X$

lemma (in *prob-space*) *distributed-RN-deriv*:

assumes *X*: *distributed* *M* *S* *X* *Px*

shows *AE* *x* in *S*. *RN-deriv* *S* (*density* *S* *Px*) *x* = *Px* *x*

<proof>

lemma (in *information-space*)

fixes *X* :: '*a* \Rightarrow '*b*

assumes *X*[*measurable*]: *distributed* *M* *MX* *X* *f* **and** *nn*: $\bigwedge x. x \in \text{space } MX \Rightarrow 0 \leq f \ x$

shows *entropy-distr*: *entropy* *b* *MX* *X* = - ($\int x. f \ x * \log b \ (f \ x) \ \partial MX$) (**is** *?eq*)
<proof>

lemma (in *information-space*) *entropy-le*:

fixes *Px* :: '*b* \Rightarrow *real* **and** *MX* :: '*b* *measure*

assumes *X*[*measurable*]: *distributed* *M* *MX* *X* *Px* **and** *Px-nn[simp]*: $\bigwedge x. x \in \text{space } MX \Rightarrow 0 \leq Px \ x$

and *fin*: *emeasure* *MX* $\{x \in \text{space } MX. Px \ x \neq 0\} \neq \text{top}$

and *int*: *integrable* *MX* $(\lambda x. - Px \ x * \log b \ (Px \ x))$

shows *entropy* *b* *MX* *X* $\leq \log b \ (\text{measure } MX \ \{x \in \text{space } MX. Px \ x \neq 0\})$
<proof>

lemma (in *information-space*) *entropy-le-space*:

fixes *Px* :: '*b* \Rightarrow *real* **and** *MX* :: '*b* *measure*

assumes *X*: *distributed* *M* *MX* *X* *Px* **and** *Px-nn[simp]*: $\bigwedge x. x \in \text{space } MX \Rightarrow 0 \leq Px \ x$

and *fin*: *finite-measure* *MX*

and *int*: *integrable* *MX* $(\lambda x. - Px \ x * \log b \ (Px \ x))$

shows *entropy* *b* *MX* *X* $\leq \log b \ (\text{measure } MX \ (\text{space } MX))$
<proof>

lemma (in *information-space*) *entropy-uniform*:

assumes *X*: *distributed* *M* *MX* *X* $(\lambda x. \text{indicator } A \ x \ / \ \text{measure } MX \ A)$ (**is** *distributed* - - - *?f*)

shows *entropy* *b* *MX* *X* = $\log b \ (\text{measure } MX \ A)$
<proof>

lemma (in *information-space*) *entropy-simple-distributed*:

simple-distributed *M* *X* *f* $\Rightarrow \mathcal{H}(X) = - (\sum x \in X' \text{space } M. f \ x * \log b \ (f \ x))$

<proof>

lemma (in *information-space*) *entropy-le-card-not-0*:

assumes X : *simple-distributed* $M X f$
shows $\mathcal{H}(X) \leq \log b \text{ (card } (X \text{ ‘ space } M \cap \{x. f x \neq 0\}))$
 $\langle \text{proof} \rangle$

lemma (*in information-space*) *entropy-le-card*:
assumes X : *simple-distributed* $M X f$
shows $\mathcal{H}(X) \leq \log b \text{ (real (card } (X \text{ ‘ space } M))$
 $\langle \text{proof} \rangle$

12.6 Conditional Mutual Information

definition (*in prob-space*)
conditional-mutual-information $b MX MY MZ X Y Z \equiv$
mutual-information $b MX (MY \otimes_M MZ) X (\lambda x. (Y x, Z x)) -$
mutual-information $b MX MZ X Z$

abbreviation (*in information-space*)
conditional-mutual-information-Pow $(\langle \mathcal{I}'(-; - | -) \rangle)$ **where**
 $\mathcal{I}(X; Y | Z) \equiv \text{conditional-mutual-information } b$
 $\text{(count-space } (X \text{ ‘ space } M)) \text{ (count-space } (Y \text{ ‘ space } M)) \text{ (count-space } (Z \text{ ‘ space } M)) X Y Z$

lemma (*in information-space*)
assumes S : *sigma-finite-measure* S **and** T : *sigma-finite-measure* T **and** P : *sigma-finite-measure* P
assumes $Px[\text{measurable}]$: *distributed* $M S X Px$
and $Px\text{-nn}[\text{simp}]$: $\bigwedge x. x \in \text{space } S \implies 0 \leq Px x$
assumes $Pz[\text{measurable}]$: *distributed* $M P Z Pz$
and $Pz\text{-nn}[\text{simp}]$: $\bigwedge z. z \in \text{space } P \implies 0 \leq Pz z$
assumes $Pyz[\text{measurable}]$: *distributed* $M (T \otimes_M P) (\lambda x. (Y x, Z x)) Pyz$
and $Pyz\text{-nn}[\text{simp}]$: $\bigwedge y z. y \in \text{space } T \implies z \in \text{space } P \implies 0 \leq Pyz (y, z)$
assumes $Pxz[\text{measurable}]$: *distributed* $M (S \otimes_M P) (\lambda x. (X x, Z x)) Pxz$
and $Pxz\text{-nn}[\text{simp}]$: $\bigwedge x z. x \in \text{space } S \implies z \in \text{space } P \implies 0 \leq Pxz (x, z)$
assumes $Pxyz[\text{measurable}]$: *distributed* $M (S \otimes_M T \otimes_M P) (\lambda x. (X x, Y x, Z x)) Pxyz$
and $Pxyz\text{-nn}[\text{simp}]$: $\bigwedge x y z. x \in \text{space } S \implies y \in \text{space } T \implies z \in \text{space } P \implies 0 \leq Pxyz (x, y, z)$
assumes $I1$: *integrable* $(S \otimes_M T \otimes_M P) (\lambda(x, y, z). Pxyz (x, y, z) * \log b (Pxyz (x, y, z) / (Px x * Pyz (y, z))))$
assumes $I2$: *integrable* $(S \otimes_M T \otimes_M P) (\lambda(x, y, z). Pxyz (x, y, z) * \log b (Pxz (x, z) / (Px x * Pz z)))$
shows *conditional-mutual-information-generic-eq*: *conditional-mutual-information* $b S T P X Y Z$
 $= (\int (x, y, z). Pxyz (x, y, z) * \log b (Pxyz (x, y, z) / (Pxz (x, z) * (Pyz (y, z) / Pz z))) \partial(S \otimes_M T \otimes_M P)) \text{ (is ?eq)}$
and *conditional-mutual-information-generic-nonneg*: $0 \leq \text{conditional-mutual-information } b S T P X Y Z \text{ (is ?nonneg)}$
 $\langle \text{proof} \rangle$

lemma (in *information-space*)
fixes $Px :: - \Rightarrow \text{real}$
assumes S : *sigma-finite-measure* S **and** T : *sigma-finite-measure* T **and** P :
sigma-finite-measure P
assumes Fx : *finite-entropy* S X Px
assumes Fz : *finite-entropy* P Z Pz
assumes Fyz : *finite-entropy* $(T \otimes_M P)$ $(\lambda x. (Y\ x, Z\ x))$ Pyz
assumes Fxz : *finite-entropy* $(S \otimes_M P)$ $(\lambda x. (X\ x, Z\ x))$ Pxz
assumes $Fxyz$: *finite-entropy* $(S \otimes_M T \otimes_M P)$ $(\lambda x. (X\ x, Y\ x, Z\ x))$ $Pxyz$
shows *conditional-mutual-information-generic-eq'*: *conditional-mutual-information*
 $b\ S\ T\ P\ X\ Y\ Z$
 $= (\int (x, y, z). Pxyz\ (x, y, z) * \log b\ (Pxyz\ (x, y, z) / (Pxz\ (x, z) * (Pyz\ (y, z) / Pz\ z))) \partial(S \otimes_M T \otimes_M P))$ (is ?eq)
and *conditional-mutual-information-generic-nonneg'*: $0 \leq \text{conditional-mutual-information}$
 $b\ S\ T\ P\ X\ Y\ Z$ (is ?nonneg)
 <proof>

lemma (in *information-space*) *conditional-mutual-information-eq*:
assumes Pz : *simple-distributed* M Z Pz
assumes Pyz : *simple-distributed* M $(\lambda x. (Y\ x, Z\ x))$ Pyz
assumes Pxz : *simple-distributed* M $(\lambda x. (X\ x, Z\ x))$ Pxz
assumes $Pxyz$: *simple-distributed* M $(\lambda x. (X\ x, Y\ x, Z\ x))$ $Pxyz$
shows $\mathcal{I}(X ; Y \mid Z) =$
 $(\sum (x, y, z) \in (\lambda x. (X\ x, Y\ x, Z\ x))' \text{space } M. Pxyz\ (x, y, z) * \log b\ (Pxyz\ (x, y, z) / (Pxz\ (x, z) * (Pyz\ (y, z) / Pz\ z))))$
 <proof>

lemma (in *information-space*) *conditional-mutual-information-nonneg*:
assumes X : *simple-function* M X **and** Y : *simple-function* M Y **and** Z : *simple-function* M Z
shows $0 \leq \mathcal{I}(X ; Y \mid Z)$
 <proof>

12.7 Conditional Entropy

definition (in *prob-space*)
conditional-entropy $b\ S\ T\ X\ Y = - (\int (x, y). \log b\ (\text{enn2real}\ (RN\text{-deriv}\ (S \otimes_M T)\ (\text{distr}\ M\ (S \otimes_M T)\ (\lambda x. (X\ x, Y\ x))))\ (x, y)) / \text{enn2real}\ (RN\text{-deriv}\ T\ (\text{distr}\ M\ T\ Y)\ y)) \partial \text{distr}\ M\ (S \otimes_M T)\ (\lambda x. (X\ x, Y\ x)))$

abbreviation (in *information-space*)
conditional-entropy-Pow $(\langle \mathcal{H}'(- \mid -) \rangle)$ **where**
 $\mathcal{H}(X \mid Y) \equiv \text{conditional-entropy}\ b\ (\text{count-space}\ (X' \text{space } M))\ (\text{count-space}\ (Y' \text{space } M))\ X\ Y$

lemma (in *information-space*) *conditional-entropy-generic-eq*:
fixes $Pxy :: - \Rightarrow \text{real}$ **and** $Py :: 'c \Rightarrow \text{real}$
assumes S : *sigma-finite-measure* S **and** T : *sigma-finite-measure* T

assumes $Py[measurable]$: distributed $M\ T\ Y\ Py$ **and** $Py-nn[simp]$: $\bigwedge x. x \in \text{space } T \implies 0 \leq Py\ x$
assumes $Pxy[measurable]$: distributed $M\ (S \otimes_M T)\ (\lambda x. (X\ x, Y\ x))\ Pxy$
and $Pxy-nn[simp]$: $\bigwedge x\ y. x \in \text{space } S \implies y \in \text{space } T \implies 0 \leq Pxy\ (x, y)$
shows conditional-entropy $b\ S\ T\ X\ Y = - (\int (x, y). Pxy\ (x, y) * \log b\ (Pxy\ (x, y) / Py\ y))\ \partial(S \otimes_M T)$
 $\langle proof \rangle$

lemma (in *information-space*) conditional-entropy-eq-entropy:

fixes $Px :: 'b \Rightarrow \text{real}$ **and** $Py :: 'c \Rightarrow \text{real}$
assumes S : sigma-finite-measure S **and** T : sigma-finite-measure T
assumes $Py[measurable]$: distributed $M\ T\ Y\ Py$
and $Py-nn[simp]$: $\bigwedge x. x \in \text{space } T \implies 0 \leq Py\ x$
assumes $Pxy[measurable]$: distributed $M\ (S \otimes_M T)\ (\lambda x. (X\ x, Y\ x))\ Pxy$
and $Pxy-nn[simp]$: $\bigwedge x\ y. x \in \text{space } S \implies y \in \text{space } T \implies 0 \leq Pxy\ (x, y)$
assumes $I1$: integrable $(S \otimes_M T)\ (\lambda x. Pxy\ x * \log b\ (Pxy\ x))$
assumes $I2$: integrable $(S \otimes_M T)\ (\lambda x. Pxy\ x * \log b\ (Py\ (snd\ x)))$
shows conditional-entropy $b\ S\ T\ X\ Y = \text{entropy } b\ (S \otimes_M T)\ (\lambda x. (X\ x, Y\ x))$
 $- \text{entropy } b\ T\ Y$
 $\langle proof \rangle$

lemma (in *information-space*) conditional-entropy-eq-entropy-simple:

assumes X : simple-function $M\ X$ **and** Y : simple-function $M\ Y$
shows $\mathcal{H}(X \mid Y) = \text{entropy } b\ (\text{count-space } (X'\text{space } M) \otimes_M \text{count-space } (Y'\text{space } M))\ (\lambda x. (X\ x, Y\ x)) - \mathcal{H}(Y)$
 $\langle proof \rangle$

lemma (in *information-space*) conditional-entropy-eq:

assumes Y : simple-distributed $M\ Y\ Py$
assumes XY : simple-distributed $M\ (\lambda x. (X\ x, Y\ x))\ Pxy$
shows $\mathcal{H}(X \mid Y) = - (\sum (x, y) \in (\lambda x. (X\ x, Y\ x))\ ' \text{space } M. Pxy\ (x, y) * \log b\ (Pxy\ (x, y) / Py\ y))$
 $\langle proof \rangle$

lemma (in *information-space*) conditional-mutual-information-eq-conditional-entropy:

assumes X : simple-function $M\ X$ **and** Y : simple-function $M\ Y$
shows $\mathcal{I}(X ; X \mid Y) = \mathcal{H}(X \mid Y)$
 $\langle proof \rangle$

lemma (in *information-space*) conditional-entropy-nonneg:

assumes X : simple-function $M\ X$ **and** Y : simple-function $M\ Y$ **shows** $0 \leq \mathcal{H}(X \mid Y)$
 $\langle proof \rangle$

12.8 Equalities

lemma (in *information-space*) mutual-information-eq-entropy-conditional-entropy-distr:

fixes $Px :: 'b \Rightarrow \text{real}$ **and** $Py :: 'c \Rightarrow \text{real}$ **and** $Pxy :: ('b \times 'c) \Rightarrow \text{real}$
assumes S : sigma-finite-measure S **and** T : sigma-finite-measure T

assumes Px [measurable]: distributed $M\ S\ X\ Px$
and Px -nn[simp]: $\bigwedge x. x \in \text{space } S \implies 0 \leq Px\ x$
and P_y [measurable]: distributed $M\ T\ Y\ P_y$
and P_y -nn[simp]: $\bigwedge x. x \in \text{space } T \implies 0 \leq P_y\ x$
and P_{xy} [measurable]: distributed $M\ (S \otimes_M T)\ (\lambda x. (X\ x, Y\ x))\ P_{xy}$
and P_{xy} -nn[simp]: $\bigwedge x\ y. x \in \text{space } S \implies y \in \text{space } T \implies 0 \leq P_{xy}\ (x, y)$
assumes I_x : integrable($S \otimes_M T$) ($\lambda x. P_{xy}\ x * \log b\ (Px\ (\text{fst } x))$)
assumes I_y : integrable($S \otimes_M T$) ($\lambda x. P_{xy}\ x * \log b\ (P_y\ (\text{snd } x))$)
assumes I_{xy} : integrable($S \otimes_M T$) ($\lambda x. P_{xy}\ x * \log b\ (P_{xy}\ x)$)
shows $\text{mutual-information } b\ S\ T\ X\ Y = \text{entropy } b\ S\ X + \text{entropy } b\ T\ Y -$
 $\text{entropy } b\ (S \otimes_M T)\ (\lambda x. (X\ x, Y\ x))$
 <proof>

lemma (in information-space) *mutual-information-eq-entropy-conditional-entropy'*:
fixes $Px :: 'b \Rightarrow \text{real}$ **and** $P_y :: 'c \Rightarrow \text{real}$ **and** $P_{xy} :: ('b \times 'c) \Rightarrow \text{real}$
assumes S : sigma-finite-measure S **and** T : sigma-finite-measure T
assumes Px : distributed $M\ S\ X\ Px$ $\bigwedge x. x \in \text{space } S \implies 0 \leq Px\ x$
and P_y : distributed $M\ T\ Y\ P_y$ $\bigwedge x. x \in \text{space } T \implies 0 \leq P_y\ x$
assumes P_{xy} : distributed $M\ (S \otimes_M T)\ (\lambda x. (X\ x, Y\ x))\ P_{xy}$
 $\bigwedge x. x \in \text{space } (S \otimes_M T) \implies 0 \leq P_{xy}\ x$
assumes I_x : integrable($S \otimes_M T$) ($\lambda x. P_{xy}\ x * \log b\ (Px\ (\text{fst } x))$)
assumes I_y : integrable($S \otimes_M T$) ($\lambda x. P_{xy}\ x * \log b\ (P_y\ (\text{snd } x))$)
assumes I_{xy} : integrable($S \otimes_M T$) ($\lambda x. P_{xy}\ x * \log b\ (P_{xy}\ x)$)
shows $\text{mutual-information } b\ S\ T\ X\ Y = \text{entropy } b\ S\ X - \text{conditional-entropy } b$
 $S\ T\ X\ Y$
 <proof>

lemma (in information-space) *mutual-information-eq-entropy-conditional-entropy*:
assumes $\text{sf-}X$: simple-function $M\ X$ **and** $\text{sf-}Y$: simple-function $M\ Y$
shows $\mathcal{I}(X ; Y) = \mathcal{H}(X) - \mathcal{H}(X \mid Y)$
 <proof>

lemma (in information-space) *mutual-information-nonneg-simple*:
assumes $\text{sf-}X$: simple-function $M\ X$ **and** $\text{sf-}Y$: simple-function $M\ Y$
shows $0 \leq \mathcal{I}(X ; Y)$
 <proof>

lemma (in information-space) *conditional-entropy-less-eq-entropy*:
assumes X : simple-function $M\ X$ **and** Z : simple-function $M\ Z$
shows $\mathcal{H}(X \mid Z) \leq \mathcal{H}(X)$
 <proof>

lemma (in information-space)
fixes $Px :: 'b \Rightarrow \text{real}$ **and** $P_y :: 'c \Rightarrow \text{real}$ **and** $P_{xy} :: ('b \times 'c) \Rightarrow \text{real}$
assumes S : sigma-finite-measure S **and** T : sigma-finite-measure T
assumes Px : finite-entropy $S\ X\ Px$ **and** P_y : finite-entropy $T\ Y\ P_y$
assumes P_{xy} : finite-entropy $(S \otimes_M T)\ (\lambda x. (X\ x, Y\ x))\ P_{xy}$
shows $\text{conditional-entropy } b\ S\ T\ X\ Y \leq \text{entropy } b\ S\ X$
 <proof>

lemma (in *information-space*) *entropy-chain-rule*:
assumes X : *simple-function* M X **and** Y : *simple-function* M Y
shows $\mathcal{H}(\lambda x. (X\ x, Y\ x)) = \mathcal{H}(X) + \mathcal{H}(Y|X)$
 ⟨*proof*⟩

lemma (in *information-space*) *entropy-partition*:
assumes X : *simple-function* M X
shows $\mathcal{H}(X) = \mathcal{H}(f \circ X) + \mathcal{H}(X|f \circ X)$
 ⟨*proof*⟩

corollary (in *information-space*) *entropy-data-processing*:
assumes *simple-function* M X **shows** $\mathcal{H}(f \circ X) \leq \mathcal{H}(X)$
 ⟨*proof*⟩

corollary (in *information-space*) *entropy-of-inj*:
assumes X : *simple-function* M X **and** inj : *inj-on* f (X 's space M)
shows $\mathcal{H}(f \circ X) = \mathcal{H}(X)$
 ⟨*proof*⟩

end

13 Properties of Various Distributions

theory *Distributions*
imports *Convolution Information*
begin

lemma (in *prob-space*) *distributed-affine*:
fixes $f :: \text{real} \Rightarrow \text{ennreal}$
assumes f : *distributed* M *lborel* X f
assumes c : $c \neq 0$
shows *distributed* M *lborel* $(\lambda x. t + c * X\ x) (\lambda x. f\ ((x - t) / c) / |c|)$
 ⟨*proof*⟩

lemma (in *prob-space*) *distributed-affineI*:
fixes $f :: \text{real} \Rightarrow \text{ennreal}$ **and** $c :: \text{real}$
assumes f : *distributed* M *lborel* $(\lambda x. (X\ x - t) / c) (\lambda x. |c| * f\ (x * c + t))$
assumes c : $c \neq 0$
shows *distributed* M *lborel* X f
 ⟨*proof*⟩

lemma (in *prob-space*) *distributed-AE2*:
assumes [*measurable*]: *distributed* M N X f *Measurable.pred* N P
shows $(AE\ x\ in\ M. P\ (X\ x)) \longleftrightarrow (AE\ x\ in\ N. 0 < f\ x \longrightarrow P\ x)$
 ⟨*proof*⟩

13.1 Erlang

lemma *nn-integral-power-times-exp-Icc*:

assumes $[arith]: 0 \leq a$

shows $(\int^+ x. ennreal (x^k * exp (-x)) * indicator \{0 .. a\} x \partial lborel) =$
 $(1 - (\sum_{n \leq k}. (a^n * exp (-a)) / fact n)) * fact k$ **(is ?I = -)**

$\langle proof \rangle$

lemma *nn-integral-power-times-exp-Ici*:

shows $(\int^+ x. ennreal (x^k * exp (-x)) * indicator \{0 ..\} x \partial lborel) = real-of-nat$
 $(fact k)$

$\langle proof \rangle$

definition *erlang-density* :: $nat \Rightarrow real \Rightarrow real \Rightarrow real$ **where**

erlang-density $k \ l \ x = (if \ x < 0 \ then \ 0 \ else \ (l^{Suc \ k} * x^k * exp (-l * x)) /$
 $fact \ k)$

definition *erlang-CDF* :: $nat \Rightarrow real \Rightarrow real \Rightarrow real$ **where**

erlang-CDF $k \ l \ x = (if \ x < 0 \ then \ 0 \ else \ 1 - (\sum_{n \leq k}. ((l * x)^n * exp (-l * x) / fact \ n)))$

lemma *erlang-density-nonneg[simp]*: $0 \leq l \implies 0 \leq erlang-density \ k \ l \ x$

$\langle proof \rangle$

lemma *borel-measurable-erlang-density[measurable]*: *erlang-density* $k \ l \in borel-measurable$
 $borel$

$\langle proof \rangle$

lemma *erlang-CDF-transform*: $0 < l \implies erlang-CDF \ k \ l \ a = erlang-CDF \ k \ 1 \ (l$
 $* \ a)$

$\langle proof \rangle$

lemma *erlang-CDF-nonneg[simp]*: **assumes** $0 < l$ **shows** $0 \leq erlang-CDF \ k \ l \ x$

$\langle proof \rangle$

lemma *nn-integral-erlang-density*:

assumes $[arith]: 0 < l$

shows $(\int^+ x. ennreal (erlang-density \ k \ l \ x) * indicator \{.. \ a\} x \partial lborel) =$
 $erlang-CDF \ k \ l \ a$

$\langle proof \rangle$

lemma *emeasure-erlang-density*:

$0 < l \implies emeasure \ (density \ lborel \ (erlang-density \ k \ l)) \ \{.. \ a\} = erlang-CDF \ k \ l$
 a

$\langle proof \rangle$

lemma *nn-integral-erlang-ith-moment*:

fixes $k \ i :: nat$ **and** $l :: real$

assumes $[arith]: 0 < l$

shows $(\int^+ x. ennreal (erlang-density \ k \ l \ x * x^i) \partial lborel) = fact \ (k + i) /$

(fact $k * l \wedge i$)
 <proof>

lemma *prob-space-erlang-density*:

assumes $l[\text{arith}]$: $0 < l$
 shows *prob-space* (density lborel (erlang-density $k\ l$)) (is prob-space ?D)
 <proof>

lemma (in *prob-space*) *erlang-distributed-le*:

assumes D : distributed M lborel X (erlang-density $k\ l$)
 assumes $[simp, \text{arith}]$: $0 < l\ 0 \leq a$
 shows $\mathcal{P}(x \text{ in } M. X\ x \leq a) = \text{erlang-CDF } k\ l\ a$
 <proof>

lemma (in *prob-space*) *erlang-distributed-gt*:

assumes $D[simp]$: distributed M lborel X (erlang-density $k\ l$)
 assumes $[\text{arith}]$: $0 < l\ 0 \leq a$
 shows $\mathcal{P}(x \text{ in } M. a < X\ x) = 1 - (\text{erlang-CDF } k\ l\ a)$
 <proof>

lemma *erlang-CDF-at0*: $\text{erlang-CDF } k\ l\ 0 = 0$

<proof>

lemma *erlang-distributedI*:

assumes $X[\text{measurable}]$: $X \in \text{borel-measurable } M$ and $[\text{arith}]$: $0 < l$
 and $X\text{-distr}$: $\bigwedge a. 0 \leq a \implies \text{emeasure } M \{x \in \text{space } M. X\ x \leq a\} = \text{erlang-CDF } k\ l\ a$
 shows distributed M lborel X (erlang-density $k\ l$)
 <proof>

lemma (in *prob-space*) *erlang-distributed-iff*:

assumes $[\text{arith}]$: $0 < l$
 shows distributed M lborel X (erlang-density $k\ l$) \longleftrightarrow
 $(X \in \text{borel-measurable } M \wedge 0 < l \wedge (\forall a \geq 0. \mathcal{P}(x \text{ in } M. X\ x \leq a) = \text{erlang-CDF } k\ l\ a))$
 <proof>

lemma (in *prob-space*) *erlang-distributed-mult-const*:

assumes erlX : distributed M lborel X (erlang-density $k\ l$)
 assumes $a\text{-pos}[\text{arith}]$: $0 < \alpha\ 0 < l$
 shows distributed M lborel $(\lambda x. \alpha * X\ x)$ (erlang-density $k\ (l / \alpha)$)
 <proof>

lemma (in *prob-space*) *has-bochner-integral-erlang-ith-moment*:

fixes $k\ i :: \text{nat}$ and $l :: \text{real}$
 assumes $[\text{arith}]$: $0 < l$ and D : distributed M lborel X (erlang-density $k\ l$)
 shows has-bochner-integral M $(\lambda x. X\ x \wedge i)$ (fact $(k + i) / (\text{fact } k * l \wedge i)$)
 <proof>

lemma (in prob-space) erlang-ith-moment-integrable:

$0 < l \implies \text{distributed } M \text{ lborel } X \text{ (erlang-density } k \text{ } l) \implies \text{integrable } M \text{ } (\lambda x. X x \wedge i)$
 ⟨proof⟩

lemma (in prob-space) erlang-ith-moment:

$0 < l \implies \text{distributed } M \text{ lborel } X \text{ (erlang-density } k \text{ } l) \implies$
 $\text{expectation } (\lambda x. X x \wedge i) = \text{fact } (k + i) / (\text{fact } k * l \wedge i)$
 ⟨proof⟩

lemma (in prob-space) erlang-distributed-variance:

assumes [arith]: $0 < l$ **and** distributed M lborel X (erlang-density k l)
shows variance $X = (k + 1) / l^2$
 ⟨proof⟩

13.2 Exponential distribution

abbreviation exponential-density :: real \Rightarrow real \Rightarrow real **where**
 exponential-density \equiv erlang-density 0

lemma exponential-density-def:

exponential-density l $x = (\text{if } x < 0 \text{ then } 0 \text{ else } l * \exp (- x * l))$
 ⟨proof⟩

lemma erlang-CDF-0: erlang-CDF 0 l $a = (\text{if } 0 \leq a \text{ then } 1 - \exp (- l * a) \text{ else } 0)$
 ⟨proof⟩

lemma prob-space-exponential-density: $0 < l \implies \text{prob-space (density lborel (exponential-density } l))$
 ⟨proof⟩

lemma (in prob-space) exponential-distributedD-le:

assumes D : distributed M lborel X (exponential-density l) **and** a : $0 \leq a$ **and** l :
 $0 < l$
shows $\mathcal{P}(x \text{ in } M. X x \leq a) = 1 - \exp (- a * l)$
 ⟨proof⟩

lemma (in prob-space) exponential-distributedD-gt:

assumes D : distributed M lborel X (exponential-density l) **and** a : $0 \leq a$ **and** l :
 $0 < l$
shows $\mathcal{P}(x \text{ in } M. a < X x) = \exp (- a * l)$
 ⟨proof⟩

lemma (in prob-space) exponential-distributed-memoryless:

assumes D : distributed M lborel X (exponential-density l) **and** a : $0 \leq a$ **and** l :
 $0 < l$ **and** t : $0 \leq t$
shows $\mathcal{P}(x \text{ in } M. a + t < X x \mid a < X x) = \mathcal{P}(x \text{ in } M. t < X x)$
 ⟨proof⟩

lemma *exponential-distributedI*:

assumes $X[\text{measurable}]$: $X \in \text{borel-measurable } M$ **and** $[\text{arith}]$: $0 < l$
and $X\text{-distr}$: $\bigwedge a. 0 \leq a \implies \text{emeasure } M \{x \in \text{space } M. X x \leq a\} = 1 - \exp(-a * l)$
shows $\text{distributed } M \text{ lborel } X \text{ (exponential-density } l)$
 $\langle \text{proof} \rangle$

lemma (*in prob-space*) *exponential-distributed-iff*:

assumes $0 < l$
shows $\text{distributed } M \text{ lborel } X \text{ (exponential-density } l) \longleftrightarrow$
 $(X \in \text{borel-measurable } M \wedge (\forall a \geq 0. \mathcal{P}(x \text{ in } M. X x \leq a) = 1 - \exp(-a * l)))$
 $\langle \text{proof} \rangle$

lemma (*in prob-space*) *exponential-distributed-expectation*:

$0 < l \implies \text{distributed } M \text{ lborel } X \text{ (exponential-density } l) \implies \text{expectation } X = 1 / l$
 $\langle \text{proof} \rangle$

lemma *exponential-density-nonneg*: $0 < l \implies 0 \leq \text{exponential-density } l x$

$\langle \text{proof} \rangle$

lemma (*in prob-space*) *exponential-distributed-min*:

assumes $0 < l \ 0 < u$
assumes $\text{exp}X$: $\text{distributed } M \text{ lborel } X \text{ (exponential-density } l)$
assumes $\text{exp}Y$: $\text{distributed } M \text{ lborel } Y \text{ (exponential-density } u)$
assumes ind : $\text{indep-var borel } X \text{ borel } Y$
shows $\text{distributed } M \text{ lborel } (\lambda x. \min (X x) (Y x)) \text{ (exponential-density } (l + u))$
 $\langle \text{proof} \rangle$

lemma (*in prob-space*) *exponential-distributed-Min*:

assumes $\text{fin}I$: $\text{finite } I$
assumes A : $I \neq \{\}$
assumes l : $\bigwedge i. i \in I \implies 0 < l i$
assumes $\text{exp}X$: $\bigwedge i. i \in I \implies \text{distributed } M \text{ lborel } (X i) \text{ (exponential-density } (l i))$
assumes ind : $\text{indep-vars } (\lambda i. \text{borel}) X I$
shows $\text{distributed } M \text{ lborel } (\lambda x. \text{Min } ((\lambda i. X i x) 'I)) \text{ (exponential-density } (\sum_{i \in I. l i}))$
 $\langle \text{proof} \rangle$

lemma (*in prob-space*) *exponential-distributed-variance*:

$0 < l \implies \text{distributed } M \text{ lborel } X \text{ (exponential-density } l) \implies \text{variance } X = 1 / l^2$
 $\langle \text{proof} \rangle$

lemma *nn-integral-zero'*: $A E x \text{ in } M. f x = 0 \implies (\int^+ x. f x \partial M) = 0$

$\langle \text{proof} \rangle$

lemma *convolution-erlang-density*:

fixes $k_1\ k_2 :: \text{nat}$
assumes $[simp, arith]: 0 < l$
shows $(\lambda x. \int^+ y. \text{ennreal} (\text{erlang-density } k_1\ l\ (x - y)) * \text{ennreal} (\text{erlang-density } k_2\ l\ y) \ \partial \text{lborel}) =$
 $(\text{erlang-density } (\text{Suc } k_1 + \text{Suc } k_2 - 1)\ l)$
(is ?LHS = ?RHS)
 $\langle \text{proof} \rangle$

lemma *(in prob-space) sum-indep-erlang*:

assumes *indep*: *indep-var borel* $X\ \text{borel } Y$
assumes $[simp, arith]: 0 < l$
assumes *erlX*: *distributed M lborel* $X\ (\text{erlang-density } k_1\ l)$
assumes *erlY*: *distributed M lborel* $Y\ (\text{erlang-density } k_2\ l)$
shows *distributed M lborel* $(\lambda x. X\ x + Y\ x)\ (\text{erlang-density } (\text{Suc } k_1 + \text{Suc } k_2 - 1)\ l)$
 $\langle \text{proof} \rangle$

lemma *(in prob-space) erlang-distributed-sum*:

assumes *finI*: *finite* I
assumes $A: I \neq \{\}$
assumes $[simp, arith]: 0 < l$
assumes *expX*: $\bigwedge i. i \in I \implies \text{distributed } M\ \text{lborel } (X\ i)\ (\text{erlang-density } (k\ i)\ l)$
assumes *ind*: *indep-vars* $(\lambda i. \text{borel})\ X\ I$
shows *distributed M lborel* $(\lambda x. \sum_{i \in I}. X\ i\ x)\ (\text{erlang-density } ((\sum_{i \in I}. \text{Suc } (k\ i)) - 1)\ l)$
 $\langle \text{proof} \rangle$

lemma *(in prob-space) exponential-distributed-sum*:

assumes *finI*: *finite* I
assumes $A: I \neq \{\}$
assumes $l: 0 < l$
assumes *expX*: $\bigwedge i. i \in I \implies \text{distributed } M\ \text{lborel } (X\ i)\ (\text{exponential-density } l)$
assumes *ind*: *indep-vars* $(\lambda i. \text{borel})\ X\ I$
shows *distributed M lborel* $(\lambda x. \sum_{i \in I}. X\ i\ x)\ (\text{erlang-density } ((\text{card } I) - 1)\ l)$
 $\langle \text{proof} \rangle$

lemma *(in information-space) entropy-exponential*:

assumes $l[simp, arith]: 0 < l$
assumes *D*: *distributed M lborel* $X\ (\text{exponential-density } l)$
shows *entropy b lborel* $X = \log b\ (\exp 1 / l)$
 $\langle \text{proof} \rangle$

13.3 Uniform distribution

lemma *uniform-distrI*:

assumes *X*: $X \in \text{measurable } M\ M'$
and *A*: $A \in \text{sets } M'\ \text{emeasure } M'\ A \neq \infty\ \text{emeasure } M'\ A \neq 0$
assumes *distr*: $\bigwedge B. B \in \text{sets } M' \implies \text{emeasure } M\ (X - ' B \cap \text{space } M) =$

emeasure $M' (A \cap B) / \text{emeasure } M' A$
shows *distr* $M M' X = \text{uniform-measure } M' A$
 ⟨*proof*⟩

lemma *uniform-distrI-borel*:

fixes $A :: \text{real set}$
assumes $X[\text{measurable}]$: $X \in \text{borel-measurable } M$ **and** A : *emeasure lborel* $A = \text{ennreal } r$ $0 < r$
and $[\text{measurable}]$: $A \in \text{sets borel}$
assumes *distr*: $\bigwedge a. \text{emeasure } M \{x \in \text{space } M. X x \leq a\} = \text{emeasure lborel } (A \cap \{.. a\}) / r$
shows *distributed* $M \text{ lborel } X (\lambda x. \text{indicator } A x / \text{measure lborel } A)$
 ⟨*proof*⟩

lemma (*in prob-space*) *uniform-distrI-borel-atLeastAtMost*:

fixes $a b :: \text{real}$
assumes X : $X \in \text{borel-measurable } M$ **and** $a < b$
assumes *distr*: $\bigwedge t. a \leq t \implies t \leq b \implies \mathcal{P}(x \text{ in } M. X x \leq t) = (t - a) / (b - a)$
shows *distributed* $M \text{ lborel } X (\lambda x. \text{indicator } \{a..b\} x / \text{measure lborel } \{a..b\})$
 ⟨*proof*⟩

lemma (*in prob-space*) *uniform-distributed-measure*:

fixes $a b :: \text{real}$
assumes D : *distributed* $M \text{ lborel } X (\lambda x. \text{indicator } \{a .. b\} x / \text{measure lborel } \{a .. b\})$
assumes t : $a \leq t \leq b$
shows $\mathcal{P}(x \text{ in } M. X x \leq t) = (t - a) / (b - a)$
 ⟨*proof*⟩

lemma (*in prob-space*) *uniform-distributed-bounds*:

fixes $a b :: \text{real}$
assumes D : *distributed* $M \text{ lborel } X (\lambda x. \text{indicator } \{a .. b\} x / \text{measure lborel } \{a .. b\})$
shows $a < b$
 ⟨*proof*⟩

lemma (*in prob-space*) *uniform-distributed-iff*:

fixes $a b :: \text{real}$
shows *distributed* $M \text{ lborel } X (\lambda x. \text{indicator } \{a..b\} x / \text{measure lborel } \{a..b\}) \longleftrightarrow (X \in \text{borel-measurable } M \wedge a < b \wedge (\forall t \in \{a .. b\}. \mathcal{P}(x \text{ in } M. X x \leq t) = (t - a) / (b - a)))$
 ⟨*proof*⟩

lemma (*in prob-space*) *uniform-distributed-expectation*:

fixes $a b :: \text{real}$
assumes D : *distributed* $M \text{ lborel } X (\lambda x. \text{indicator } \{a .. b\} x / \text{measure lborel } \{a .. b\})$
shows *expectation* $X = (a + b) / 2$

<proof>

lemma (in prob-space) uniform-distributed-variance:

fixes $a\ b :: \text{real}$

assumes D : distributed M lborel X ($\lambda x.$ indicator $\{a .. b\}$ x / measure lborel $\{a .. b\}$)

shows variance $X = (b - a)^2 / 12$

<proof>

13.4 Normal distribution

definition normal-density :: $\text{real} \Rightarrow \text{real} \Rightarrow \text{real} \Rightarrow \text{real}$ **where**

normal-density $\mu\ \sigma\ x = 1 / \text{sqrt}\ (2 * \pi * \sigma^2) * \exp\ (-(x - \mu)^2 / (2 * \sigma^2))$

abbreviation std-normal-density :: $\text{real} \Rightarrow \text{real}$ **where**

std-normal-density \equiv normal-density $0\ 1$

lemma std-normal-density-def: std-normal-density $x = (1 / \text{sqrt}\ (2 * \pi)) * \exp\ (-x^2 / 2)$

<proof>

lemma normal-density-nonneg[simp]: $0 \leq \text{normal-density}\ \mu\ \sigma\ x$

<proof>

lemma normal-density-pos: $0 < \sigma \implies 0 < \text{normal-density}\ \mu\ \sigma\ x$

<proof>

lemma borel-measurable-normal-density[measurable]: normal-density $\mu\ \sigma \in \text{borel-measurable borel}$

<proof>

lemma gaussian-moment-0:

has-bochner-integral lborel ($\lambda x.$ indicator $\{0..\}$ $x *_{\mathbb{R}} \exp\ (-x^2)$) ($\text{sqrt}\ \pi / 2$)

<proof>

lemma gaussian-moment-1:

has-bochner-integral lborel ($\lambda x::\text{real}.$ indicator $\{0..\}$ $x *_{\mathbb{R}} (\exp\ (-x^2) * x)$) ($1 / 2$)

<proof>

lemma

fixes $k :: \text{nat}$

shows gaussian-moment-even-pos:

has-bochner-integral lborel ($\lambda x::\text{real}.$ indicator $\{0..\}$ $x *_{\mathbb{R}} (\exp\ (-x^2) * x^{\wedge}(2 * k))$)

(($\text{sqrt}\ \pi / 2$) * ($\text{fact}\ (2 * k) / (2^{\wedge}(2 * k) * \text{fact}\ k)$))

(is ?even)

and gaussian-moment-odd-pos:

has-bochner-integral lborel ($\lambda x::\text{real}.$ indicator $\{0..\}$ $x *_{\mathbb{R}} (\exp\ (-x^2) * x^{\wedge}(2 * k)$

$k + 1)))$ (*fact* $k / 2$)
 (**is** *?odd*)
 \langle *proof* \rangle

context

fixes $k :: \text{nat}$ **and** $\mu \sigma :: \text{real}$ **assumes** [*arith*]: $0 < \sigma$
begin

lemma *normal-moment-even*:

has-bochner-integral lborel ($\lambda x. \text{normal-density } \mu \sigma x * (x - \mu) ^{(2 * k)}$) (*fact*
 $(2 * k) / ((2 / \sigma^2) ^k * \text{fact } k)$)
 \langle *proof* \rangle

lemma *normal-moment-abs-odd*:

has-bochner-integral lborel ($\lambda x. \text{normal-density } \mu \sigma x * |x - \mu| ^{(2 * k + 1)}$) ($2 ^k$
 $* \sigma ^{(2 * k + 1) * \text{fact } k * \text{sqrt } (2 / \text{pi})}$)
 \langle *proof* \rangle

lemma *normal-moment-odd*:

has-bochner-integral lborel ($\lambda x. \text{normal-density } \mu \sigma x * (x - \mu) ^{(2 * k + 1)}$) 0
 \langle *proof* \rangle

lemma *integral-normal-moment-even*:

integral^L lborel ($\lambda x. \text{normal-density } \mu \sigma x * (x - \mu) ^{(2 * k)}$) = *fact* $(2 * k) /$
 $((2 / \sigma^2) ^k * \text{fact } k)$
 \langle *proof* \rangle

lemma *integral-normal-moment-abs-odd*:

integral^L lborel ($\lambda x. \text{normal-density } \mu \sigma x * |x - \mu| ^{(2 * k + 1)}$) = $2 ^k * \sigma$
 $^{(2 * k + 1) * \text{fact } k * \text{sqrt } (2 / \text{pi})}$
 \langle *proof* \rangle

lemma *integral-normal-moment-odd*:

integral^L lborel ($\lambda x. \text{normal-density } \mu \sigma x * (x - \mu) ^{(2 * k + 1)}$) = 0
 \langle *proof* \rangle

end

context

fixes $\sigma :: \text{real}$
assumes $\sigma\text{-pos}$ [*arith*]: $0 < \sigma$
begin

lemma *normal-moment-nz-1*: *has-bochner-integral lborel* ($\lambda x. \text{normal-density } \mu \sigma$
 $x * x$) μ
 \langle *proof* \rangle

lemma *integral-normal-moment-nz-1*:

$\text{integral}^L \text{ lborel } (\lambda x. \text{ normal-density } \mu \sigma x * x) = \mu$
 $\langle \text{proof} \rangle$

lemma *integrable-normal-moment-nz-1*: $\text{integrable lborel } (\lambda x. \text{ normal-density } \mu \sigma x * x * x)$
 $\langle \text{proof} \rangle$

lemma *integrable-normal-moment*: $\text{integrable lborel } (\lambda x. \text{ normal-density } \mu \sigma x * (x - \mu) \wedge^k)$
 $\langle \text{proof} \rangle$

lemma *integrable-normal-moment-abs*: $\text{integrable lborel } (\lambda x. \text{ normal-density } \mu \sigma x * |x - \mu| \wedge^k)$
 $\langle \text{proof} \rangle$

lemma *integrable-normal-density[simp, intro]*: $\text{integrable lborel } (\text{normal-density } \mu \sigma)$
 $\langle \text{proof} \rangle$

lemma *integral-normal-density[simp]*: $(\int x. \text{ normal-density } \mu \sigma x \partial \text{lborel}) = 1$
 $\langle \text{proof} \rangle$

lemma *prob-space-normal-density*:
 $\text{prob-space } (\text{density lborel } (\text{normal-density } \mu \sigma))$
 $\langle \text{proof} \rangle$

end

context
fixes $k :: \text{nat}$
begin

lemma *std-normal-moment-even*:
 $\text{has-bochner-integral lborel } (\lambda x. \text{ std-normal-density } x * x \wedge (2 * k)) (\text{fact } (2 * k) / (2 \wedge^k * \text{fact } k))$
 $\langle \text{proof} \rangle$

lemma *std-normal-moment-abs-odd*:
 $\text{has-bochner-integral lborel } (\lambda x. \text{ std-normal-density } x * |x| \wedge (2 * k + 1)) (\text{sqrt } (2/\text{pi}) * 2 \wedge^k * \text{fact } k)$
 $\langle \text{proof} \rangle$

lemma *std-normal-moment-odd*:
 $\text{has-bochner-integral lborel } (\lambda x. \text{ std-normal-density } x * x \wedge (2 * k + 1)) 0$
 $\langle \text{proof} \rangle$

lemma *integral-std-normal-moment-even*:

$\text{integral}^L \text{ lborel } (\lambda x. \text{std-normal-density } x * x^{\wedge}(2 * k)) = \text{fact } (2 * k) / (2^{\wedge}k * \text{fact } k)$
 ⟨proof⟩

lemma *integral-std-normal-moment-abs-odd:*

$\text{integral}^L \text{ lborel } (\lambda x. \text{std-normal-density } x * |x|^{\wedge}(2 * k + 1)) = \text{sqrt } (2 / \text{pi}) * 2^{\wedge}k * \text{fact } k$
 ⟨proof⟩

lemma *integral-std-normal-moment-odd:*

$\text{integral}^L \text{ lborel } (\lambda x. \text{std-normal-density } x * x^{\wedge}(2 * k + 1)) = 0$
 ⟨proof⟩

lemma *integrable-std-normal-moment-abs: integrable lborel* $(\lambda x. \text{std-normal-density } x * |x|^{\wedge}k)$
 ⟨proof⟩

lemma *integrable-std-normal-moment: integrable lborel* $(\lambda x. \text{std-normal-density } x * x^{\wedge}k)$
 ⟨proof⟩

end

lemma (in *prob-space*) *normal-density-affine:*

assumes *X: distributed M lborel X* (normal-density $\mu \sigma$)

assumes [*simp, arith*]: $0 < \sigma \wedge \alpha \neq 0$

shows *distributed M lborel* $(\lambda x. \beta + \alpha * X x)$ (normal-density $(\beta + \alpha * \mu) (|\alpha| * \sigma)$)
 ⟨proof⟩

lemma (in *prob-space*) *normal-standard-normal-convert:*

assumes *pos-var* [*simp, arith*]: $0 < \sigma$

shows *distributed M lborel X* (normal-density $\mu \sigma$) = *distributed M lborel* $(\lambda x. (X x - \mu) / \sigma)$ *std-normal-density*
 ⟨proof⟩

lemma *conv-normal-density-zero-mean:*

assumes [*simp, arith*]: $0 < \sigma \wedge 0 < \tau$

shows $(\lambda x. \int^+ y. \text{ennreal } (\text{normal-density } 0 \sigma (x - y) * \text{normal-density } 0 \tau y) \partial \text{lborel}) =$
 $\text{normal-density } 0 (\text{sqrt } (\sigma^2 + \tau^2))$ (is ?LHS = ?RHS)
 ⟨proof⟩

lemma *conv-std-normal-density:*

$(\lambda x. \int^+ y. \text{ennreal } (\text{std-normal-density } (x - y) * \text{std-normal-density } y) \partial \text{lborel}) =$
 $\text{normal-density } 0 (\text{sqrt } 2)$
 ⟨proof⟩

lemma (in *prob-space*) *add-indep-normal*:
assumes *indep*: *indep-var* *borel* *X* *borel* *Y*
assumes *pos-var*[*arith*]: $0 < \sigma$ $0 < \tau$
assumes *normalX*[*simp*]: *distributed* *M* *lborel* *X* (*normal-density* μ σ)
assumes *normalY*[*simp*]: *distributed* *M* *lborel* *Y* (*normal-density* ν τ)
shows *distributed* *M* *lborel* $(\lambda x. X\ x + Y\ x)$ (*normal-density* $(\mu + \nu)$ $(\text{sqrt } (\sigma^2 + \tau^2))$)
 <proof>

lemma (in *prob-space*) *diff-indep-normal*:
assumes *indep*[*simp*]: *indep-var* *borel* *X* *borel* *Y*
assumes [*simp*, *arith*]: $0 < \sigma$ $0 < \tau$
assumes *normalX*[*simp*]: *distributed* *M* *lborel* *X* (*normal-density* μ σ)
assumes *normalY*[*simp*]: *distributed* *M* *lborel* *Y* (*normal-density* ν τ)
shows *distributed* *M* *lborel* $(\lambda x. X\ x - Y\ x)$ (*normal-density* $(\mu - \nu)$ $(\text{sqrt } (\sigma^2 + \tau^2))$)
 <proof>

lemma (in *prob-space*) *sum-indep-normal*:
assumes *finite* *I* $I \neq \{\}$ *indep-vars* $(\lambda i. \text{borel } X\ i)$
assumes $\bigwedge i. i \in I \implies 0 < \sigma\ i$
assumes *normal*: $\bigwedge i. i \in I \implies \text{distributed } M\ \text{lborel } (X\ i)$ (*normal-density* $(\mu\ i)$ $(\sigma\ i)$)
shows *distributed* *M* *lborel* $(\lambda x. \sum i \in I. X\ i\ x)$ (*normal-density* $(\sum i \in I. \mu\ i)$ $(\text{sqrt } (\sum i \in I. (\sigma\ i)^2))$)
 <proof>

lemma (in *prob-space*) *standard-normal-distributed-expectation*:
assumes *D*: *distributed* *M* *lborel* *X* *std-normal-density*
shows *expectation* *X* = 0
 <proof>

lemma (in *prob-space*) *normal-distributed-expectation*:
assumes σ [*arith*]: $0 < \sigma$
assumes *D*: *distributed* *M* *lborel* *X* (*normal-density* μ σ)
shows *expectation* *X* = μ
 <proof>

lemma (in *prob-space*) *normal-distributed-variance*:
fixes *a* *b* :: *real*
assumes [*simp*, *arith*]: $0 < \sigma$
assumes *D*: *distributed* *M* *lborel* *X* (*normal-density* μ σ)
shows *variance* *X* = σ^2
 <proof>

lemma (in *prob-space*) *standard-normal-distributed-variance*:
distributed *M* *lborel* *X* *std-normal-density* \implies *variance* *X* = 1
 <proof>

lemma (*in information-space*) *entropy-normal-density*:
assumes [*arith*]: $0 < \sigma$
assumes *D*: *distributed M lborel X (normal-density $\mu \sigma$)*
shows *entropy b lborel X = log b (2 * pi * exp 1 * σ^2) / 2*
 \langle *proof* \rangle
end

14 Characteristic Functions

theory *Characteristic-Functions*
imports *Weak-Convergence Independent-Family Distributions*
begin

lemma *mult-min-right*: $a \geq 0 \implies (a :: \text{real}) * \min b c = \min (a * b) (a * c)$
 \langle *proof* \rangle

lemma *sequentially-even-odd*:
assumes *E*: *eventually* $(\lambda n. P (2 * n))$ *sequentially* **and** *O*: *eventually* $(\lambda n. P (2 * n + 1))$ *sequentially*
shows *eventually P sequentially*
 \langle *proof* \rangle

lemma *limseq-even-odd*:
assumes $(\lambda n. f (2 * n)) \longrightarrow (l :: 'a :: \text{topological-space})$
and $(\lambda n. f (2 * n + 1)) \longrightarrow l$
shows $f \longrightarrow l$
 \langle *proof* \rangle

14.1 Application of the FTC: integrating $e^i x$

abbreviation *iexp* :: *real* \Rightarrow *complex* **where**
iexp $\equiv (\lambda x. \text{exp } (i * \text{complex-of-real } x))$

lemma *isCont-iexp [simp]*: *isCont iexp x*
 \langle *proof* \rangle

lemma *has-vector-derivative-iexp[derivative-intros]*:
*(iexp has-vector-derivative i * iexp x) (at x within s)*
 \langle *proof* \rangle

lemma *interval-integral-iexp*:
fixes *a b* :: *real*
shows $(\text{CLBINT } x=a..b. \text{iexp } x) = i * \text{iexp } a - i * \text{iexp } b$
 \langle *proof* \rangle

14.2 The Characteristic Function of a Real Measure.

definition

char :: *real measure* \Rightarrow *real* \Rightarrow *complex*
where *char* *M* *t* \equiv *CLINT* *x* | *M*. *iexp* (*t* * *x*)

lemma (*in real-distribution*) *char-zero*: *char* *M* 0 = 1
 <proof>

lemma (*in prob-space*) *integrable-iexp*:
assumes *f*: *f* \in *borel-measurable* *M* \wedge *x*. *Im* (*f* *x*) = 0
shows *integrable* *M* (λx . *exp* (*i* * (*f* *x*)))
 <proof>

lemma (*in real-distribution*) *cmmod-char-le-1*: *norm* (*char* *M* *t*) \leq 1
 <proof>

lemma (*in real-distribution*) *isCont-char*: *isCont* (*char* *M*) *t*
 <proof>

lemma (*in real-distribution*) *char-measurable* [*measurable*]: *char* *M* \in *borel-measurable*
borel
 <proof>

14.3 Independence

lemma (*in prob-space*) *char-distr-add*:
fixes *X1* *X2* :: 'a \Rightarrow *real* **and** *t* :: *real*
assumes *indep-var* *borel* *X1* *borel* *X2*
shows *char* (*distr* *M* *borel* ($\lambda \omega$. *X1* ω + *X2* ω)) *t* =
char (*distr* *M* *borel* *X1*) *t* * *char* (*distr* *M* *borel* *X2*) *t*
 <proof>

lemma (*in prob-space*) *char-distr-sum*:
indep-vars (λi . *borel*) *X* *A* \Rightarrow
char (*distr* *M* *borel* ($\lambda \omega$. $\sum_{i \in A} X\ i\ \omega$)) *t* = ($\prod_{i \in A}$ *char* (*distr* *M* *borel* (*X*
i)) *t*)
 <proof>

14.4 Approximations to e^{ix}

Proofs from Billingsley, page 343.

lemma *CLBINT-I0c-power-mirror-iexp*:
fixes *x* :: *real* **and** *n* :: *nat*
defines *f* *s* *m* \equiv *complex-of-real* ((*x* - *s*) \wedge *m*)
shows (*CLBINT* *s=0..x*. *f* *s* *n* * *iexp* *s*) =
x \wedge *Suc* *n* / *Suc* *n* + (*i* / *Suc* *n*) * (*CLBINT* *s=0..x*. *f* *s* (*Suc* *n*) * *iexp* *s*)
 <proof>

lemma *iexp-eq1*:
fixes *x* :: *real*
defines *f* *s* *m* \equiv *complex-of-real* ((*x* - *s*) \wedge *m*)

shows $iexp\ x =$
 $(\sum k \leq n. (i * x)^{\wedge} k / (fact\ k)) + ((i^{\wedge} (Suc\ n)) / (fact\ n)) * (CLBINT\ s=0..x.$
 $(f\ s\ n) * (iexp\ s))\ (\text{is } ?P\ n)$
 $\langle proof \rangle$

lemma $iexp-eq2$:
fixes $x :: real$
defines $f\ s\ m \equiv complex-of-real\ ((x - s)^{\wedge} m)$
shows $iexp\ x = (\sum k \leq Suc\ n. (i * x)^{\wedge} k / fact\ k) + i^{\wedge} Suc\ n / fact\ n * (CLBINT\ s=0..x. f\ s\ n * (iexp\ s - 1))$
 $\langle proof \rangle$

lemma $abs-LBINT-I0c-abs-power-diff$:
 $|LBINT\ s=0..x. |(x - s)^{\wedge} n|| = |x^{\wedge} (Suc\ n) / (Suc\ n)|$
 $\langle proof \rangle$

lemma $iexp-approx1$: $cmod\ (iexp\ x - (\sum k \leq n. (i * x)^{\wedge} k / fact\ k)) \leq |x|^{\wedge} (Suc\ n) / fact\ (Suc\ n)$
 $\langle proof \rangle$

lemma $iexp-approx2$: $cmod\ (iexp\ x - (\sum k \leq n. (i * x)^{\wedge} k / fact\ k)) \leq 2 * |x|^{\wedge} n / fact\ n$
 $\langle proof \rangle$

lemma $(in\ real-distribution)\ char-approx1$:
assumes $integrable-moments: \bigwedge k. k \leq n \implies integrable\ M\ (\lambda x. x^{\wedge} k)$
shows $cmod\ (char\ M\ t - (\sum k \leq n. ((i * t)^{\wedge} k / fact\ k) * expectation\ (\lambda x. x^{\wedge} k)))$
 \leq
 $(2 * |t|^{\wedge} n / fact\ n) * expectation\ (\lambda x. |x|^{\wedge} n)\ (\text{is } cmod\ (char\ M\ t - ?t1) \leq -)$
 $\langle proof \rangle$

lemma $(in\ real-distribution)\ char-approx2$:
assumes $integrable-moments: \bigwedge k. k \leq n \implies integrable\ M\ (\lambda x. x^{\wedge} k)$
shows $cmod\ (char\ M\ t - (\sum k \leq n. ((i * t)^{\wedge} k / fact\ k) * expectation\ (\lambda x. x^{\wedge} k)))$
 \leq
 $(|t|^{\wedge} n / fact\ (Suc\ n)) * expectation\ (\lambda x. min\ (2 * |x|^{\wedge} n * Suc\ n)\ (|t| * |x|^{\wedge} Suc\ n))$
 $(\text{is } cmod\ (char\ M\ t - ?t1) \leq -)$
 $\langle proof \rangle$

lemma $(in\ real-distribution)\ char-approx3$:
fixes t
assumes
 $integrable-1: integrable\ M\ (\lambda x. x)$ **and**
 $integral-1: expectation\ (\lambda x. x) = 0$ **and**
 $integrable-2: integrable\ M\ (\lambda x. x^{\wedge} 2)$ **and**
 $integral-2: variance\ (\lambda x. x) = \sigma^2$
shows $cmod\ (char\ M\ t - (1 - t^{\wedge} 2 * \sigma^2 / 2)) \leq$
 $(t^{\wedge} 2 / 6) * expectation\ (\lambda x. min\ (6 * x^{\wedge} 2)\ (abs\ t * (abs\ x)^{\wedge} 3))$

<proof>

This is a more familiar textbook formulation in terms of random variables, but we will use the previous version for the CLT.

lemma (in *prob-space*) *char-approx3'*:
fixes $\mu :: \text{real measure}$ **and** X
assumes $rv\text{-}X$ [*simp*]: *random-variable borel* X
and [*simp*]: *integrable* $M\ X$ *integrable* M $(\lambda x. (X\ x)^2)$ *expectation* $X = 0$
and $var\text{-}X$: *variance* $X = \sigma^2$
and $\mu\text{-def}$: $\mu = \text{distr } M \text{ borel } X$
shows $cmod (\text{char } \mu\ t - (1 - t^2 * \sigma^2 / 2)) \leq$
 $(t^2 / 6) * \text{expectation } (\lambda x. \min (6 * (X\ x)^2) (|t| * |X\ x|^3))$
<proof>

this is the formulation in the book – in terms of a random variable *with* the distribution, rather the distribution itself. I don’t know which is more useful, though in principal we can go back and forth between them.

lemma (in *prob-space*) *char-approx1'*:
fixes $\mu :: \text{real measure}$ **and** X
assumes *integrable-moments* : $\bigwedge k. k \leq n \implies \text{integrable } M (\lambda x. X\ x^k)$
and $rv\text{-}X$ [*measurable*]: *random-variable borel* X
and $\mu\text{-distr}$: *distr* $M \text{ borel } X = \mu$
shows $cmod (\text{char } \mu\ t - (\sum k \leq n. ((i * t)^k / \text{fact } k) * \text{expectation } (\lambda x. (X\ x)^k))) \leq$
 $(2 * |t|^n / \text{fact } n) * \text{expectation } (\lambda x. |X\ x|^n)$
<proof>

14.5 Calculation of the Characteristic Function of the Standard Distribution

abbreviation

$\text{std-normal-distribution} \equiv \text{density lborel std-normal-density}$

lemma *real-dist-normal-dist: real-distribution std-normal-distribution*
<proof>

lemma *std-normal-distribution-even-moments:*

fixes $k :: \text{nat}$
shows $(\text{LINT } x | \text{std-normal-distribution}. x^{(2 * k)}) = \text{fact } (2 * k) / (2^k * \text{fact } k)$
and *integrable std-normal-distribution* $(\lambda x. x^{(2 * k)})$
<proof>

lemma *integrable-std-normal-distribution-moment: integrable std-normal-distribution*
 $(\lambda x. x^k)$
<proof>

lemma *integral-std-normal-distribution-moment-odd:*

$odd\ k \implies \text{integral}^L\ \text{std-normal-distribution}\ (\lambda x. x^k) = 0$
 ⟨proof⟩

lemma *std-normal-distribution-even-moments-abs*:

fixes $k :: nat$

shows $(LINT\ x | \text{std-normal-distribution}. |x|^{(2 * k)}) = \text{fact}\ (2 * k) / (2^k * \text{fact}\ k)$
 ⟨proof⟩

lemma *std-normal-distribution-odd-moments-abs*:

fixes $k :: nat$

shows $(LINT\ x | \text{std-normal-distribution}. |x|^{(2 * k + 1)}) = \text{sqrt}\ (2 / \pi) * 2^k * \text{fact}\ k$
 ⟨proof⟩

theorem *char-std-normal-distribution*:

$\text{char}\ \text{std-normal-distribution} = (\lambda t. \text{complex-of-real}\ (\exp\ (-\ (t^2) / 2)))$
 ⟨proof⟩

end

15 Helly’s selection theorem

The set of bounded, monotone, right continuous functions is sequentially compact

theory *Helly-Selection*

imports *HOL-Library.Diagonal-Subsequence Weak-Convergence*

begin

lemma *minus-one-less*: $x - 1 < (x :: real)$

⟨proof⟩

theorem *Helly-selection*:

fixes $f :: nat \Rightarrow real \Rightarrow real$

assumes *rcont*: $\bigwedge n\ x. \text{continuous}\ (\text{at-right}\ x)\ (f\ n)$

assumes *mono*: $\bigwedge n. \text{mono}\ (f\ n)$

assumes *bdd*: $\bigwedge n\ x. |f\ n\ x| \leq M$

shows $\exists s. \text{strict-mono}\ (s :: nat \Rightarrow nat) \wedge (\exists F. (\forall x. \text{continuous}\ (\text{at-right}\ x)\ F) \wedge \text{mono}\ F \wedge (\forall x. |F\ x| \leq M) \wedge$

$(\forall x. \text{continuous}\ (\text{at}\ x)\ F \longrightarrow (\lambda n. f\ (s\ n)\ x) \longrightarrow F\ x))$

⟨proof⟩

definition

$\text{tight} :: (nat \Rightarrow real\ \text{measure}) \Rightarrow bool$

where

$\text{tight}\ \mu \equiv (\forall n. \text{real-distribution}\ (\mu\ n)) \wedge (\forall (\varepsilon :: real) > 0. \exists a\ b :: real. a < b \wedge (\forall n.$

measure (μ n) $\{a < .. b\} > 1 - \varepsilon$)

theorem *tight-imp-convergent-subsubsequence*:

assumes μ : *tight* μ *strict-mono* s

shows $\exists r$ M . *strict-mono* ($r :: \text{nat} \Rightarrow \text{nat}$) \wedge *real-distribution* $M \wedge$ *weak-conv-m* ($\mu \circ s \circ r$) M
 $\langle \text{proof} \rangle$

corollary *tight-subseq-weak-converge*:

fixes $\mu :: \text{nat} \Rightarrow \text{real measure}$ **and** $M :: \text{real measure}$

assumes $\bigwedge n$. *real-distribution* (μ n) *real-distribution* M **and** *tight*: *tight* μ **and**

subseq: $\bigwedge s$ ν . *strict-mono* $s \implies$ *real-distribution* $\nu \implies$ *weak-conv-m* ($\mu \circ s$) ν
 \implies *weak-conv-m* ($\mu \circ s$) M

shows *weak-conv-m* μ M

$\langle \text{proof} \rangle$

end

16 Integral of sinc

theory *Sinc-Integral*

imports *Distributions*

begin

16.1 Various preparatory integrals

Naming convention The theorem name consists of the following parts:

- Kind of integral: *has-bochner-integral* / *integrable* / *LBINT*
- Interval: Interval (0 / infinity / open / closed) (infinity / open / closed)
- Name of the occurring constants: power, exp, m (for minus), scale, sin, ...

lemma *has-bochner-integral-I0i-power-exp-m'*:

has-bochner-integral lborel ($\lambda x. x^k * \exp(-x) * \text{indicator } \{0 ..\} x :: \text{real}$) (*fact* k)

$\langle \text{proof} \rangle$

lemma *has-bochner-integral-I0i-power-exp-m*:

has-bochner-integral lborel ($\lambda x. x^k * \exp(-x) * \text{indicator } \{0 < ..\} x :: \text{real}$) (*fact* k)

$\langle \text{proof} \rangle$

lemma *integrable-I0i-exp-mscale*: $0 < (u :: \text{real}) \implies \text{set-integrable lborel } \{0 < ..\}$

($\lambda x. \exp(-(x * u))$)

$\langle \text{proof} \rangle$

lemma *LBINT-I0i-exp-mscale*: $0 < (u::real) \implies LBINT\ x=0..\infty. \exp(-(x * u)) = 1 / u$
 <proof>

lemma *LBINT-I0c-exp-mscale-sin*:
 $LBINT\ x=0..t. \exp(-(u * x)) * \sin x = (1 / (1 + u^2)) * (1 - \exp(-(u * t)) * (u * \sin t + \cos t))$ (**is** - = ?F t)
 <proof>

lemma *LBINT-I0i-exp-mscale-sin*:
 assumes $0 < x$
 shows $LBINT\ u=0..\infty. |\exp(-u * x) * \sin x| = |\sin x| / x$
 <proof>

lemma
 shows *integrable-inverse-1-plus-square*:
 $set\text{-}integrable\ lborel\ (einterval\ (-\infty)\ \infty)\ (\lambda x. inverse\ (1 + x^2))$
 and *LBINT-inverse-1-plus-square*:
 $LBINT\ x=-\infty..\infty. inverse\ (1 + x^2) = \pi$
 <proof>

lemma
 shows *integrable-I0i-1-div-plus-square*:
 $interval\text{-}lebesgue\text{-}integrable\ lborel\ 0\ \infty\ (\lambda x. 1 / (1 + x^2))$
 and *LBINT-I0i-1-div-plus-square*:
 $LBINT\ x=0..\infty. 1 / (1 + x^2) = \pi / 2$
 <proof>

17 The sinc function, and the sine integral (Si)

abbreviation *sinc* :: $real \Rightarrow real$ **where**
 $sinc \equiv (\lambda x. \text{if } x = 0 \text{ then } 1 \text{ else } \sin x / x)$

lemma *sinc-at-0*: $((\lambda x. \sin x / x)::real) \longrightarrow 1$ (at 0)
 <proof>

lemma *isCont-sinc*: $isCont\ sinc\ x$
 <proof>

lemma *continuous-on-sinc*[*continuous-intros*]:
 $continuous\text{-}on\ S\ f \implies continuous\text{-}on\ S\ (\lambda x. sinc\ (f\ x))$
 <proof>

lemma *borel-measurable-sinc*[*measurable*]: $sinc \in borel\text{-}measurable\ borel$
 <proof>

lemma *sinc-AE*: $AE\ x\ in\ lborel. \sin x / x = sinc\ x$
 <proof>

definition $Si :: real \Rightarrow real$ **where** $Si\ t \equiv LBINT\ x=0..t.\ sin\ x\ /\ x$

lemma *sinc-neg* [simp]: $sinc\ (-\ x) = sinc\ x$
 $\langle proof \rangle$

lemma *Si-alt-def* : $Si\ t = LBINT\ x=0..t.\ sinc\ x$
 $\langle proof \rangle$

lemma *Si-neg*:
assumes $T \geq 0$ **shows** $Si\ (-\ T) = -\ Si\ T$
 $\langle proof \rangle$

lemma *integrable-sinc'*:
 $interval\text{-}lebesgue\text{-}integrable\ lborel\ (ereal\ 0)\ (ereal\ T)\ (\lambda t.\ sin\ (t * \vartheta) /\ t)$
 $\langle proof \rangle$

lemma *DERIV-Si*: $(Si\ has\text{-}real\text{-}derivative\ sinc\ x)\ (at\ x)$
 $\langle proof \rangle$

lemma *isCont-Si*: $isCont\ Si\ x$
 $\langle proof \rangle$

lemma *borel-measurable-Si*[measurable]: $Si \in borel\text{-}measurable\ borel$
 $\langle proof \rangle$

lemma *Si-at-top-LBINT*:
 $((\lambda t.\ (LBINT\ x=0..\infty.\ exp\ (- (x * t)) * (x * sin\ t + cos\ t) /\ (1 + x^2))) \longrightarrow 0)\ at\text{-}top)$
 $\langle proof \rangle$

lemma *Si-at-top-integrable*:
assumes $t \geq 0$
shows $interval\text{-}lebesgue\text{-}integrable\ lborel\ 0\ \infty\ (\lambda x.\ exp\ (- (x * t)) * (x * sin\ t + cos\ t) /\ (1 + x^2))$
 $\langle proof \rangle$

lemma *Si-at-top*: $(Si \longrightarrow pi / 2)\ at\text{-}top$
 $\langle proof \rangle$

17.1 The final theorems: boundedness and scalability

lemma *bounded-Si*: $\exists B.\ \forall T.\ |Si\ T| \leq B$
 $\langle proof \rangle$

lemma *LBINT-I0c-sin-scale-divide*:
assumes $T \geq 0$

shows $LBINT\ t=0..T. \sin(t * \vartheta) / t = \text{sgn } \vartheta * Si(T * |\vartheta|)$
 $\langle proof \rangle$

end

18 The Levy inversion theorem, and the Levy continuity theorem.

theory *Levy*
imports *Characteristic-Functions Helly-Selection Sinc-Integral*
begin

18.1 The Levy inversion theorem

lemma *Levy-Inversion-aux1*:
fixes $a\ b :: \text{real}$
assumes $a \leq b$
shows $((\lambda t. (\text{iexp } -(t * a)) - \text{iexp } -(t * b))) / (\text{i} * t)) \longrightarrow b - a \text{ (at } 0)$
 $(\text{is } (?F \longrightarrow -) \text{ (at } -))$
 $\langle proof \rangle$

lemma *Levy-Inversion-aux2*:
fixes $a\ b\ t :: \text{real}$
assumes $a \leq b$ **and** $t \neq 0$
shows $\text{cmod } ((\text{iexp } (t * b)) - \text{iexp } (t * a)) / (\text{i} * t) \leq b - a \text{ (is } ?F \leq -)$
 $\langle proof \rangle$

theorem (**in** *real-distribution*) *Levy-Inversion*:
fixes $a\ b :: \text{real}$
assumes $a \leq b$
defines $\mu \equiv \text{measure } M$ **and** $\varphi \equiv \text{char } M$
assumes $\mu \{a\} = 0$ **and** $\mu \{b\} = 0$
shows $(\lambda T. 1 / (2 * \pi) * (\text{CLBINT } t=-T..T. (\text{iexp } -(t * a)) - \text{iexp } -(t * b))) / (\text{i} * t) * \varphi\ t)) \longrightarrow \mu \{a <..b\}$
 $(\text{is } (\lambda T. 1 / (2 * \pi) * (\text{CLBINT } t=-T..T. ?F\ t * \varphi\ t)) \longrightarrow \text{of-real } (\mu \{a <..b\})))$
 $\langle proof \rangle$

theorem *Levy-uniqueness*:
fixes $M1\ M2 :: \text{real measure}$
assumes *real-distribution* $M1$ *real-distribution* $M2$ **and**
 $\text{char } M1 = \text{char } M2$
shows $M1 = M2$
 $\langle proof \rangle$

18.2 The Levy continuity theorem

theorem *levy-continuity1:*

fixes $M :: \text{nat} \Rightarrow \text{real measure}$ **and** $M' :: \text{real measure}$

assumes $\bigwedge n. \text{real-distribution } (M\ n) \text{ real-distribution } M' \text{ weak-conv-m } M\ M'$

shows $(\lambda n. \text{char } (M\ n)\ t) \longrightarrow \text{char } M'\ t$

<proof>

theorem *levy-continuity:*

fixes $M :: \text{nat} \Rightarrow \text{real measure}$ **and** $M' :: \text{real measure}$

assumes $\text{real-distr-M} : \bigwedge n. \text{real-distribution } (M\ n)$

and $\text{real-distr-M'} : \text{real-distribution } M'$

and $\text{char-conv} : \bigwedge t. (\lambda n. \text{char } (M\ n)\ t) \longrightarrow \text{char } M'\ t$

shows $\text{weak-conv-m } M\ M'$

<proof>

end

19 The Central Limit Theorem

theory *Central-Limit-Theorem*

imports *Levy*

begin

theorem (*in prob-space*) *central-limit-theorem-zero-mean:*

fixes $X :: \text{nat} \Rightarrow 'a \Rightarrow \text{real}$

and $\mu :: \text{real measure}$

and $\sigma :: \text{real}$

and $S :: \text{nat} \Rightarrow 'a \Rightarrow \text{real}$

assumes $X\text{-indep} : \text{indep-vars } (\lambda i. \text{borel})\ X\ UNIV$

and $X\text{-mean-0} : \bigwedge n. \text{expectation } (X\ n) = 0$

and $\sigma\text{-pos} : \sigma > 0$

and $X\text{-square-integrable} : \bigwedge n. \text{integrable } M\ (\lambda x. (X\ n\ x)^2)$

and $X\text{-variance} : \bigwedge n. \text{variance } (X\ n) = \sigma^2$

and $X\text{-distrib} : \bigwedge n. \text{distr } M\ \text{borel } (X\ n) = \mu$

defines $S\ n \equiv \lambda x. \sum_{i < n.} X\ i\ x$

shows $\text{weak-conv-m } (\lambda n. \text{distr } M\ \text{borel } (\lambda x. S\ n\ x / \text{sqrt } (n * \sigma^2)))\ \text{std-normal-distribution}$

<proof>

theorem (*in prob-space*) *central-limit-theorem:*

fixes $X :: \text{nat} \Rightarrow 'a \Rightarrow \text{real}$

and $\mu :: \text{real measure}$

and $\sigma :: \text{real}$

and $S :: \text{nat} \Rightarrow 'a \Rightarrow \text{real}$

assumes $X\text{-indep} : \text{indep-vars } (\lambda i. \text{borel})\ X\ UNIV$

and $X\text{-mean} : \bigwedge n. \text{expectation } (X\ n) = m$

and $\sigma\text{-pos} : \sigma > 0$

and $X\text{-square-integrable} : \bigwedge n. \text{integrable } M\ (\lambda x. (X\ n\ x)^2)$

and $X\text{-variance} : \bigwedge n. \text{variance } (X\ n) = \sigma^2$

```

and X-distrib:  $\bigwedge n. \text{distr } M \text{ borel } (X \ n) = \mu$ 
defines  $X' \ i \ x \equiv X \ i \ x - m$ 
shows weak-conv-m  $(\lambda n. \text{distr } M \text{ borel } (\lambda x. (\sum i < n. X' \ i \ x) / \text{sqrt } (n * \sigma^2)))$ 
std-normal-distribution
 $\langle \text{proof} \rangle$ 

end

```

```

theory Discrete-Topology
imports HOL-Analysis.Analysis
begin

```

Copy of discrete types with discrete topology. This space is polish.

```

typedef 'a discrete = UNIV::'a set
morphisms of-discrete discrete
 $\langle \text{proof} \rangle$ 

```

```

instantiation discrete :: (type) metric-space
begin

```

```

definition dist-discrete :: 'a discrete  $\Rightarrow$  'a discrete  $\Rightarrow$  real
  where dist-discrete n m = (if n = m then 0 else 1)

```

```

definition uniformity-discrete :: ('a discrete  $\times$  'a discrete) filter where
  (uniformity::('a discrete  $\times$  'a discrete) filter) = (INF e  $\in \{0 < ..\}$ . principal  $\{(x,$ 
y). dist x y < e})

```

```

definition open-discrete :: 'a discrete set  $\Rightarrow$  bool where
  (open::'a discrete set  $\Rightarrow$  bool) U  $\longleftrightarrow$  ( $\forall x \in U. \text{eventually } (\lambda(x', y). x' = x \longrightarrow y$ 
 $\in U)$  uniformity)

```

```

instance  $\langle \text{proof} \rangle$ 
end

```

```

lemma open-discrete: open (S :: 'a discrete set)
 $\langle \text{proof} \rangle$ 

```

```

instance discrete :: (type) complete-space
 $\langle \text{proof} \rangle$ 

```

```

instance discrete :: (countable) countable
 $\langle \text{proof} \rangle$ 

```

```

instance discrete :: (countable) second-countable-topology
 $\langle \text{proof} \rangle$ 

```

```

instance discrete :: (countable) polish-space  $\langle \text{proof} \rangle$ 

```

end

20 Probability mass function

theory *Probability-Mass-Function*

imports

Giry-Monad

HOL-Library.Multiset

begin

Conflicting notation from *HOL-Analysis.Infinite-Sum*

no-notation *Infinite-Sum.abs-summable-on* (**infixr** $\langle \text{abs}'\text{-summable}'\text{-on} \rangle$ 46)

lemma *AE-emeasure-singleton*:

assumes x : *emeasure* $M \{x\} \neq 0$ **and** ae : *AE* x *in* M . $P \ x$ **shows** $P \ x$
 $\langle \text{proof} \rangle$

lemma *AE-measure-singleton*: *measure* $M \{x\} \neq 0 \implies \text{AE } x \text{ in } M. P \ x \implies P \ x$
 $\langle \text{proof} \rangle$

lemma (**in** *finite-measure*) *AE-support-countable*:

assumes [*simp*]: *sets* $M = \text{UNIV}$
shows (*AE* x *in* M . *measure* $M \{x\} \neq 0$) $\longleftrightarrow (\exists S. \text{countable } S \wedge (\text{AE } x \text{ in } M. x \in S))$
 $\langle \text{proof} \rangle$

20.1 PMF as measure

typedef $'a \text{ pmf} = \{M :: 'a \text{ measure. prob-space } M \wedge \text{sets } M = \text{UNIV} \wedge (\text{AE } x \text{ in } M. \text{measure } M \{x\} \neq 0)\}$

morphisms *measure-pmf* *Abs-pmf*
 $\langle \text{proof} \rangle$

declare $[[\text{coercion } \text{measure-pmf}]]$

lemma *prob-space-measure-pmf*: *prob-space* (*measure-pmf* p)
 $\langle \text{proof} \rangle$

interpretation *measure-pmf*: *prob-space* *measure-pmf* M **for** M
 $\langle \text{proof} \rangle$

interpretation *measure-pmf*: *subprob-space* *measure-pmf* M **for** M
 $\langle \text{proof} \rangle$

lemma *subprob-space-measure-pmf*: *subprob-space* (*measure-pmf* x)
 $\langle \text{proof} \rangle$

locale *pmf-as-measure*

begin

setup-lifting *type-definition-pmf*

end

context
begin

interpretation *pmf-as-measure* $\langle \text{proof} \rangle$

lemma *sets-measure-pmf[simp]*: $\text{sets } (\text{measure-pmf } p) = \text{UNIV}$
 $\langle \text{proof} \rangle$

lemma *sets-measure-pmf-count-space[measurable-cong]*:
 $\text{sets } (\text{measure-pmf } M) = \text{sets } (\text{count-space } \text{UNIV})$
 $\langle \text{proof} \rangle$

lemma *space-measure-pmf[simp]*: $\text{space } (\text{measure-pmf } p) = \text{UNIV}$
 $\langle \text{proof} \rangle$

lemma *measure-pmf-UNIV [simp]*: $\text{measure } (\text{measure-pmf } p) \text{ UNIV} = 1$
 $\langle \text{proof} \rangle$

lemma *measure-pmf-in-subprob-algebra[measurable (raw)]*: $\text{measure-pmf } x \in \text{space}$
 $(\text{subprob-algebra } (\text{count-space } \text{UNIV}))$
 $\langle \text{proof} \rangle$

lemma *measurable-pmf-measure1[simp]*: $\text{measurable } (M :: 'a \text{ pmf}) \ N = \text{UNIV} \rightarrow$
 $\text{space } N$
 $\langle \text{proof} \rangle$

lemma *measurable-pmf-measure2[simp]*: $\text{measurable } N \ (M :: 'a \text{ pmf}) = \text{measurable}$
 $N \ (\text{count-space } \text{UNIV})$
 $\langle \text{proof} \rangle$

lemma *measurable-pair-restrict-pmf2*:
assumes *countable* A
assumes *[measurable]*: $\bigwedge y. y \in A \implies (\lambda x. f \ (x, y)) \in \text{measurable } M \ L$
shows $f \in \text{measurable } (M \otimes_M \text{restrict-space } (\text{measure-pmf } N) \ A) \ L$ **(is** $f \in$
 $\text{measurable } ?M \ -)$
 $\langle \text{proof} \rangle$

lemma *measurable-pair-restrict-pmf1*:
assumes *countable* A
assumes *[measurable]*: $\bigwedge x. x \in A \implies (\lambda y. f \ (x, y)) \in \text{measurable } N \ L$
shows $f \in \text{measurable } (\text{restrict-space } (\text{measure-pmf } M) \ A \otimes_M N) \ L$
 $\langle \text{proof} \rangle$

lift-definition *pmf* $:: 'a \text{ pmf} \Rightarrow 'a \Rightarrow \text{real}$ **is** $\lambda M \ x. \text{measure } M \ \{x\}$ $\langle \text{proof} \rangle$

lift-definition *set-pmf* :: 'a pmf \Rightarrow 'a set is $\lambda M. \{x. \text{measure } M \{x\} \neq 0\}$ $\langle \text{proof} \rangle$
declare [[*coercion set-pmf*]]

lemma *AE-measure-pmf*: $AE\ x\ in\ (M::'a\ pmf).\ x \in M$
 $\langle \text{proof} \rangle$

lemma *emeasure-pmf-single-eq-zero-iff*:
fixes $M :: 'a\ pmf$
shows $\text{emeasure } M \{y\} = 0 \longleftrightarrow y \notin M$
 $\langle \text{proof} \rangle$

lemma *AE-measure-pmf-iff*: $(AE\ x\ in\ \text{measure-pmf } M.\ P\ x) \longleftrightarrow (\forall y \in M.\ P\ y)$
 $\langle \text{proof} \rangle$

lemma *AE-pmfI*: $(\bigwedge y. y \in \text{set-pmf } M \implies P\ y) \implies \text{almost-everywhere } (\text{measure-pmf } M)\ P$
 $\langle \text{proof} \rangle$

lemma *countable-set-pmf [simp]*: *countable* (*set-pmf* p)
 $\langle \text{proof} \rangle$

lemma *pmf-positive*: $x \in \text{set-pmf } p \implies 0 < \text{pmf } p\ x$
 $\langle \text{proof} \rangle$

lemma *pmf-nonneg [simp]*: $0 \leq \text{pmf } p\ x$
 $\langle \text{proof} \rangle$

lemma *pmf-not-neg [simp]*: $\neg \text{pmf } p\ x < 0$
 $\langle \text{proof} \rangle$

lemma *pmf-pos [simp]*: $\text{pmf } p\ x \neq 0 \implies \text{pmf } p\ x > 0$
 $\langle \text{proof} \rangle$

lemma *pmf-le-1*: $\text{pmf } p\ x \leq 1$
 $\langle \text{proof} \rangle$

lemma *set-pmf-not-empty*: $\text{set-pmf } M \neq \{\}$
 $\langle \text{proof} \rangle$

lemma *set-pmf-iff*: $x \in \text{set-pmf } M \longleftrightarrow \text{pmf } M\ x \neq 0$
 $\langle \text{proof} \rangle$

lemma *pmf-positive-iff*: $0 < \text{pmf } p\ x \longleftrightarrow x \in \text{set-pmf } p$
 $\langle \text{proof} \rangle$

lemma *set-pmf-eq*: $\text{set-pmf } M = \{x. \text{pmf } M\ x \neq 0\}$
 $\langle \text{proof} \rangle$

lemma *set-pmf-eq'*: $\text{set-pmf } p = \{x. \text{pmf } p \ x > 0\}$
 <proof>

lemma *emeasure-pmf-single*:
 fixes $M :: 'a \text{ pmf}$
 shows $\text{emeasure } M \ \{x\} = \text{pmf } M \ x$
 <proof>

lemma *measure-pmf-single*: $\text{measure } (\text{measure-pmf } M) \ \{x\} = \text{pmf } M \ x$
 <proof>

lemma *emeasure-measure-pmf-finite*: $\text{finite } S \implies \text{emeasure } (\text{measure-pmf } M) \ S = (\sum_{s \in S. \text{pmf } M \ s})$
 <proof>

lemma *measure-measure-pmf-finite*: $\text{finite } S \implies \text{measure } (\text{measure-pmf } M) \ S = \text{sum } (\text{pmf } M) \ S$
 <proof>

lemma *sum-pmf-eq-1*:
 assumes $\text{finite } A \ \text{set-pmf } p \subseteq A$
 shows $(\sum_{x \in A. \text{pmf } p \ x}) = 1$
 <proof>

lemma *nn-integral-measure-pmf-support*:
 fixes $f :: 'a \Rightarrow \text{ennreal}$
 assumes $f: \text{finite } A \ \text{and } nn: \bigwedge x. x \in A \implies 0 \leq f \ x \ \bigwedge x. x \in \text{set-pmf } M \implies x \notin A \implies f \ x = 0$
 shows $(\int^+ x. f \ x \ \partial \text{measure-pmf } M) = (\sum_{x \in A. f \ x * \text{pmf } M \ x})$
 <proof>

lemma *nn-integral-measure-pmf-finite*:
 fixes $f :: 'a \Rightarrow \text{ennreal}$
 assumes $f: \text{finite } (\text{set-pmf } M) \ \text{and } nn: \bigwedge x. x \in \text{set-pmf } M \implies 0 \leq f \ x$
 shows $(\int^+ x. f \ x \ \partial \text{measure-pmf } M) = (\sum_{x \in \text{set-pmf } M. f \ x * \text{pmf } M \ x})$
 <proof>

lemma *integrable-measure-pmf-finite*:
 fixes $f :: 'a \Rightarrow 'b::\{\text{banach, second-countable-topology}\}$
 shows $\text{finite } (\text{set-pmf } M) \implies \text{integrable } M \ f$
 <proof>

lemma *integral-measure-pmf-real*:
 assumes $[simp]: \text{finite } A \ \text{and } \bigwedge a. a \in \text{set-pmf } M \implies f \ a \neq 0 \implies a \in A$
 shows $(\int x. f \ x \ \partial \text{measure-pmf } M) = (\sum_{a \in A. f \ a * \text{pmf } M \ a})$
 <proof>

lemma *integrable-pmf*: $\text{integrable } (\text{count-space } X) \ (\text{pmf } M)$
 <proof>

lemma *integral-pmf*: $(\int x. \text{pmf } M \ x \ \partial \text{count-space } X) = \text{measure } M \ X$
 $\langle \text{proof} \rangle$

lemma *integral-pmf-restrict*:
 $(f :: 'a \Rightarrow 'b :: \{\text{banach, second-countable-topology}\}) \in \text{borel-measurable } (\text{count-space } \text{UNIV}) \implies$
 $(\int x. f \ x \ \partial \text{measure-pmf } M) = (\int x. f \ x \ \partial \text{restrict-space } M \ M)$
 $\langle \text{proof} \rangle$

lemma *emeasure-pmf*: $\text{emeasure } (M :: 'a \text{ pmf}) \ M = 1$
 $\langle \text{proof} \rangle$

lemma *emeasure-pmf-UNIV [simp]*: $\text{emeasure } (\text{measure-pmf } M) \ \text{UNIV} = 1$
 $\langle \text{proof} \rangle$

lemma *in-null-sets-measure-pmfI*:
 $A \cap \text{set-pmf } p = \{\} \implies A \in \text{null-sets } (\text{measure-pmf } p)$
 $\langle \text{proof} \rangle$

lemma *measure-subprob*: $\text{measure-pmf } M \in \text{space } (\text{subprob-algebra } (\text{count-space } \text{UNIV}))$
 $\langle \text{proof} \rangle$

20.2 Monad Interpretation

lemma *measurable-measure-pmf[measurable]*:
 $(\lambda x. \text{measure-pmf } (M \ x)) \in \text{measurable } (\text{count-space } \text{UNIV}) \ (\text{subprob-algebra } (\text{count-space } \text{UNIV}))$
 $\langle \text{proof} \rangle$

lemma *bind-measure-pmf-cong*:
assumes $\bigwedge x. A \ x \in \text{space } (\text{subprob-algebra } N) \ \bigwedge x. B \ x \in \text{space } (\text{subprob-algebra } N)$
assumes $\bigwedge i. i \in \text{set-pmf } x \implies A \ i = B \ i$
shows $\text{bind } (\text{measure-pmf } x) \ A = \text{bind } (\text{measure-pmf } x) \ B$
 $\langle \text{proof} \rangle$

lift-definition *bind-pmf* :: $'a \text{ pmf} \Rightarrow ('a \Rightarrow 'b \text{ pmf}) \Rightarrow 'b \text{ pmf}$ **is** *bind*
 $\langle \text{proof} \rangle$

adhoc-overloading *Monad-Syntax.bind* \equiv *bind-pmf*

lemma *ennreal-pmf-bind*: $\text{pmf } (\text{bind-pmf } N \ f) \ i = (\int^+ x. \text{pmf } (f \ x) \ i \ \partial \text{measure-pmf } N)$
 $\langle \text{proof} \rangle$

lemma *pmf-bind*: $\text{pmf } (\text{bind-pmf } N \ f) \ i = (\int x. \text{pmf } (f \ x) \ i \ \partial \text{measure-pmf } N)$
 $\langle \text{proof} \rangle$

lemma *bind-pmf-const[simp]*: $\text{bind-pmf } M (\lambda x. c) = c$
 $\langle \text{proof} \rangle$

lemma *set-bind-pmf[simp]*: $\text{set-pmf } (\text{bind-pmf } M N) = (\bigcup M \in \text{set-pmf } M. \text{set-pmf } (N M))$
 $\langle \text{proof} \rangle$

lemma *bind-pmf-cong [fundef-cong]*:
assumes $p = q$
shows $(\bigwedge x. x \in \text{set-pmf } q \implies f x = g x) \implies \text{bind-pmf } p f = \text{bind-pmf } q g$
 $\langle \text{proof} \rangle$

lemma *bind-pmf-cong-simp*:
 $p = q \implies (\bigwedge x. x \in \text{set-pmf } q = \text{simp} \implies f x = g x) \implies \text{bind-pmf } p f = \text{bind-pmf } q g$
 $\langle \text{proof} \rangle$

lemma *measure-pmf-bind*: $\text{measure-pmf } (\text{bind-pmf } M f) = (\text{measure-pmf } M \gg (\lambda x. \text{measure-pmf } (f x)))$
 $\langle \text{proof} \rangle$

lemma *nn-integral-bind-pmf[simp]*: $(\int^+ x. f x \partial \text{bind-pmf } M N) = (\int^+ x. \int^+ y. f y \partial N x \partial M)$
 $\langle \text{proof} \rangle$

lemma *emeasure-bind-pmf[simp]*: $\text{emeasure } (\text{bind-pmf } M N) X = (\int^+ x. \text{emeasure } (N x) X \partial M)$
 $\langle \text{proof} \rangle$

lift-definition *return-pmf* :: $'a \Rightarrow 'a \text{ pmf}$ **is** *return* (count-space UNIV)
 $\langle \text{proof} \rangle$

lemma *bind-return-pmf*: $\text{bind-pmf } (\text{return-pmf } x) f = f x$
 $\langle \text{proof} \rangle$

lemma *set-return-pmf[simp]*: $\text{set-pmf } (\text{return-pmf } x) = \{x\}$
 $\langle \text{proof} \rangle$

lemma *bind-return-pmf'*: $\text{bind-pmf } N \text{return-pmf} = N$
 $\langle \text{proof} \rangle$

lemma *bind-assoc-pmf*: $\text{bind-pmf } (\text{bind-pmf } A B) C = \text{bind-pmf } A (\lambda x. \text{bind-pmf } (B x) C)$
 $\langle \text{proof} \rangle$

definition *map-pmf* $f M = \text{bind-pmf } M (\lambda x. \text{return-pmf } (f x))$

lemma *map-bind-pmf*: $\text{map-pmf } f (\text{bind-pmf } M g) = \text{bind-pmf } M (\lambda x. \text{map-pmf } f g x)$

$(g\ x))$
 $\langle proof \rangle$

lemma *bind-map-pmf*: $bind\text{-}pmf\ (map\text{-}pmf\ f\ M)\ g = bind\text{-}pmf\ M\ (\lambda x. g\ (f\ x))$
 $\langle proof \rangle$

lemma *map-pmf-transfer[transfer-rule]*:
 $rel\text{-}fun\ (=)\ (rel\text{-}fun\ cr\text{-}pmf\ cr\text{-}pmf)\ (\lambda f\ M. distr\ M\ (count\text{-}space\ UNIV)\ f)$
 $map\text{-}pmf$
 $\langle proof \rangle$

lemma *map-pmf-rep-eq*:
 $measure\text{-}pmf\ (map\text{-}pmf\ f\ M) = distr\ (measure\text{-}pmf\ M)\ (count\text{-}space\ UNIV)\ f$
 $\langle proof \rangle$

lemma *map-pmf-id[simp]*: $map\text{-}pmf\ id = id$
 $\langle proof \rangle$

lemma *map-pmf-ident[simp]*: $map\text{-}pmf\ (\lambda x. x) = (\lambda x. x)$
 $\langle proof \rangle$

lemma *map-pmf-compose*: $map\text{-}pmf\ (f \circ g) = map\text{-}pmf\ f \circ map\text{-}pmf\ g$
 $\langle proof \rangle$

lemma *map-pmf-comp*: $map\text{-}pmf\ f\ (map\text{-}pmf\ g\ M) = map\text{-}pmf\ (\lambda x. f\ (g\ x))\ M$
 $\langle proof \rangle$

lemma *map-pmf-cong*: $p = q \implies (\bigwedge x. x \in set\text{-}pmf\ q \implies f\ x = g\ x) \implies map\text{-}pmf\ f\ p = map\text{-}pmf\ f\ q$
 $\langle proof \rangle$

lemma *pmf-set-map*: $set\text{-}pmf \circ map\text{-}pmf\ f = (\cdot)\ f \circ set\text{-}pmf$
 $\langle proof \rangle$

lemma *set-map-pmf[simp]*: $set\text{-}pmf\ (map\text{-}pmf\ f\ M) = f'\text{-}set\text{-}pmf\ M$
 $\langle proof \rangle$

lemma *emeasure-map-pmf[simp]*: $emeasure\ (map\text{-}pmf\ f\ M)\ X = emeasure\ M\ (f\ -'\ X)$
 $\langle proof \rangle$

lemma *measure-map-pmf[simp]*: $measure\ (map\text{-}pmf\ f\ M)\ X = measure\ M\ (f\ -'\ X)$
 $\langle proof \rangle$

lemma *nn-integral-map-pmf[simp]*: $(\int^{+x}. f\ x\ \partial map\text{-}pmf\ g\ M) = (\int^{+x}. f\ (g\ x)\ \partial M)$
 $\langle proof \rangle$

lemma *ennreal-pmf-map*: $\text{pmf } (\text{map-pmf } f \text{ } p) \text{ } x = (\int^+ y. \text{indicator } (f - \cdot \{x\}) \text{ } y \text{ } \partial \text{measure-pmf } p)$

<proof>

lemma *pmf-map*: $\text{pmf } (\text{map-pmf } f \text{ } p) \text{ } x = \text{measure } p \text{ } (f - \cdot \{x\})$

<proof>

lemma *nn-integral-pmf*: $(\int^+ x. \text{pmf } p \text{ } x \text{ } \partial \text{count-space } A) = \text{emeasure } (\text{measure-pmf } p) \text{ } A$

<proof>

lemma *integral-map-pmf[simp]*:

fixes $f :: 'a \Rightarrow 'b :: \{\text{banach, second-countable-topology}\}$

shows $\text{integral}^L (\text{map-pmf } g \text{ } p) \text{ } f = \text{integral}^L p \text{ } (\lambda x. f (g \text{ } x))$

<proof>

lemma *integrable-map-pmf-eq [simp]*:

fixes $g :: 'a \Rightarrow 'b :: \{\text{banach, second-countable-topology}\}$

shows $\text{integrable } (\text{map-pmf } f \text{ } p) \text{ } g \longleftrightarrow \text{integrable } (\text{measure-pmf } p) (\lambda x. g (f \text{ } x))$

<proof>

lemma *integrable-map-pmf [intro]*:

fixes $g :: 'a \Rightarrow 'b :: \{\text{banach, second-countable-topology}\}$

shows $\text{integrable } (\text{measure-pmf } p) (\lambda x. g (f \text{ } x)) \implies \text{integrable } (\text{map-pmf } f \text{ } p) \text{ } g$

<proof>

lemma *pmf-abs-summable [intro]*: $\text{pmf } p \text{ } \text{abs-summable-on } A$

<proof>

lemma *measure-pmf-conv-infsetsum*: $\text{measure } (\text{measure-pmf } p) \text{ } A = \text{infsetsum } (\text{pmf } p) \text{ } A$

<proof>

lemma *infsetsum-pmf-eq-1*:

assumes $\text{set-pmf } p \subseteq A$

shows $\text{infsetsum } (\text{pmf } p) \text{ } A = 1$

<proof>

lemma *map-return-pmf [simp]*: $\text{map-pmf } f \text{ } (\text{return-pmf } x) = \text{return-pmf } (f \text{ } x)$

<proof>

lemma *map-pmf-const[simp]*: $\text{map-pmf } (\lambda \cdot. c) \text{ } M = \text{return-pmf } c$

<proof>

lemma *pmf-return [simp]*: $\text{pmf } (\text{return-pmf } x) \text{ } y = \text{indicator } \{y\} \text{ } x$

<proof>

lemma *nn-integral-return-pmf[simp]*: $0 \leq f \text{ } x \implies (\int^+ x. f \text{ } x \text{ } \partial \text{return-pmf } x) = f \text{ } x$

<proof>

lemma *emeasure-return-pmf[simp]:* $\text{emeasure } (\text{return-pmf } x) \ X = \text{indicator } X \ x$
 $\langle \text{proof} \rangle$

lemma *measure-return-pmf [simp]:* $\text{measure-pmf.prob } (\text{return-pmf } x) \ A = \text{indicator } A \ x$
 $\langle \text{proof} \rangle$

lemma *return-pmf-inj[simp]:* $\text{return-pmf } x = \text{return-pmf } y \longleftrightarrow x = y$
 $\langle \text{proof} \rangle$

lemma *map-pmf-eq-return-pmf-iff:*
 $\text{map-pmf } f \ p = \text{return-pmf } x \longleftrightarrow (\forall y \in \text{set-pmf } p. f \ y = x)$
 $\langle \text{proof} \rangle$

definition *pair-pmf* $A \ B = \text{bind-pmf } A \ (\lambda x. \text{bind-pmf } B \ (\lambda y. \text{return-pmf } (x, y)))$

lemma *pmf-pair:* $\text{pmf } (\text{pair-pmf } M \ N) \ (a, b) = \text{pmf } M \ a * \text{pmf } N \ b$
 $\langle \text{proof} \rangle$

lemma *set-pair-pmf[simp]:* $\text{set-pmf } (\text{pair-pmf } A \ B) = \text{set-pmf } A \times \text{set-pmf } B$
 $\langle \text{proof} \rangle$

lemma *measure-pmf-in-subprob-space[measurable (raw)]:*
 $\text{measure-pmf } M \in \text{space } (\text{subprob-algebra } (\text{count-space } \text{UNIV}))$
 $\langle \text{proof} \rangle$

lemma *nn-integral-pair-pmf':* $(\int^+ x. f \ x \ \partial \text{pair-pmf } A \ B) = (\int^+ a. \int^+ b. f \ (a, b) \ \partial B \ \partial A)$
 $\langle \text{proof} \rangle$

lemma *bind-pair-pmf:*
assumes $M[\text{measurable}]: M \in \text{measurable } (\text{count-space } \text{UNIV} \otimes_M \text{count-space } \text{UNIV}) \ (\text{subprob-algebra } N)$
shows $\text{measure-pmf } (\text{pair-pmf } A \ B) \ggg M = (\text{measure-pmf } A \ggg (\lambda x. \text{measure-pmf } B \ggg (\lambda y. M \ (x, y))))$
(is ?L = ?R)
 $\langle \text{proof} \rangle$

lemma *map-fst-pair-pmf:* $\text{map-pmf } \text{fst} \ (\text{pair-pmf } A \ B) = A$
 $\langle \text{proof} \rangle$

lemma *map-snd-pair-pmf:* $\text{map-pmf } \text{snd} \ (\text{pair-pmf } A \ B) = B$
 $\langle \text{proof} \rangle$

lemma *nn-integral-pmf':*
 $\text{inj-on } f \ A \implies (\int^+ x. \text{pmf } p \ (f \ x) \ \partial \text{count-space } A) = \text{emeasure } p \ (f \ ^\circ A)$
 $\langle \text{proof} \rangle$

lemma *pmf-le-0-iff*[simp]: $\text{pmf } M \ p \leq 0 \longleftrightarrow \text{pmf } M \ p = 0$
 ⟨proof⟩

lemma *min-pmf-0*[simp]: $\min (\text{pmf } M \ p) \ 0 = 0 \ \min \ 0 \ (\text{pmf } M \ p) = 0$
 ⟨proof⟩

lemma *pmf-eq-0-set-pmf*: $\text{pmf } M \ p = 0 \longleftrightarrow p \notin \text{set-pmf } M$
 ⟨proof⟩

lemma *pmf-map-inj*: $\text{inj-on } f \ (\text{set-pmf } M) \implies x \in \text{set-pmf } M \implies \text{pmf } (\text{map-pmf } f \ M) \ (f \ x) = \text{pmf } M \ x$
 ⟨proof⟩

lemma *pair-return-pmf* [simp]: $\text{pair-pmf } (\text{return-pmf } x) \ (\text{return-pmf } y) = \text{return-pmf } (x, y)$
 ⟨proof⟩

lemma *pmf-map-inj'*: $\text{inj } f \implies \text{pmf } (\text{map-pmf } f \ M) \ (f \ x) = \text{pmf } M \ x$
 ⟨proof⟩

lemma *expectation-pair-pmf-fst* [simp]:
 fixes $f :: 'a \Rightarrow 'b::\{\text{banach, second-countable-topology}\}$
 shows $\text{measure-pmf.expectation } (\text{pair-pmf } p \ q) \ (\lambda x. f \ (\text{fst } x)) = \text{measure-pmf.expectation } p \ f$
 ⟨proof⟩

lemma *expectation-pair-pmf-snd* [simp]:
 fixes $f :: 'a \Rightarrow 'b::\{\text{banach, second-countable-topology}\}$
 shows $\text{measure-pmf.expectation } (\text{pair-pmf } p \ q) \ (\lambda x. f \ (\text{snd } x)) = \text{measure-pmf.expectation } q \ f$
 ⟨proof⟩

lemma *pmf-map-outside*: $x \notin f \ ` \text{set-pmf } M \implies \text{pmf } (\text{map-pmf } f \ M) \ x = 0$
 ⟨proof⟩

lemma *measurable-set-pmf*[measurable]: $\text{Measurable.pred } (\text{count-space } \text{UNIV}) \ (\lambda x. x \in \text{set-pmf } M)$
 ⟨proof⟩

20.3 PMFs as function

context
 fixes $f :: 'a \Rightarrow \text{real}$
 assumes *nonneg*: $\bigwedge x. 0 \leq f \ x$
 assumes *prob*: $(\int^+ x. f \ x \ \partial \text{count-space } \text{UNIV}) = 1$
begin

lift-definition *embed-pmf* :: $'a \ \text{pmf} \ \text{is density } (\text{count-space } \text{UNIV}) \ (\text{ennreal} \circ f)$
 ⟨proof⟩

lemma *pmf-embed-pmf*: $\text{pmf } \text{embed-pmf } x = f x$

<proof>

lemma *set-embed-pmf*: $\text{set-pmf } \text{embed-pmf} = \{x. f x \neq 0\}$

<proof>

end

lemma *embed-pmf-transfer*:

rel-fun ($\text{eq-onp } (\lambda f. (\forall x. 0 \leq f x) \wedge (\int^+ x. \text{ennreal } (f x) \text{ } \partial \text{count-space UNIV}) = 1))$ $\text{pmf-as-measure.cr-pmf } (\lambda f. \text{density } (\text{count-space UNIV}) (\text{ennreal } \circ f))$ *embed-pmf*
<proof>

lemma *measure-pmf-eq-density*: $\text{measure-pmf } p = \text{density } (\text{count-space UNIV})$

$(\text{pmf } p)$

<proof>

lemma *td-pmf-embed-pmf*:

type-definition $\text{pmf } \text{embed-pmf} \{f::'a \Rightarrow \text{real}. (\forall x. 0 \leq f x) \wedge (\int^+ x. \text{ennreal } (f x) \text{ } \partial \text{count-space UNIV}) = 1\}$
<proof>

end

lemma *nn-integral-measure-pmf*: $(\int^+ x. f x \text{ } \partial \text{measure-pmf } p) = \int^+ x. \text{ennreal } (\text{pmf } p x) * f x \text{ } \partial \text{count-space UNIV}$

<proof>

lemma *integral-measure-pmf*:

fixes $f :: 'a \Rightarrow 'b::\{\text{banach}, \text{second-countable-topology}\}$
assumes $A: \text{finite } A$
shows $(\bigwedge a. a \in \text{set-pmf } M \implies f a \neq 0 \implies a \in A) \implies (\text{LINT } x|M. f x) = (\sum_{a \in A. \text{pmf } M a *_{\mathbb{R}} f a)$
<proof>

lemma *expectation-return-pmf* [*simp*]:

fixes $f :: 'a \Rightarrow 'b::\{\text{banach}, \text{second-countable-topology}\}$
shows $\text{measure-pmf.expectation } (\text{return-pmf } x) f = f x$
<proof>

lemma *pmf-expectation-bind*:

fixes $p :: 'a \text{ pmf}$ **and** $f :: 'a \Rightarrow 'b \text{ pmf}$
and $h :: 'b \Rightarrow 'c::\{\text{banach}, \text{second-countable-topology}\}$
assumes $\text{finite } A \bigwedge x. x \in A \implies \text{finite } (\text{set-pmf } (f x)) \text{ set-pmf } p \subseteq A$
shows $\text{measure-pmf.expectation } (p \gg f) h = (\sum_{a \in A. \text{pmf } p a *_{\mathbb{R}} \text{measure-pmf.expectation } (f a) h)$
<proof>

lemma *continuous-on-LINT-pmf*: — This is dominated convergence!?

fixes $f :: 'i \Rightarrow 'a::\text{topological-space} \Rightarrow 'b::\{\text{banach}, \text{second-countable-topology}\}$

assumes $f: \bigwedge i. i \in \text{set-pmf } M \implies \text{continuous-on } A (f i)$

and $\text{bnd}: \bigwedge a. a \in A \implies i \in \text{set-pmf } M \implies \text{norm } (f i a) \leq B$

shows $\text{continuous-on } A (\lambda a. \text{LINT } i|M. f i a)$

<proof>

lemma *continuous-on-LBINT*:

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

assumes $f: \bigwedge b. a \leq b \implies \text{set-integrable lborel } \{a..b\} f$

shows $\text{continuous-on UNIV } (\lambda b. \text{LBINT } x:\{a..b\}. f x)$

<proof>

locale *pmf-as-function*

begin

setup-lifting *td-pmf-embed-pmf*

lemma *set-pmf-transfer[transfer-rule]*:

assumes *bi-total* A

shows $\text{rel-fun } (\text{pcr-pmf } A) (\text{rel-set } A) (\lambda f. \{x. f x \neq 0\}) \text{ set-pmf}$

<proof>

end

context

begin

interpretation *pmf-as-function* *<proof>*

lemma *pmf-eqI*: $(\bigwedge i. \text{pmf } M i = \text{pmf } N i) \implies M = N$

<proof>

lemma *pmf-eq-iff*: $M = N \longleftrightarrow (\forall i. \text{pmf } M i = \text{pmf } N i)$

<proof>

lemma *pmf-neq-exists-less*:

assumes $M \neq N$

shows $\exists x. \text{pmf } M x < \text{pmf } N x$

<proof>

lemma *bind-commute-pmf*: $\text{bind-pmf } A (\lambda x. \text{bind-pmf } B (C x)) = \text{bind-pmf } B (\lambda y. \text{bind-pmf } A (\lambda x. C x y))$

<proof>

lemma *pair-map-pmf1*: $\text{pair-pmf } (\text{map-pmf } f A) B = \text{map-pmf } (\text{apfst } f) (\text{pair-pmf } A B)$

<proof>

lemma *pair-map-pmf2*: $\text{pair-pmf } A \ (\text{map-pmf } f \ B) = \text{map-pmf } (\text{apsnd } f) \ (\text{pair-pmf } A \ B)$
 $\langle \text{proof} \rangle$

lemma *map-pair*: $\text{map-pmf } (\lambda(a, b). \ (f \ a, \ g \ b)) \ (\text{pair-pmf } A \ B) = \text{pair-pmf } (\text{map-pmf } f \ A) \ (\text{map-pmf } g \ B)$
 $\langle \text{proof} \rangle$

end

lemma *pair-return-pmf1*: $\text{pair-pmf } (\text{return-pmf } x) \ y = \text{map-pmf } (\text{Pair } x) \ y$
 $\langle \text{proof} \rangle$

lemma *pair-return-pmf2*: $\text{pair-pmf } x \ (\text{return-pmf } y) = \text{map-pmf } (\lambda x. \ (x, \ y)) \ x$
 $\langle \text{proof} \rangle$

lemma *pair-pair-pmf*: $\text{pair-pmf } (\text{pair-pmf } u \ v) \ w = \text{map-pmf } (\lambda(x, (y, z)). \ ((x, y), \ z)) \ (\text{pair-pmf } u \ (\text{pair-pmf } v \ w))$
 $\langle \text{proof} \rangle$

lemma *pair-commute-pmf*: $\text{pair-pmf } x \ y = \text{map-pmf } (\lambda(x, y). \ (y, \ x)) \ (\text{pair-pmf } y \ x)$
 $\langle \text{proof} \rangle$

lemma *set-pmf-subset-singleton*: $\text{set-pmf } p \subseteq \{x\} \longleftrightarrow p = \text{return-pmf } x$
 $\langle \text{proof} \rangle$

lemma *bind-eq-return-pmf*:
 $\text{bind-pmf } p \ f = \text{return-pmf } x \longleftrightarrow (\forall y \in \text{set-pmf } p. \ f \ y = \text{return-pmf } x)$
 $(\text{is } ?lhs \longleftrightarrow ?rhs)$
 $\langle \text{proof} \rangle$

lemma *pmf-False-conv-True*: $\text{pmf } p \ \text{False} = 1 - \text{pmf } p \ \text{True}$
 $\langle \text{proof} \rangle$

lemma *pmf-True-conv-False*: $\text{pmf } p \ \text{True} = 1 - \text{pmf } p \ \text{False}$
 $\langle \text{proof} \rangle$

20.4 Conditional Probabilities

lemma *measure-pmf-zero-iff*: $\text{measure } (\text{measure-pmf } p) \ s = 0 \longleftrightarrow \text{set-pmf } p \cap s = \{\}$
 $\langle \text{proof} \rangle$

context

fixes $p :: 'a \ \text{pmf}$ **and** $s :: 'a \ \text{set}$
assumes *not-empty*: $\text{set-pmf } p \cap s \neq \{\}$
begin

interpretation *pmf-as-measure* $\langle \text{proof} \rangle$

lemma *emeasure-measure-pmf-not-zero*: *emeasure* (*measure-pmf* *p*) *s* $\neq 0$
 $\langle \text{proof} \rangle$

lemma *measure-measure-pmf-not-zero*: *measure* (*measure-pmf* *p*) *s* $\neq 0$
 $\langle \text{proof} \rangle$

lift-definition *cond-pmf* :: 'a pmf is
uniform-measure (*measure-pmf* *p*) *s*
 $\langle \text{proof} \rangle$

lemma *pmf-cond*: *pmf cond-pmf* *x* = (if *x* \in *s* then *pmf* *p* *x* / *measure* *p* *s* else 0)
 $\langle \text{proof} \rangle$

lemma *set-cond-pmf[simp]*: *set-pmf cond-pmf* = *set-pmf* *p* \cap *s*
 $\langle \text{proof} \rangle$

end

lemma *measure-pmf-posI*: *x* \in *set-pmf* *p* \implies *x* \in *A* \implies *measure-pmf.prob* *p* *A* > 0
 $\langle \text{proof} \rangle$

lemma *cond-map-pmf*:
assumes *set-pmf* *p* \cap *f* $- 's \neq \{\}$
shows *cond-pmf* (*map-pmf* *f* *p*) *s* = *map-pmf* *f* (*cond-pmf* *p* (*f* $- 's$))
 $\langle \text{proof} \rangle$

lemma *bind-cond-pmf-cancel*:
assumes [*simp*]: $\bigwedge x. x \in \text{set-pmf } p \implies \text{set-pmf } q \cap \{y. R \ x \ y\} \neq \{\}$
assumes [*simp*]: $\bigwedge y. y \in \text{set-pmf } q \implies \text{set-pmf } p \cap \{x. R \ x \ y\} \neq \{\}$
assumes [*simp*]: $\bigwedge x \ y. x \in \text{set-pmf } p \implies y \in \text{set-pmf } q \implies R \ x \ y \implies \text{measure } q \ \{y. R \ x \ y\} = \text{measure } p \ \{x. R \ x \ y\}$
shows *bind-pmf* *p* ($\lambda x. \text{cond-pmf } q \ \{y. R \ x \ y\}$) = *q*
 $\langle \text{proof} \rangle$

20.5 Relator

inductive *rel-pmf* :: ('a \Rightarrow 'b \Rightarrow bool) \Rightarrow 'a pmf \Rightarrow 'b pmf \Rightarrow bool
for *R* *p* *q*

where

$\llbracket \bigwedge x \ y. (x, y) \in \text{set-pmf } pq \implies R \ x \ y;$
 $\text{map-pmf } \text{fst } pq = p; \text{map-pmf } \text{snd } pq = q \rrbracket$
 $\implies \text{rel-pmf } R \ p \ q$

lemma *rel-pmfI*:
assumes *R*: *rel-set* *R* (*set-pmf* *p*) (*set-pmf* *q*)

assumes $eq: \bigwedge x y. x \in \text{set-pmf } p \implies y \in \text{set-pmf } q \implies R x y \implies$
 $\text{measure } p \{x. R x y\} = \text{measure } q \{y. R x y\}$
shows $\text{rel-pmf } R p q$
 $\langle \text{proof} \rangle$

lemma $\text{rel-pmf-imp-rel-set}: \text{rel-pmf } R p q \implies \text{rel-set } R (\text{set-pmf } p) (\text{set-pmf } q)$
 $\langle \text{proof} \rangle$

lemma rel-pmfD-measure :
assumes $\text{rel-R}: \text{rel-pmf } R p q$ **and** $R: \bigwedge a b. R a b \implies R a y \longleftrightarrow R x b$
assumes $x \in \text{set-pmf } p \ y \in \text{set-pmf } q$
shows $\text{measure } p \{x. R x y\} = \text{measure } q \{y. R x y\}$
 $\langle \text{proof} \rangle$

lemma rel-pmf-measureD :
assumes $\text{rel-pmf } R p q$
shows $\text{measure } (\text{measure-pmf } p) A \leq \text{measure } (\text{measure-pmf } q) \{y. \exists x \in A. R x y\}$ **(is ?lhs \leq ?rhs)**
 $\langle \text{proof} \rangle$

lemma $\text{rel-pmf-iff-measure}$:
assumes $\text{symp } R \ \text{transp } R$
shows $\text{rel-pmf } R p q \longleftrightarrow$
 $\text{rel-set } R (\text{set-pmf } p) (\text{set-pmf } q) \wedge$
 $(\forall x \in \text{set-pmf } p. \forall y \in \text{set-pmf } q. R x y \longrightarrow \text{measure } p \{x. R x y\} = \text{measure } q \{y. R x y\})$
 $\langle \text{proof} \rangle$

lemma $\text{quotient-rel-set-disjoint}$:
 $\text{equivp } R \implies C \in \text{UNIV} // \{(x, y). R x y\} \implies \text{rel-set } R A B \implies A \cap C = \{\}$
 $\longleftrightarrow B \cap C = \{\}$
 $\langle \text{proof} \rangle$

lemma $\text{quotientD}: \text{equiv } X R \implies A \in X // R \implies x \in A \implies A = R \text{ “ } \{x\}$
 $\langle \text{proof} \rangle$

lemma $\text{rel-pmf-iff-equivp}$:
assumes $\text{equivp } R$
shows $\text{rel-pmf } R p q \longleftrightarrow (\forall C \in \text{UNIV} // \{(x, y). R x y\}. \text{measure } p C = \text{measure } q C)$
(is - \longleftrightarrow ($\forall C \in - // ?R. -$))
 $\langle \text{proof} \rangle$

bnf $\text{pmf}: 'a \ \text{pmf} \ \text{map}: \text{map-pmf} \ \text{sets}: \text{set-pmf} \ \text{bd} : \text{card-suc} \ \text{natLeq} \ \text{rel}: \text{rel-pmf}$
 $\langle \text{proof} \rangle$

lemma $\text{map-pmf-idI}: (\bigwedge x. x \in \text{set-pmf } p \implies f x = x) \implies \text{map-pmf } f p = p$
 $\langle \text{proof} \rangle$

lemma *rel-pmf-conj[simp]*:

rel-pmf $(\lambda x y. P \wedge Q x y) x y \longleftrightarrow P \wedge \text{rel-pmf } Q x y$

rel-pmf $(\lambda x y. Q x y \wedge P) x y \longleftrightarrow P \wedge \text{rel-pmf } Q x y$

$\langle \text{proof} \rangle$

lemma *rel-pmf-top[simp]*: *rel-pmf top* = *top*

$\langle \text{proof} \rangle$

lemma *rel-pmf-return-pmf1*: *rel-pmf R (return-pmf x) M* $\longleftrightarrow (\forall a \in M. R x a)$

$\langle \text{proof} \rangle$

lemma *rel-pmf-return-pmf2*: *rel-pmf R M (return-pmf x)* $\longleftrightarrow (\forall a \in M. R a x)$

$\langle \text{proof} \rangle$

lemma *rel-return-pmf[simp]*: *rel-pmf R (return-pmf x1) (return-pmf x2)* = *R x1 x2*

$\langle \text{proof} \rangle$

lemma *rel-pmf-False[simp]*: *rel-pmf* $(\lambda x y. \text{False}) x y = \text{False}$

$\langle \text{proof} \rangle$

lemma *rel-pmf-rel-prod*:

rel-pmf $(\text{rel-prod } R S) (\text{pair-pmf } A A') (\text{pair-pmf } B B') \longleftrightarrow \text{rel-pmf } R A B \wedge \text{rel-pmf } S A' B'$

$\langle \text{proof} \rangle$

lemma *rel-pmf-reflI*:

assumes $\bigwedge x. x \in \text{set-pmf } p \implies P x x$

shows *rel-pmf P p p*

$\langle \text{proof} \rangle$

lemma *rel-pmf-bij-betw*:

assumes *f*: *bij-betw f (set-pmf p) (set-pmf q)*

and *eq*: $\bigwedge x. x \in \text{set-pmf } p \implies \text{pmf } p x = \text{pmf } q (f x)$

shows *rel-pmf* $(\lambda x y. f x = y) p q$

$\langle \text{proof} \rangle$

context

begin

interpretation *pmf-as-measure* $\langle \text{proof} \rangle$

definition *join-pmf* *M* = *bind-pmf M* $(\lambda x. x)$

lemma *bind-eq-join-pmf*: *bind-pmf M f* = *join-pmf (map-pmf f M)*

$\langle \text{proof} \rangle$

lemma *join-eq-bind-pmf*: *join-pmf M* = *bind-pmf M id*

$\langle \text{proof} \rangle$

lemma *pmf-join*: $\text{pmf } (\text{join-pmf } N) \ i = (\int M. \text{pmf } M \ i \ \partial \text{measure-pmf } N)$
 ⟨proof⟩

lemma *ennreal-pmf-join*: $\text{ennreal } (\text{pmf } (\text{join-pmf } N) \ i) = (\int^+ M. \text{pmf } M \ i \ \partial \text{measure-pmf } N)$
 ⟨proof⟩

lemma *set-pmf-join-pmf[simp]*: $\text{set-pmf } (\text{join-pmf } f) = (\bigcup p \in \text{set-pmf } f. \text{set-pmf } p)$
 ⟨proof⟩

lemma *join-return-pmf*: $\text{join-pmf } (\text{return-pmf } M) = M$
 ⟨proof⟩

lemma *map-join-pmf*: $\text{map-pmf } f \ (\text{join-pmf } AA) = \text{join-pmf } (\text{map-pmf } (\text{map-pmf } f) \ AA)$
 ⟨proof⟩

lemma *join-map-return-pmf*: $\text{join-pmf } (\text{map-pmf } \text{return-pmf } A) = A$
 ⟨proof⟩

end

lemma *rel-pmf-joinI*:
 assumes $\text{rel-pmf } (\text{rel-pmf } P) \ p \ q$
 shows $\text{rel-pmf } P \ (\text{join-pmf } p) \ (\text{join-pmf } q)$
 ⟨proof⟩

lemma *rel-pmf-bindI*:
 assumes $pq: \text{rel-pmf } R \ p \ q$
 and $fg: \bigwedge x \ y. R \ x \ y \implies \text{rel-pmf } P \ (f \ x) \ (g \ y)$
 shows $\text{rel-pmf } P \ (\text{bind-pmf } p \ f) \ (\text{bind-pmf } q \ g)$
 ⟨proof⟩

Proof that *rel-pmf* preserves orders. Antisymmetry proof follows Thm. 1 in N. Saheb-Djahromi, Cpo’s of measures for nondeterminism, Theoretical Computer Science 12(1):19–37, 1980, [https://doi.org/10.1016/0304-3975\(80\)90003-1](https://doi.org/10.1016/0304-3975(80)90003-1)

lemma
 assumes *: $\text{rel-pmf } R \ p \ q$
 and *reft*: $\text{reftp } R$ and *trans*: $\text{transp } R$
 shows *measure-Ici*: $\text{measure } p \ \{y. R \ x \ y\} \leq \text{measure } q \ \{y. R \ x \ y\}$ (is ?thesis1)
 and *measure-Ioi*: $\text{measure } p \ \{y. R \ x \ y \wedge \neg R \ y \ x\} \leq \text{measure } q \ \{y. R \ x \ y \wedge \neg R \ y \ x\}$ (is ?thesis2)
 ⟨proof⟩

lemma *rel-pmf-inf*:
 fixes $p \ q :: 'a \ \text{pmf}$

assumes 1: $\text{rel-pmf } R \ p \ q$
assumes 2: $\text{rel-pmf } R \ q \ p$
and $\text{refl: reflp } R$ **and** $\text{trans: transp } R$
shows $\text{rel-pmf } (\text{inf } R \ R^{-1-1}) \ p \ q$
 $\langle \text{proof} \rangle$

lemma rel-pmf-antisym:
fixes $p \ q :: 'a \ \text{pmf}$
assumes 1: $\text{rel-pmf } R \ p \ q$
assumes 2: $\text{rel-pmf } R \ q \ p$
and $\text{refl: reflp } R$ **and** $\text{trans: transp } R$ **and** $\text{antisym: antisymp } R$
shows $p = q$
 $\langle \text{proof} \rangle$

lemma $\text{reflp-rel-pmf: reflp } R \implies \text{reflp } (\text{rel-pmf } R)$
 $\langle \text{proof} \rangle$

lemma antisymp-rel-pmf:
 $\llbracket \text{reflp } R; \text{transp } R; \text{antisymp } R \rrbracket$
 $\implies \text{antisymp } (\text{rel-pmf } R)$
 $\langle \text{proof} \rangle$

lemma transp-rel-pmf:
assumes $\text{transp } R$
shows $\text{transp } (\text{rel-pmf } R)$
 $\langle \text{proof} \rangle$

20.6 Distributions

context
begin

interpretation $\text{pmf-as-function } \langle \text{proof} \rangle$

20.6.1 Bernoulli Distribution

lift-definition $\text{bernoulli-pmf} :: \text{real} \Rightarrow \text{bool pmf}$ **is**
 $\lambda p \ b. ((\lambda p. \text{if } b \text{ then } p \text{ else } 1 - p) \circ \text{min } 1 \circ \text{max } 0) \ p$
 $\langle \text{proof} \rangle$

lemma $\text{pmf-bernoulli-True[simp]: } 0 \leq p \implies p \leq 1 \implies \text{pmf } (\text{bernoulli-pmf } p)$
 $\text{True} = p$
 $\langle \text{proof} \rangle$

lemma $\text{pmf-bernoulli-False[simp]: } 0 \leq p \implies p \leq 1 \implies \text{pmf } (\text{bernoulli-pmf } p)$
 $\text{False} = 1 - p$
 $\langle \text{proof} \rangle$

lemma $\text{set-pmf-bernoulli[simp]: } 0 < p \implies p < 1 \implies \text{set-pmf } (\text{bernoulli-pmf } p)$
 $= \text{UNIV}$

$\langle proof \rangle$

lemma *nn-integral-bernoulli-pmf*[simp]:
assumes [simp]: $0 \leq p \leq 1 \wedge x. 0 \leq f x$
shows $(\int^+ x. f x \partial \text{bernoulli-pmf } p) = f \text{ True} * p + f \text{ False} * (1 - p)$
 $\langle proof \rangle$

lemma *integral-bernoulli-pmf*[simp]:
assumes [simp]: $0 \leq p \leq 1$
shows $(\int x. f x \partial \text{bernoulli-pmf } p) = f \text{ True} * p + f \text{ False} * (1 - p)$
 $\langle proof \rangle$

lemma *pmf-bernoulli-half* [simp]: $\text{pmf } (\text{bernoulli-pmf } (1 / 2)) x = 1 / 2$
 $\langle proof \rangle$

lemma *measure-pmf-bernoulli-half*: $\text{measure-pmf } (\text{bernoulli-pmf } (1 / 2)) = \text{uniform-count-measure UNIV}$
 $\langle proof \rangle$

20.6.2 Geometric Distribution

context
fixes $p :: \text{real}$ **assumes** $p[\text{arith}]: 0 < p \leq 1$
begin

lift-definition *geometric-pmf* :: $\text{nat} \rightarrow \text{pmf}$ **is** $\lambda n. (1 - p)^{\wedge n} * p$
 $\langle proof \rangle$

lemma *pmf-geometric*[simp]: $\text{pmf } \text{geometric-pmf } n = (1 - p)^{\wedge n} * p$
 $\langle proof \rangle$

end

lemma *geometric-pmf-1* [simp]: $\text{geometric-pmf } 1 = \text{return-pmf } 0$
 $\langle proof \rangle$

lemma *set-pmf-geometric*: $0 < p \implies p < 1 \implies \text{set-pmf } (\text{geometric-pmf } p) = \text{UNIV}$
 $\langle proof \rangle$

lemma *geometric-sums-times-n*:
fixes $c :: 'a :: \{\text{banach}, \text{real-normed-field}\}$
assumes $\text{norm } c < 1$
shows $(\lambda n. c^{\wedge n} * \text{of-nat } n) \text{ sums } (c / (1 - c)^2)$
 $\langle proof \rangle$

lemma *geometric-sums-times-norm*:
fixes $c :: 'a :: \{\text{banach}, \text{real-normed-field}\}$
assumes $\text{norm } c < 1$

shows $(\lambda n. \text{norm } (c \hat{n} * \text{of-nat } n)) \text{ sums } (\text{norm } c / (1 - \text{norm } c)^2)$
 $\langle \text{proof} \rangle$

lemma *integrable-real-geometric-pmf*:
assumes $p \in \{0 < .. 1\}$
shows *integrable (geometric-pmf p) real*
 $\langle \text{proof} \rangle$

lemma *expectation-geometric-pmf*:
assumes $p \in \{0 < .. 1\}$
shows *measure-pmf.expectation (geometric-pmf p) real = (1 - p) / p*
 $\langle \text{proof} \rangle$

lemma *geometric-bind-pmf-unfold*:
assumes $p \in \{0 < .. 1\}$
shows *geometric-pmf p =*
do {b ← bernoulli-pmf p;
if b then return-pmf 0 else map-pmf Suc (geometric-pmf p)}
 $\langle \text{proof} \rangle$

20.6.3 Uniform Multiset Distribution

context
fixes $M :: 'a \text{ multiset}$ **assumes** *M-not-empty: M ≠ {#}*
begin

lift-definition *pmf-of-multiset* :: $'a \text{ pmf}$ **is** $\lambda x. \text{count } M \ x / \text{size } M$
 $\langle \text{proof} \rangle$

lemma *pmf-of-multiset[simp]*: *pmf pmf-of-multiset x = count M x / size M*
 $\langle \text{proof} \rangle$

lemma *set-pmf-of-multiset[simp]*: *set-pmf pmf-of-multiset = set-mset M*
 $\langle \text{proof} \rangle$

end

20.6.4 Uniform Distribution

context
fixes $S :: 'a \text{ set}$ **assumes** *S-not-empty: S ≠ {}* **and** *S-finite: finite S*
begin

lift-definition *pmf-of-set* :: $'a \text{ pmf}$ **is** $\lambda x. \text{indicator } S \ x / \text{card } S$
 $\langle \text{proof} \rangle$

lemma *pmf-of-set[simp]*: *pmf pmf-of-set x = indicator S x / card S*
 $\langle \text{proof} \rangle$

lemma *set-pmf-of-set[simp]*: *set-pmf pmf-of-set = S*

$\langle \text{proof} \rangle$

lemma *emeasure-pmf-of-set-space[simp]*: *emeasure pmf-of-set* $S = 1$
 $\langle \text{proof} \rangle$

lemma *nn-integral-pmf-of-set*: *nn-integral* (*measure-pmf pmf-of-set*) $f = \text{sum } f \ S$
 $/ \ \text{card } S$
 $\langle \text{proof} \rangle$

lemma *integral-pmf-of-set*: *integral*^L (*measure-pmf pmf-of-set*) $f = \text{sum } f \ S / \ \text{card } S$
 $\langle \text{proof} \rangle$

lemma *emeasure-pmf-of-set*: *emeasure* (*measure-pmf pmf-of-set*) $A = \text{card } (S \cap A) / \ \text{card } S$
 $\langle \text{proof} \rangle$

lemma *measure-pmf-of-set*: *measure* (*measure-pmf pmf-of-set*) $A = \text{card } (S \cap A) / \ \text{card } S$
 $\langle \text{proof} \rangle$

end

lemma *pmf-expectation-bind-pmf-of-set*:
fixes $A :: 'a \text{ set}$ **and** $f :: 'a \Rightarrow 'b \text{ pmf}$
and $h :: 'b \Rightarrow 'c :: \{\text{banach, second-countable-topology}\}$
assumes $A \neq \{\}$ *finite* $A \wedge x. x \in A \implies \text{finite } (\text{set-pmf } (f \ x))$
shows *measure-pmf.expectation* (*pmf-of-set* $A \gg f$) $h =$
 $(\sum a \in A. \text{measure-pmf.expectation } (f \ a) \ h) /_{\mathbb{R}} \text{real } (\text{card } A)$
 $\langle \text{proof} \rangle$

lemma *map-pmf-of-set*:
assumes *finite* $A \ A \neq \{\}$
shows *map-pmf* f (*pmf-of-set* A) = *pmf-of-multiset* (*image-mset* f (*mset-set* A))
(is ?lhs = ?rhs)
 $\langle \text{proof} \rangle$

lemma *pmf-bind-pmf-of-set*:
assumes $A \neq \{\}$ *finite* A
shows *pmf* (*bind-pmf* (*pmf-of-set* A) f) $x =$
 $(\sum xa \in A. \text{pmf } (f \ xa) \ x) / \ \text{real-of-nat } (\text{card } A)$ **(is ?lhs = ?rhs)**
 $\langle \text{proof} \rangle$

lemma *pmf-of-set-singleton*: *pmf-of-set* $\{x\} = \text{return-pmf } x$
 $\langle \text{proof} \rangle$

lemma *map-pmf-of-set-inj*:
assumes $f: \text{inj-on } f \ A$

and $[simp]: A \neq \{\}$ *finite* A
shows $map\text{-}pmf\ f\ (pmf\text{-}of\text{-}set\ A) = pmf\text{-}of\text{-}set\ (f\ ` A)$ (**is** $?lhs = ?rhs$)
 $\langle proof \rangle$

lemma $map\text{-}pmf\text{-}of\text{-}set\text{-}bij\text{-}betw$:
assumes $bij\text{-}betw\ f\ A\ B$ $A \neq \{\}$ *finite* A
shows $map\text{-}pmf\ f\ (pmf\text{-}of\text{-}set\ A) = pmf\text{-}of\text{-}set\ B$
 $\langle proof \rangle$

Choosing an element uniformly at random from the union of a disjoint family of finite non-empty sets with the same size is the same as first choosing a set from the family uniformly at random and then choosing an element from the chosen set uniformly at random.

lemma $pmf\text{-}of\text{-}set\text{-}UN$:
assumes $finite\ (\bigcup (f\ ` A))$ $A \neq \{\}$ $\bigwedge x. x \in A \implies f\ x \neq \{\}$
 $\bigwedge x. x \in A \implies card\ (f\ x) = n$ *disjoint-family-on* $f\ A$
shows $pmf\text{-}of\text{-}set\ (\bigcup (f\ ` A)) = do\ \{x \leftarrow pmf\text{-}of\text{-}set\ A; pmf\text{-}of\text{-}set\ (f\ x)\}$
(**is** $?lhs = ?rhs$)
 $\langle proof \rangle$

lemma $bernoulli\text{-}pmf\text{-}half\text{-}conv\text{-}pmf\text{-}of\text{-}set$: $bernoulli\text{-}pmf\ (1 / 2) = pmf\text{-}of\text{-}set\ UNIV$
 $\langle proof \rangle$

20.6.5 Poisson Distribution

context

fixes $rate :: real$ **assumes** $rate\text{-}pos$: $0 < rate$

begin

lift-definition $poisson\text{-}pmf :: nat\ pmf$ **is** $\lambda k. rate^k / fact\ k * exp\ (-rate)$
 $\langle proof \rangle$

lemma $pmf\text{-}poisson[simp]$: $pmf\ poisson\text{-}pmf\ k = rate^k / fact\ k * exp\ (-rate)$
 $\langle proof \rangle$

lemma $set\text{-}pmf\text{-}poisson[simp]$: $set\text{-}pmf\ poisson\text{-}pmf = UNIV$
 $\langle proof \rangle$

end

20.6.6 Binomial Distribution

context

fixes $n :: nat$ **and** $p :: real$ **assumes** $p\text{-nonneg}$: $0 \leq p$ **and** $p\text{-le-1}$: $p \leq 1$

begin

lift-definition $binomial\text{-}pmf :: nat\ pmf$ **is** $\lambda k. (n\ choose\ k) * p^k * (1 - p)^{(n - k)}$
 $\langle proof \rangle$

lemma *pmf-binomial[simp]*: $\text{pmf binomial-pmf } k = (n \text{ choose } k) * p^k * (1 - p)^{(n - k)}$
 ⟨proof⟩

lemma *set-pmf-binomial-eq*: $\text{set-pmf binomial-pmf} = (\text{if } p = 0 \text{ then } \{0\} \text{ else if } p = 1 \text{ then } \{n\} \text{ else } \{.. n\})$
 ⟨proof⟩

end

end

lemma *set-pmf-binomial-0[simp]*: $\text{set-pmf (binomial-pmf } n \ 0) = \{0\}$
 ⟨proof⟩

lemma *set-pmf-binomial-1[simp]*: $\text{set-pmf (binomial-pmf } n \ 1) = \{n\}$
 ⟨proof⟩

lemma *set-pmf-binomial[simp]*: $0 < p \implies p < 1 \implies \text{set-pmf (binomial-pmf } n \ p) = \{..n\}$
 ⟨proof⟩

lemma *finite-set-pmf-binomial-pmf [intro]*: $p \in \{0..1\} \implies \text{finite (set-pmf (binomial-pmf } n \ p))}$
 ⟨proof⟩

lemma *expectation-binomial-pmf'*:
fixes $f :: \text{nat} \Rightarrow 'a :: \{\text{banach, second-countable-topology}\}$
assumes $p: p \in \{0..1\}$
shows $\text{measure-pmf.expectation (binomial-pmf } n \ p) f = (\sum k \leq n. (\text{real } (n \text{ choose } k) * p^k * (1 - p)^{(n - k)}) *_R f \ k)$
 ⟨proof⟩

lemma *integrable-binomial-pmf [simp, intro]*:
fixes $f :: \text{nat} \Rightarrow 'a :: \{\text{banach, second-countable-topology}\}$
assumes $p: p \in \{0..1\}$
shows $\text{integrable (binomial-pmf } n \ p) f$
 ⟨proof⟩

context includes *lifting-syntax*
begin

lemma *bind-pmf-parametric [transfer-rule]*:
 $(\text{rel-pmf } A \implies (A \implies \text{rel-pmf } B) \implies \text{rel-pmf } B) \text{ bind-pmf bind-pmf}$
 ⟨proof⟩

lemma *return-pmf-parametric [transfer-rule]*: $(A \implies \text{rel-pmf } A) \text{ return-pmf return-pmf}$

<proof>

end

primrec *replicate-pmf* :: *nat* \Rightarrow *'a pmf* \Rightarrow *'a list pmf* **where**

replicate-pmf 0 = *return-pmf* []

| *replicate-pmf* (Suc *n*) *p* = *do* {*x* \leftarrow *p*; *xs* \leftarrow *replicate-pmf* *n* *p*; *return-pmf* (*x*#*xs*)}

lemma *replicate-pmf-1*: *replicate-pmf* 1 *p* = *map-pmf* ($\lambda x. [x]$) *p*

<proof>

lemma *set-replicate-pmf*:

set-pmf (*replicate-pmf* *n* *p*) = {*xs* ∈ *lists* (*set-pmf* *p*). *length xs* = *n*}

<proof>

lemma *replicate-pmf-distrib*:

replicate-pmf (*m* + *n*) *p* =

do {*xs* \leftarrow *replicate-pmf* *m* *p*; *ys* \leftarrow *replicate-pmf* *n* *p*; *return-pmf* (*xs* @ *ys*)}

<proof>

lemma *power-diff'*:

assumes *b* \leq *a*

shows $x \wedge (a - b) = (\text{if } x = 0 \wedge a = b \text{ then } 1 \text{ else } x \wedge a / (x::'a::\text{field}) \wedge b)$

<proof>

lemma *binomial-pmf-Suc*:

assumes *p* \in {0..1}

shows *binomial-pmf* (Suc *n*) *p* =

do {*b* \leftarrow *bernoulli-pmf* *p*;

k \leftarrow *binomial-pmf* *n* *p*;

return-pmf ((*if* *b* *then* 1 *else* 0) + *k*)}

(**is** - = ?*rhs*)

<proof>

lemma *binomial-pmf-0*: *p* \in {0..1} \implies *binomial-pmf* 0 *p* = *return-pmf* 0

<proof>

lemma *binomial-pmf-altdef*:

assumes *p* \in {0..1}

shows *binomial-pmf* *n* *p* = *map-pmf* (*length* \circ *filter id*) (*replicate-pmf* *n* (*bernoulli-pmf* *p*))

<proof>

20.7 Negative Binomial distribution

The negative binomial distribution counts the number of times a weighted coin comes up tails before having come up heads *n* times. In other words: how many failures do we see before seeing the *n*-th success?

An alternative view is that the negative binomial distribution is the sum of n i.i.d. geometric variables (this is the definition that we use).

Note that there are sometimes different conventions for this distributions in the literature; for instance, sometimes the number of *attempts* is counted instead of the number of failures. This only shifts the entire distribution by a constant number and is thus not a big difference. I think that the convention we use is the most natural one since the support of the distribution starts at 0, whereas for the other convention it starts at n .

primrec *neg-binomial-pmf* :: *nat* \Rightarrow *real* \Rightarrow *nat pmf* **where**
 neg-binomial-pmf 0 p = *return-pmf* 0
 | *neg-binomial-pmf* (Suc n) p =
 map-pmf ($\lambda(x,y). (x + y)$) (*pair-pmf* (*geometric-pmf* p) (*neg-binomial-pmf* n p))

lemma *neg-binomial-pmf-Suc-0* [simp]: *neg-binomial-pmf* (Suc 0) p = *geometric-pmf* p
 <proof>

lemmas *neg-binomial-pmf-Suc* [simp del] = *neg-binomial-pmf.simps*(2)

lemma *neg-binomial-prob-1* [simp]: *neg-binomial-pmf* n 1 = *return-pmf* 0
 <proof>

We can now show the aforementioned intuition about counting the failures before the n -th success with the following recurrence:

lemma *neg-binomial-pmf-unfold*:
 assumes $p: p \in \{0 <.. 1\}$
 shows *neg-binomial-pmf* (Suc n) p =
 do { $b \leftarrow \text{bernoulli-pmf } p$;
 if b then *neg-binomial-pmf* n p else *map-pmf* Suc (*neg-binomial-pmf* (Suc n) p) }
 (**is** - = ?*rhs*)
 <proof>

Next, we show an explicit formula for the probability mass function of the negative binomial distribution:

lemma *pmf-neg-binomial*:
 assumes $p: p \in \{0 <.. 1\}$
 shows *pmf* (*neg-binomial-pmf* n p) k = *real* (($k + n - 1$) *choose* k) * p^k * ($1 - p$) k
 <proof>

lemma *gbinomial-0-left*: 0 *gchoose* k = (if $k = 0$ then 1 else 0)
 <proof>

The following alternative formula highlights why it is called ‘negative binomial distribution’:

lemma *pmf-neg-binomial'*:
assumes $p: p \in \{0 < .. 1\}$
shows $\text{pmf } (\text{neg-binomial-pmf } n \ p) \ k = (-1) \wedge k * ((-\text{real } n) \text{ gchoose } k) * p \wedge n * (1 - p) \wedge k$
 $\langle \text{proof} \rangle$

The cumulative distribution function of the negative binomial distribution can be expressed in terms of that of the ‘normal’ binomial distribution.

lemma *prob-neg-binomial-pmf-atMost*:
assumes $p: p \in \{0 < .. 1\}$
shows $\text{measure-pmf.prob } (\text{neg-binomial-pmf } n \ p) \ \{..k\} = \text{measure-pmf.prob } (\text{binomial-pmf } (n + k) \ (1 - p)) \ \{..k\}$
 $\langle \text{proof} \rangle$

lemma *prob-neg-binomial-pmf-lessThan*:
assumes $p: p \in \{0 < .. 1\}$
shows $\text{measure-pmf.prob } (\text{neg-binomial-pmf } n \ p) \ \{..<k\} = \text{measure-pmf.prob } (\text{binomial-pmf } (n + k - 1) \ (1 - p)) \ \{..<k\}$
 $\langle \text{proof} \rangle$

The expected value of the negative binomial distribution is $n(1 - p)/p$:

lemma *nn-integral-neg-binomial-pmf-real*:
assumes $p: p \in \{0 < .. 1\}$
shows $\text{nn-integral } (\text{measure-pmf } (\text{neg-binomial-pmf } n \ p)) \ \text{of-nat} = \text{ennreal } (n * (1 - p) / p)$
 $\langle \text{proof} \rangle$

lemma *integrable-neg-binomial-pmf-real*:
assumes $p: p \in \{0 < .. 1\}$
shows $\text{integrable } (\text{measure-pmf } (\text{neg-binomial-pmf } n \ p)) \ \text{real}$
 $\langle \text{proof} \rangle$

lemma *expectation-neg-binomial-pmf*:
assumes $p: p \in \{0 < .. 1\}$
shows $\text{measure-pmf.expectation } (\text{neg-binomial-pmf } n \ p) \ \text{real} = n * (1 - p) / p$
 $\langle \text{proof} \rangle$

20.8 PMFs from association lists

definition *pmf-of-list* :: $('a \times \text{real}) \text{ list} \Rightarrow 'a \text{ pmf}$ **where**
 $\text{pmf-of-list } xs = \text{embed-pmf } (\lambda x. \text{sum-list } (\text{map snd } (\text{filter } (\lambda z. \text{fst } z = x) \ xs)))$

definition *pmf-of-list-wf* **where**
 $\text{pmf-of-list-wf } xs \longleftrightarrow (\forall x \in \text{set } (\text{map snd } xs) . x \geq 0) \wedge \text{sum-list } (\text{map snd } xs) = 1$

lemma *pmf-of-list-wfI*:
 $(\bigwedge x. x \in \text{set } (\text{map snd } xs) \Rightarrow x \geq 0) \Rightarrow \text{sum-list } (\text{map snd } xs) = 1 \Rightarrow \text{pmf-of-list-wf } xs$

$\langle proof \rangle$

context
begin

private lemma *pmf-of-list-aux*:

assumes $\bigwedge x. x \in \text{set } (\text{map } \text{snd } xs) \implies x \geq 0$

assumes $\text{sum-list } (\text{map } \text{snd } xs) = 1$

shows $(\int^+ x. \text{ennreal } (\text{sum-list } (\text{map } \text{snd } [z \leftarrow xs . \text{fst } z = x]))) \partial \text{count-space UNIV} = 1$

$\langle proof \rangle$

lemma *pmf-pmf-of-list*:

assumes *pmf-of-list-wf xs*

shows $\text{pmf } (\text{pmf-of-list } xs) \ x = \text{sum-list } (\text{map } \text{snd } (\text{filter } (\lambda z. \text{fst } z = x) \ xs))$

$\langle proof \rangle$

end

lemma *set-pmf-of-list*:

assumes *pmf-of-list-wf xs*

shows $\text{set-pmf } (\text{pmf-of-list } xs) \subseteq \text{set } (\text{map } \text{fst } xs)$

$\langle proof \rangle$

lemma *finite-set-pmf-of-list*:

assumes *pmf-of-list-wf xs*

shows $\text{finite } (\text{set-pmf } (\text{pmf-of-list } xs))$

$\langle proof \rangle$

lemma *emeasure-Int-set-pmf*:

$\text{emeasure } (\text{measure-pmf } p) \ (A \cap \text{set-pmf } p) = \text{emeasure } (\text{measure-pmf } p) \ A$

$\langle proof \rangle$

lemma *measure-Int-set-pmf*:

$\text{measure } (\text{measure-pmf } p) \ (A \cap \text{set-pmf } p) = \text{measure } (\text{measure-pmf } p) \ A$

$\langle proof \rangle$

lemma *measure-prob-cong-0*:

assumes $\bigwedge x. x \in A - B \implies \text{pmf } p \ x = 0$

assumes $\bigwedge x. x \in B - A \implies \text{pmf } p \ x = 0$

shows $\text{measure } (\text{measure-pmf } p) \ A = \text{measure } (\text{measure-pmf } p) \ B$

$\langle proof \rangle$

lemma *emeasure-pmf-of-list*:

assumes *pmf-of-list-wf xs*

shows $\text{emeasure } (\text{pmf-of-list } xs) \ A = \text{ennreal } (\text{sum-list } (\text{map } \text{snd } (\text{filter } (\lambda x. \text{fst } x \in A) \ xs))))$

$\langle proof \rangle$

lemma *measure-pmf-of-list*:
assumes *pmf-of-list-wf xs*
shows $\text{measure } (\text{pmf-of-list } xs) \ A = \text{sum-list } (\text{map } \text{snd } (\text{filter } (\lambda x. \text{fst } x \in A) \ xs))$
 $\langle \text{proof} \rangle$

lemma *sum-list-nonneg-eq-zero-iff*:
fixes $xs :: 'a :: \text{linordered-ab-group-add list}$
shows $(\bigwedge x. x \in \text{set } xs \implies x \geq 0) \implies \text{sum-list } xs = 0 \longleftrightarrow \text{set } xs \subseteq \{0\}$
 $\langle \text{proof} \rangle$

lemma *sum-list-filter-nonzero*:
 $\text{sum-list } (\text{filter } (\lambda x. x \neq 0) \ xs) = \text{sum-list } xs$
 $\langle \text{proof} \rangle$

lemma *set-pmf-of-list-eq*:
assumes $\text{pmf-of-list-wf } xs \ \bigwedge x. x \in \text{snd } ' \text{ set } xs \implies x > 0$
shows $\text{set-pmf } (\text{pmf-of-list } xs) = \text{fst } ' \text{ set } xs$
 $\langle \text{proof} \rangle$

lemma *pmf-of-list-remove-zeros*:
assumes $\text{pmf-of-list-wf } xs$
defines $xs' \equiv \text{filter } (\lambda z. \text{snd } z \neq 0) \ xs$
shows $\text{pmf-of-list-wf } xs' \ \text{pmf-of-list } xs' = \text{pmf-of-list } xs$
 $\langle \text{proof} \rangle$

end

21 Code generation for PMFs

theory *PMF-Impl*
imports *Probability-Mass-Function HOL-Library.AList-Mapping*
begin

21.1 General code generation setup

definition *pmf-of-mapping* :: $('a, \text{real}) \text{ mapping} \Rightarrow 'a \text{ pmf}$ **where**
 $\text{pmf-of-mapping } m = \text{embed-pmf } (\text{Mapping.lookup-default } 0 \ m)$

lemma *nn-integral-lookup-default*:
fixes $m :: ('a, \text{real}) \text{ mapping}$
assumes $\text{finite } (\text{Mapping.keys } m) \ \text{All-mapping } m \ (\lambda x. x \geq 0)$
shows $\text{nn-integral } (\text{count-space UNIV}) \ (\lambda k. \text{ennreal } (\text{Mapping.lookup-default } 0 \ m \ k)) =$
 $\text{ennreal } (\sum k \in \text{Mapping.keys } m. \text{Mapping.lookup-default } 0 \ m \ k)$
 $\langle \text{proof} \rangle$

lemma *pmf-of-mapping*:

assumes *finite* (*Mapping.keys* *m*) *All-mapping* *m* ($\lambda p. p \geq 0$)
assumes $(\sum_{x \in \text{Mapping.keys } m} \text{Mapping.lookup-default } 0 \text{ } m \text{ } x) = 1$
shows $\text{pmf } (\text{pmf-of-mapping } m) \text{ } x = \text{Mapping.lookup-default } 0 \text{ } m \text{ } x$
 $\langle \text{proof} \rangle$

lemma *pmf-of-set-pmf-of-mapping*:

assumes $A \neq \{\}$ *set* *xs* = *A* *distinct* *xs*
shows $\text{pmf-of-set } A = \text{pmf-of-mapping } (\text{Mapping.tabulate } xs \text{ } (\lambda x. 1 / \text{real } (\text{length } xs)))$
 $(\text{is } ?lhs = ?rhs)$
 $\langle \text{proof} \rangle$

lift-definition *mapping-of-pmf* :: $'a \text{ pmf} \Rightarrow ('a, \text{real}) \text{ mapping}$ **is**

$\lambda p \text{ } x. \text{ if } \text{pmf } p \text{ } x = 0 \text{ then None else Some } (\text{pmf } p \text{ } x) \langle \text{proof} \rangle$

lemma *lookup-default-mapping-of-pmf*:

$\text{Mapping.lookup-default } 0 \text{ } (\text{mapping-of-pmf } p) \text{ } x = \text{pmf } p \text{ } x$
 $\langle \text{proof} \rangle$

context

begin

interpretation *pmf-as-function* $\langle \text{proof} \rangle$

lemma *nn-integral-pmf-eq-1*: $(\int^+ x. \text{ennreal } (\text{pmf } p \text{ } x) \text{ } \partial \text{count-space } UNIV) = 1$
 $\langle \text{proof} \rangle$

end

lemma *pmf-of-mapping-mapping-of-pmf* [*code abstype*]:

$\text{pmf-of-mapping } (\text{mapping-of-pmf } p) = p$
 $\langle \text{proof} \rangle$

lemma *mapping-of-pmfI*:

assumes $\bigwedge x. x \in \text{Mapping.keys } m \implies \text{Mapping.lookup } m \text{ } x = \text{Some } (\text{pmf } p \text{ } x)$
assumes $\text{Mapping.keys } m = \text{set-pmf } p$
shows $\text{mapping-of-pmf } p = m$
 $\langle \text{proof} \rangle$

lemma *mapping-of-pmfI'*:

assumes $\bigwedge x. x \in \text{Mapping.keys } m \implies \text{Mapping.lookup-default } 0 \text{ } m \text{ } x = \text{pmf } p \text{ } x$
assumes $\text{Mapping.keys } m = \text{set-pmf } p$
shows $\text{mapping-of-pmf } p = m$
 $\langle \text{proof} \rangle$

lemma *return-pmf-code* [*code abstract*]:

$\text{mapping-of-pmf } (\text{return-pmf } x) = \text{Mapping.update } x \text{ } 1 \text{ } \text{Mapping.empty}$
 $\langle \text{proof} \rangle$

lemma *pmf-of-set-code-aux*:

assumes $A \neq \{\}$ *set* $xs = A$ *distinct* xs
shows $\text{mapping-of-pmf } (\text{pmf-of-set } A) = \text{Mapping.tabulate } xs \ (\lambda_. 1 / \text{real } (\text{length } xs))$
<proof>

definition *pmf-of-set-impl* **where**

$\text{pmf-of-set-impl } A = \text{mapping-of-pmf } (\text{pmf-of-set } A)$

lemma *pmf-of-set-impl-code-alt*:

assumes $A \neq \{\}$ *finite* A
shows $\text{pmf-of-set-impl } A =$
 $(\text{let } p = 1 / \text{real } (\text{card } A)$
 $\text{in } \text{Finite-Set.fold } (\lambda x. \text{Mapping.update } x \ p) \ \text{Mapping.empty } A)$
<proof>

lemma *pmf-of-set-impl-code* [code]:

$\text{pmf-of-set-impl } (\text{set } xs) =$
 $(\text{if } xs = [] \text{ then}$
 $\text{Code.abort } (\text{STR "pmf-of-set of empty set"}) \ (\lambda_. \text{mapping-of-pmf } (\text{pmf-of-set } (\text{set } xs)))$
 $\text{else let } xs' = \text{remdups } xs; p = 1 / \text{real } (\text{length } xs') \text{ in}$
 $\text{Mapping.tabulate } xs' \ (\lambda_. p))$
<proof>

lemma *pmf-of-set-code* [code abstract]:

$\text{mapping-of-pmf } (\text{pmf-of-set } A) = \text{pmf-of-set-impl } A$
<proof>

lemma *pmf-of-multiset-pmf-of-mapping*:

assumes $A \neq \{\#\}$ *set* $xs = \text{set-mset } A$ *distinct* xs
shows $\text{mapping-of-pmf } (\text{pmf-of-multiset } A) = \text{Mapping.tabulate } xs \ (\lambda x. \text{count } A \ x / \text{real } (\text{size } A))$
<proof>

definition *pmf-of-multiset-impl* **where**

$\text{pmf-of-multiset-impl } A = \text{mapping-of-pmf } (\text{pmf-of-multiset } A)$

lemma *pmf-of-multiset-impl-code-alt*:

assumes $A \neq \{\#\}$
shows $\text{pmf-of-multiset-impl } A =$
 $(\text{let } p = 1 / \text{real } (\text{size } A)$
 $\text{in } \text{fold-mset } (\lambda x. \text{Mapping.map-default } x \ 0 \ ((+) \ p)) \ \text{Mapping.empty } A)$
<proof>

lemma *pmf-of-multiset-impl-code* [code]:

```

pmf-of-multiset-impl (mset xs) =
  (if xs = [] then
    Code.abort (STR "pmf-of-multiset of empty multiset")
    (λ-. mapping-of-pmf (pmf-of-multiset (mset xs)))
  else let xs' = remdups xs; p = 1 / real (length xs) in
    Mapping.tabulate xs' (λx. real (count (mset xs) x) * p))
⟨proof⟩

```

lemma *pmf-of-multiset-code* [code abstract]:
mapping-of-pmf (pmf-of-multiset A) = pmf-of-multiset-impl A
 ⟨proof⟩

lemma *bernoulli-pmf-code* [code abstract]:
mapping-of-pmf (bernoulli-pmf p) =
 (if $p \leq 0$ then *Mapping.update False 1 Mapping.empty*
 else if $p \geq 1$ then *Mapping.update True 1 Mapping.empty*
 else *Mapping.update False (1 - p) (Mapping.update True p Mapping.empty)*)
 ⟨proof⟩

lemma *pmf-code* [code]: *pmf p x = Mapping.lookup-default 0 (mapping-of-pmf p) x*
 ⟨proof⟩

lemma *set-pmf-code* [code]: *set-pmf p = Mapping.keys (mapping-of-pmf p)*
 ⟨proof⟩

lemma *keys-mapping-of-pmf* [simp]: *Mapping.keys (mapping-of-pmf p) = set-pmf p*
 ⟨proof⟩

definition *fold-combine-plus* **where**

fold-combine-plus = comm-monoid-set.F (Mapping.combine ((+) :: real ⇒ -)) Mapping.empty

context
begin

interpretation *fold-combine-plus*: *combine-mapping-abel-semigroup (+) :: real ⇒*

-

```

⟨proof⟩ lemma lookup-default-fold-combine-plus:
fixes  $A :: 'b$  set and  $f :: 'b \Rightarrow ('a, \text{real})$  mapping
assumes finite A
shows Mapping.lookup-default 0 (fold-combine-plus f A) x =
  ( $\sum y \in A. \text{Mapping.lookup-default } 0 (f y) x$ )
⟨proof⟩ lemma keys-fold-combine-plus:

```

$finite\ A \implies Mapping.keys\ (fold-combine-plus\ f\ A) = (\bigcup_{x \in A}. Mapping.keys\ (f\ x))$
 ⟨proof⟩ **lemma** *fold-combine-plus-code* [code]:
 $fold-combine-plus\ g\ (set\ xs) = foldr\ (\lambda x. Mapping.combine\ (+)\ (g\ x))\ (remdups\ xs)\ Mapping.empty$
 ⟨proof⟩ **lemma** *lookup-default-0-map-values*:
assumes $f\ x\ 0 = 0$
shows $Mapping.lookup-default\ 0\ (Mapping.map-values\ f\ m)\ x = f\ x\ (Mapping.lookup-default\ 0\ m\ x)$
 ⟨proof⟩ **lemma** *mapping-of-bind-pmf*:
assumes $finite\ (set-pmf\ p)$
shows $mapping-of-pmf\ (bind-pmf\ p\ f) = fold-combine-plus\ (\lambda x. Mapping.map-values\ (\lambda-. (*)\ (pmf\ p\ x))\ (mapping-of-pmf\ (f\ x)))\ (set-pmf\ p)$
 ⟨proof⟩

lift-definition *bind-pmf-aux* :: $'a\ pmf \Rightarrow ('a \Rightarrow 'b\ pmf) \Rightarrow 'a\ set \Rightarrow ('b, real)$
mapping is
 $\lambda(p :: 'a\ pmf)\ (f :: 'a \Rightarrow 'b\ pmf)\ (A :: 'a\ set)\ (x :: 'b).$
 if $x \in (\bigcup_{y \in A}. set-pmf\ (f\ y))$ then
 $Some\ (measure-pmf.expectation\ p\ (\lambda y. indicator\ A\ y * pmf\ (f\ y)\ x))$
 else $None$ ⟨proof⟩

lemma *keys-bind-pmf-aux* [simp]:
 $Mapping.keys\ (bind-pmf-aux\ p\ f\ A) = (\bigcup_{x \in A}. set-pmf\ (f\ x))$
 ⟨proof⟩

lemma *lookup-default-bind-pmf-aux*:
 $Mapping.lookup-default\ 0\ (bind-pmf-aux\ p\ f\ A)\ x =$
 (if $x \in (\bigcup_{y \in A}. set-pmf\ (f\ y))$ then
 $measure-pmf.expectation\ p\ (\lambda y. indicator\ A\ y * pmf\ (f\ y)\ x)$ else 0)
 ⟨proof⟩

lemma *lookup-default-bind-pmf-aux'* [simp]:
 $Mapping.lookup-default\ 0\ (bind-pmf-aux\ p\ f\ (set-pmf\ p))\ x = pmf\ (bind-pmf\ p\ f)\ x$
 ⟨proof⟩

lemma *bind-pmf-aux-correct*:
 $mapping-of-pmf\ (bind-pmf\ p\ f) = bind-pmf-aux\ p\ f\ (set-pmf\ p)$
 ⟨proof⟩

lemma *bind-pmf-aux-code-aux*:
assumes $finite\ A$
shows $bind-pmf-aux\ p\ f\ A = fold-combine-plus\ (\lambda x. Mapping.map-values\ (\lambda-. (*)\ (pmf\ p\ x))\ (mapping-of-pmf\ (f\ x)))\ A\ (is\ ?lhs = ?rhs)$
 ⟨proof⟩

lemma *bind-pmf-aux-code* [code]:
 $\text{bind-pmf-aux } p \ f \ (\text{set } xs) =$
 $\text{fold-combine-plus } (\lambda x. \text{Mapping.map-values } (\lambda -. \ (*)) \ (\text{pmf } p \ x))$
 $\quad (\text{mapping-of-pmf } (f \ x)) \ (\text{set } xs)$
 ⟨proof⟩

lemmas *bind-pmf-code* [code abstract] = *bind-pmf-aux-correct*

end

hide-const (**open**) *fold-combine-plus*

lift-definition *cond-pmf-impl* :: 'a pmf \Rightarrow 'a set \Rightarrow ('a, real) mapping option **is**
 $\lambda p \ A. \text{if } A \cap \text{set-pmf } p = \{\}$ then None else
 Some $(\lambda x. \text{if } x \in A \cap \text{set-pmf } p \text{ then Some } (\text{pmf } p \ x \ / \ \text{measure-pmf.prob } p \ A)$
 else None) ⟨proof⟩

lemma *cond-pmf-impl-code-alt*:
assumes *finite A*
shows $\text{cond-pmf-impl } p \ A =$
 $\text{let } C = A \cap \text{set-pmf } p;$
 $\text{prob} = (\sum_{x \in C}. \text{pmf } p \ x)$
 $\text{in if prob} = 0 \text{ then}$
 None
 else
 Some $(\text{Mapping.map-values } (\lambda y. y \ / \ \text{prob})$
 $\quad (\text{Mapping.filter } (\lambda k -. k \in C) \ (\text{mapping-of-pmf } p)))$
 ⟨proof⟩

lemma *cond-pmf-impl-code* [code]:
 $\text{cond-pmf-impl } p \ (\text{set } xs) =$
 $\text{let } C = \text{set } xs \cap \text{set-pmf } p;$
 $\text{prob} = (\sum_{x \in C}. \text{pmf } p \ x)$
 $\text{in if prob} = 0 \text{ then}$
 None
 else
 Some $(\text{Mapping.map-values } (\lambda y. y \ / \ \text{prob})$
 $\quad (\text{Mapping.filter } (\lambda k -. k \in C) \ (\text{mapping-of-pmf } p)))$
 ⟨proof⟩

lemma *cond-pmf-code* [code abstract]:
 $\text{mapping-of-pmf } (\text{cond-pmf } p \ A) =$
 $(\text{case cond-pmf-impl } p \ A \text{ of}$
 $\quad \text{None} \Rightarrow \text{Code.abort } (\text{STR "cond-pmf with set of probability 0"})$
 $\quad \quad (\lambda -. \text{mapping-of-pmf } (\text{cond-pmf } p \ A))$
 $\quad | \text{Some } m \Rightarrow m)$
 ⟨proof⟩

lemma *binomial-pmf-code* [code abstract]:
 $\text{mapping-of-pmf } (\text{binomial-pmf } n \ p) =$
 if $p < 0 \vee p > 1$ then
 $\text{Code.abort } (\text{STR } \text{"binomial-pmf with invalid probability"})$
 $(\lambda_. \text{mapping-of-pmf } (\text{binomial-pmf } n \ p))$
 else if $p = 0$ then $\text{Mapping.update } 0 \ 1 \ \text{Mapping.empty}$
 else if $p = 1$ then $\text{Mapping.update } n \ 1 \ \text{Mapping.empty}$
 else $\text{Mapping.tabulate } [0..<\text{Suc } n] \ (\lambda k. \text{real } (n \text{ choose } k) * p^k * (1 - p)^{(n - k)})$
 <proof>

lemma *pred-pmf-code* [code]:
 $\text{pred-pmf } P \ p = (\forall x \in \text{set-pmf } p. \ P \ x)$
 <proof>

lemma *mapping-of-pmf-pmf-of-list*:
assumes $\bigwedge x. x \in \text{snd } \text{'set } xs \implies x > 0 \text{ sum-list } (\text{map } \text{snd } xs) = 1$
shows $\text{mapping-of-pmf } (\text{pmf-of-list } xs) =$
 $\text{Mapping.tabulate } (\text{remdups } (\text{map } \text{fst } xs))$
 $(\lambda x. \text{sum-list } (\text{map } \text{snd } (\text{filter } (\lambda z. \text{fst } z = x) \ xs))))$
 <proof>

lemma *mapping-of-pmf-pmf-of-list'*:
assumes $\text{pmf-of-list-wf } xs$
defines $xs' \equiv \text{filter } (\lambda z. \text{snd } z \neq 0) \ xs$
shows $\text{mapping-of-pmf } (\text{pmf-of-list } xs) =$
 $\text{Mapping.tabulate } (\text{remdups } (\text{map } \text{fst } xs'))$
 $(\lambda x. \text{sum-list } (\text{map } \text{snd } (\text{filter } (\lambda z. \text{fst } z = x) \ xs')))$ (is - = ?rhs)
 <proof>

lemma *pmf-of-list-wf-code* [code]:
 $\text{pmf-of-list-wf } xs \longleftrightarrow \text{list-all } (\lambda z. \text{snd } z \geq 0) \ xs \wedge \text{sum-list } (\text{map } \text{snd } xs) = 1$
 <proof>

lemma *pmf-of-list-code* [code abstract]:
 $\text{mapping-of-pmf } (\text{pmf-of-list } xs) =$
 if $\text{pmf-of-list-wf } xs$ then
 let $xs' = \text{filter } (\lambda z. \text{snd } z \neq 0) \ xs$
 in $\text{Mapping.tabulate } (\text{remdups } (\text{map } \text{fst } xs'))$
 $(\lambda x. \text{sum-list } (\text{map } \text{snd } (\text{filter } (\lambda z. \text{fst } z = x) \ xs')))$
 else
 $\text{Code.abort } (\text{STR } \text{"Invalid list for pmf-of-list"}) \ (\lambda_. \text{mapping-of-pmf } (\text{pmf-of-list } xs))$
 <proof>

lemma *mapping-of-pmf-eq-iff* [simp]:

$\text{mapping-of-pmf } p = \text{mapping-of-pmf } q \longleftrightarrow p = (q :: 'a \text{ pmf})$
 $\langle \text{proof} \rangle$

21.2 Code abbreviations for integrals and probabilities

Integrals and probabilities are defined for general measures, so we cannot give any code equations directly. We can, however, specialise these constants them to PMFs, give code equations for these specialised constants, and tell the code generator to unfold the original constants to the specialised ones whenever possible.

definition *pmf-integral* **where**

$\text{pmf-integral } p \ f = \text{lebesgue-integral } (\text{measure-pmf } p) \ (f :: - \Rightarrow \text{real})$

definition *pmf-set-integral* **where**

$\text{pmf-set-integral } p \ f \ A = \text{lebesgue-integral } (\text{measure-pmf } p) \ (\lambda x. \text{indicator } A \ x * f \ x :: \text{real})$

definition *pmf-prob* **where**

$\text{pmf-prob } p \ A = \text{measure-pmf.prob } p \ A$

lemma *pmf-prob-compl*: $\text{pmf-prob } p \ (-A) = 1 - \text{pmf-prob } p \ A$
 $\langle \text{proof} \rangle$

lemma *pmf-integral-pmf-set-integral* [code]:

$\text{pmf-integral } p \ f = \text{pmf-set-integral } p \ f \ (\text{set-pmf } p)$
 $\langle \text{proof} \rangle$

lemma *pmf-prob-pmf-set-integral*:

$\text{pmf-prob } p \ A = \text{pmf-set-integral } p \ (\lambda -. 1) \ A$
 $\langle \text{proof} \rangle$

lemma *pmf-set-integral-code-alt-finite*:

$\text{finite } A \implies \text{pmf-set-integral } p \ f \ A = (\sum x \in A. \text{pmf } p \ x * f \ x)$
 $\langle \text{proof} \rangle$

lemma *pmf-set-integral-code* [code]:

$\text{pmf-set-integral } p \ f \ (\text{set } xs) = (\sum x \in \text{set } xs. \text{pmf } p \ x * f \ x)$
 $\langle \text{proof} \rangle$

lemma *pmf-prob-code-alt-finite*:

$\text{finite } A \implies \text{pmf-prob } p \ A = (\sum x \in A. \text{pmf } p \ x)$
 $\langle \text{proof} \rangle$

lemma *pmf-prob-code* [code]:

$\text{pmf-prob } p \ (\text{set } xs) = (\sum x \in \text{set } xs. \text{pmf } p \ x)$
 $\text{pmf-prob } p \ (\text{List.coset } xs) = 1 - (\sum x \in \text{set } xs. \text{pmf } p \ x)$
 $\langle \text{proof} \rangle$

lemma *pmf-prob-code-unfold* [*code-abbrev*]: *pmf-prob* *p* = *measure-pmf.prob* *p*
 ⟨*proof*⟩

lemma *pmf-integral-code-unfold* [*code-abbrev*]: *pmf-integral* *p* = *measure-pmf.expectation* *p*
 ⟨*proof*⟩

definition *pmf-of-alist* *xs* = *embed-pmf* ($\lambda x. \text{case map-of } xs \text{ } x \text{ of } \text{Some } p \Rightarrow p \mid \text{None} \Rightarrow ()$)

lemma *pmf-of-mapping-Mapping* [*code-post*]:
pmf-of-mapping (*Mapping* *xs*) = *pmf-of-alist* *xs*
 ⟨*proof*⟩

instantiation *pmf* :: (*equal*) *equal*
begin

definition *equal-pmf* *p* *q* = (*mapping-of-pmf* *p* = *mapping-of-pmf* (*q* :: 'a *pmf*))

instance ⟨*proof*⟩
end

definition *single* :: 'a \Rightarrow 'a *multiset* **where**
single *s* = {#*s*#}

instantiation *pmf* :: (*random*) *random*
begin

context
includes *state-combinator-syntax* **and** *term-syntax*
begin

definition
pmfify :: ('b::typerep *multiset* \times (*unit* \Rightarrow *Code-Evaluation.term*)) \Rightarrow
 'b \times (*unit* \Rightarrow *Code-Evaluation.term*) \Rightarrow
 'b *pmf* \times (*unit* \Rightarrow *Code-Evaluation.term*) **where**
 [*code-unfold*]: *pmfify* *A* *x* =
Code-Evaluation.valtermify *pmf-of-multiset* {·}
 (*Code-Evaluation.valtermify* (+) {·} *A* {·})
 (*Code-Evaluation.valtermify* *single* {·} *x*)

definition
Quickcheck-Random.random *i* =

```

    Quickcheck-Random.random i  $\circ \rightarrow$  ( $\lambda A.$ 
      Quickcheck-Random.random i  $\circ \rightarrow$  ( $\lambda x.$  Pair (pmfify A x)))

instance <proof>

end

end

instantiation pmf :: (full-exhaustive) full-exhaustive
begin

definition full-exhaustive-pmf :: ('a pmf  $\times$  (unit  $\Rightarrow$  term)  $\Rightarrow$  (bool  $\times$  term list)
option)  $\Rightarrow$  natural  $\Rightarrow$  (bool  $\times$  term list) option
where
  full-exhaustive-pmf f i =
    Quickcheck-Exhaustive.full-exhaustive ( $\lambda A.$ 
      Quickcheck-Exhaustive.full-exhaustive ( $\lambda x.$  f (pmfify A x)) i) i

instance <proof>

end

end

```

22 Finite Maps

```

theory Fin-Map
  imports HOL-Analysis.Finite-Product-Measure HOL-Library.Finite-Map
begin

```

The *fmap* type can be instantiated to *polish-space*, needed for the proof of projective limit. *extensional* functions are used for the representation in order to stay close to the developments of (finite) products Pi_E and their sigma-algebra Pi_M .

```

type-notation fmap ( $\langle \langle \text{notation} = \langle \text{infix fmap} \rangle \rangle - \Rightarrow_F / - \rangle$  [22, 21] 21)

```

```

unbundle fmap.lifting

```

22.1 Domain and Application

```

lift-definition domain :: ('i  $\Rightarrow_F$  'a)  $\Rightarrow$  'i set is dom <proof>

```

```

lemma finite-domain[simp, intro]: finite (domain P)
  <proof>

```

```

lift-definition proj :: ('i  $\Rightarrow_F$  'a)  $\Rightarrow$  'i  $\Rightarrow$  'a
  ( $\langle \langle \text{indent} = 1 \text{ notation} = \langle \text{mixfix proj} \rangle \rangle '(-)_F \rangle$  [0] 1000) is

```

$\lambda f x. \text{ if } x \in \text{dom } f \text{ then the } (f x) \text{ else undefined } \langle \text{proof} \rangle$

declare $[[\text{coercion proj}]]$

lemma *extensional-proj* $[\text{simp}, \text{intro}]$: $(P)_F \in \text{extensional } (\text{domain } P)$
 $\langle \text{proof} \rangle$

lemma *proj-undefined* $[\text{simp}, \text{intro}]$: $i \notin \text{domain } P \implies P i = \text{undefined}$
 $\langle \text{proof} \rangle$

lemma *finmap-eq-iff*: $P = Q \iff (\text{domain } P = \text{domain } Q \wedge (\forall i \in \text{domain } P. P i = Q i))$
 $\langle \text{proof} \rangle$

22.2 Constructor of Finite Maps

lift-definition *finmap-of*: $'i \text{ set} \Rightarrow ('i \Rightarrow 'a) \Rightarrow ('i \Rightarrow_F 'a)$ **is**
 $\lambda I f x. \text{ if } x \in I \wedge \text{finite } I \text{ then Some } (f x) \text{ else None}$
 $\langle \text{proof} \rangle$

lemma *proj-finmap-of* $[\text{simp}]$:
assumes *finite inds*
shows $(\text{finmap-of } \text{inds } f)_F = \text{restrict } f \text{ inds}$
 $\langle \text{proof} \rangle$

lemma *domain-finmap-of* $[\text{simp}]$:
assumes *finite inds*
shows $\text{domain } (\text{finmap-of } \text{inds } f) = \text{inds}$
 $\langle \text{proof} \rangle$

lemma *finmap-of-eq-iff* $[\text{simp}]$:
assumes *finite i finite j*
shows $\text{finmap-of } i m = \text{finmap-of } j n \iff i = j \wedge (\forall k \in i. m k = n k)$
 $\langle \text{proof} \rangle$

lemma *finmap-of-inj-on-extensional-finite*:
assumes *finite K*
assumes $S \subseteq \text{extensional } K$
shows $\text{inj-on } (\text{finmap-of } K) S$
 $\langle \text{proof} \rangle$

22.3 Product set of Finite Maps

This is *Pi* for Finite Maps, most of this is copied

definition *Pi*': $'i \text{ set} \Rightarrow ('i \Rightarrow 'a \text{ set}) \Rightarrow ('i \Rightarrow_F 'a) \text{ set}$ **where**
 $Pi' I A = \{ P. \text{domain } P = I \wedge (\forall i. i \in I \longrightarrow (P)_F i \in A i) \}$

syntax
 $-Pi' :: [\text{pttrn}, 'a \text{ set}, 'b \text{ set}] \Rightarrow ('a \Rightarrow 'b) \text{ set}$

$\langle \langle \langle \text{indent}=3 \text{ notation}=\langle \text{binder } \Pi' \rangle \Pi'' \text{ } \neg \in \cdot / \cdot \rangle \rangle \rangle \quad 10)$
syntax-consts
 $\neg Pi' == Pi'$
translations
 $\Pi' x \in A. B == \text{CONST } Pi' A (\lambda x. B)$

22.3.1 Basic Properties of Pi'

lemma $Pi'-I[\text{intro}]$: $\text{domain } f = A \implies (\bigwedge x. x \in A \implies f x \in B x) \implies f \in Pi' A B$
 $\langle \text{proof} \rangle$

lemma $Pi'-I'[\text{simp}]$: $\text{domain } f = A \implies (\bigwedge x. x \in A \longrightarrow f x \in B x) \implies f \in Pi' A B$
 $\langle \text{proof} \rangle$

lemma $Pi'-\text{mem}$: $f \in Pi' A B \implies x \in A \implies f x \in B x$
 $\langle \text{proof} \rangle$

lemma $Pi'-\text{iff}$: $f \in Pi' I X \longleftrightarrow \text{domain } f = I \wedge (\forall i \in I. f i \in X i)$
 $\langle \text{proof} \rangle$

lemma $Pi'-E[\text{elim}]$:
 $f \in Pi' A B \implies (f x \in B x \implies \text{domain } f = A \implies Q) \implies (x \notin A \implies Q) \implies Q$
 $\langle \text{proof} \rangle$

lemma $\text{in-}Pi'-\text{cong}$:
 $\text{domain } f = \text{domain } g \implies (\bigwedge w. w \in A \implies f w = g w) \implies f \in Pi' A B \longleftrightarrow g \in Pi' A B$
 $\langle \text{proof} \rangle$

lemma $Pi'-\text{eq-empty}[\text{simp}]$:
assumes $\text{finite } A$ **shows** $(Pi' A B) = \{\} \longleftrightarrow (\exists x \in A. B x = \{\})$
 $\langle \text{proof} \rangle$

lemma $Pi'-\text{mono}$: $(\bigwedge x. x \in A \implies B x \subseteq C x) \implies Pi' A B \subseteq Pi' A C$
 $\langle \text{proof} \rangle$

lemma $Pi-Pi'$: $\text{finite } A \implies (Pi_E A B) = \text{proj } ' Pi' A B$
 $\langle \text{proof} \rangle$

22.4 Topological Space of Finite Maps

instantiation $\text{fmap} :: (\text{type}, \text{topological-space}) \text{topological-space}$
begin

definition $\text{open-fmap} :: ('a \Rightarrow_F 'b) \text{set} \Rightarrow \text{bool}$ **where**
 $[\text{code del}]$: $\text{open-fmap} = \text{generate-topology } \{Pi' a b \mid a \in b. \forall i \in a. \text{open } (b i)\}$

lemma *open-Pi'I*: $(\bigwedge i. i \in I \implies \text{open } (A \ i)) \implies \text{open } (Pi' \ I \ A)$
 $\langle \text{proof} \rangle$

instance $\langle \text{proof} \rangle$

end

lemma *open-restricted-space*:
shows $\text{open } \{m. P \ (\text{domain } m)\}$
 $\langle \text{proof} \rangle$

lemma *closed-restricted-space*:
shows $\text{closed } \{m. P \ (\text{domain } m)\}$
 $\langle \text{proof} \rangle$

lemma *tendsto-proj*: $((\lambda x. x) \longrightarrow a) \ F \implies ((\lambda x. (x)_F \ i) \longrightarrow (a)_F \ i) \ F$
 $\langle \text{proof} \rangle$

lemma *continuous-proj*:
shows *continuous-on* $s \ (\lambda x. (x)_F \ i)$
 $\langle \text{proof} \rangle$

instance *fmap* :: $(\text{type}, \text{first-countable-topology}) \ \text{first-countable-topology}$
 $\langle \text{proof} \rangle$

22.5 Metric Space of Finite Maps

instantiation *fmap* :: $(\text{type}, \text{metric-space}) \ \text{dist}$
begin

definition *dist-fmap* **where**
 $\text{dist } P \ Q = \text{Max } (\text{range } (\lambda i. \text{dist } ((P)_F \ i) ((Q)_F \ i))) + (\text{if } \text{domain } P = \text{domain } Q \text{ then } 0 \text{ else } 1)$

instance $\langle \text{proof} \rangle$
end

instantiation *fmap* :: $(\text{type}, \text{metric-space}) \ \text{uniformity-dist}$
begin

definition $[\text{code del}]$:
 $(\text{uniformity} :: ((\text{'a}, \text{'b}) \ \text{fmap} \times (\text{'a} \Rightarrow_F \text{'b})) \ \text{filter}) =$
 $(\text{INF } e \in \{0 < ..\}. \text{principal } \{(x, y). \text{dist } x \ y < e\})$

instance
 $\langle \text{proof} \rangle$
end

declare *uniformity-Absort* **[where** $\text{'a} = (\text{'a} \Rightarrow_F \text{'b} :: \text{metric-space}), \text{code}]$

instantiation *fmap* :: (*type*, *metric-space*) *metric-space*
begin

lemma *finite-proj-image'*: $x \notin \text{domain } P \implies \text{finite } ((P)_F \text{ ` } S)$
 ⟨*proof*⟩

lemma *finite-proj-image*: $\text{finite } ((P)_F \text{ ` } S)$
 ⟨*proof*⟩

lemma *finite-proj-diag*: $\text{finite } ((\lambda i. d ((P)_F i) ((Q)_F i)) \text{ ` } S)$
 ⟨*proof*⟩

lemma *dist-le-1-imp-domain-eq*:
 shows $\text{dist } P \ Q < 1 \implies \text{domain } P = \text{domain } Q$
 ⟨*proof*⟩

lemma *dist-proj*:
 shows $\text{dist } ((x)_F i) ((y)_F i) \leq \text{dist } x \ y$
 ⟨*proof*⟩

lemma *dist-fmap-lessI*:
 assumes $\text{domain } P = \text{domain } Q$
 assumes $0 < e$
 assumes $\bigwedge i. i \in \text{domain } P \implies \text{dist } (P \ i) (Q \ i) < e$
 shows $\text{dist } P \ Q < e$
 ⟨*proof*⟩

instance
 ⟨*proof*⟩

end

22.6 Complete Space of Finite Maps

lemma *tendsto-fmap*:
 fixes $f :: \text{nat} \Rightarrow ('i \Rightarrow_F ('a :: \text{metric-space}))$
 assumes *ind-f*: $\bigwedge n. \text{domain } (f \ n) = \text{domain } g$
 assumes *proj-g*: $\bigwedge i. i \in \text{domain } g \implies (\lambda n. (f \ n) \ i) \longrightarrow g \ i$
 shows $f \longrightarrow g$
 ⟨*proof*⟩

instance *fmap* :: (*type*, *complete-space*) *complete-space*
 ⟨*proof*⟩

22.7 Second Countable Space of Finite Maps

instantiation *fmap* :: (*countable*, *second-countable-topology*) *second-countable-topology*
begin

definition *basis-proj*::'b set set

where *basis-proj* = (SOME B. countable B \wedge topological-basis B)

lemma *countable-basis-proj*: countable *basis-proj* **and** *basis-proj*: topological-basis *basis-proj*

<proof>

definition *basis-finmap*::('a \Rightarrow_F 'b) set set

where *basis-finmap* = {Pi' I S | I S. finite I \wedge ($\forall i \in I$. S i \in *basis-proj*)}

lemma *in-basis-finmap*I:

assumes finite I **assumes** $\bigwedge i. i \in I \implies S i \in \text{basis-proj}$

shows Pi' I S \in *basis-finmap*

<proof>

lemma *basis-finmap-eq*:

assumes *basis-proj* $\neq \{\}$

shows *basis-finmap* = (λf . Pi' (domain f) (λi . from-nat-into *basis-proj* ((f)_F i))) ‘

(UNIV::('a \Rightarrow_F nat) set) (is - = ?f ‘ -)

<proof>

lemma *basis-finmap-eq-empty*: *basis-proj* = $\{\}$ \implies *basis-finmap* = {Pi' $\{\}$ undefined}

<proof>

lemma *countable-basis-finmap*: countable *basis-finmap*

<proof>

lemma *finmap-topological-basis*:

topological-basis *basis-finmap*

<proof>

lemma *range-enum-basis-finmap-imp-open*:

assumes $x \in \text{basis-finmap}$

shows open x

<proof>

instance *<proof>*

end

22.8 Polish Space of Finite Maps

instance *fmap* :: (countable, polish-space) polish-space *<proof>*

22.9 Product Measurable Space of Finite Maps

definition *PiF* I M \equiv

$\text{sigma } (\bigcup J \in I. (\Pi' j \in J. \text{space } (M j))) \{(\Pi' j \in J. X j) \mid X J. J \in I \wedge X \in (\Pi j \in J. \text{sets } (M j))\}$

abbreviation

$Pi_F I M \equiv PiF I M$

syntax

$-PiF :: \text{pttrn} \Rightarrow 'i \text{ set} \Rightarrow 'a \text{ measure} \Rightarrow ('i \Rightarrow 'a) \text{ measure}$
 $(\langle \langle \text{indent}=3 \text{ notation}=\text{binder } \Pi_F \rangle \rangle \Pi_F -\in -./ -) \rangle 10)$

syntax-consts

$-PiF == PiF$

translations

$\Pi_F x \in I. M == \text{CONST } PiF I (\%x. M)$

lemma $PiF\text{-gen-subset}$: $\{(\Pi' j \in J. X j) \mid X J. J \in I \wedge X \in (\Pi j \in J. \text{sets } (M j))\}$

\subseteq

$\text{Pow } (\bigcup J \in I. (\Pi' j \in J. \text{space } (M j)))$
 $\langle \text{proof} \rangle$

lemma $\text{space-}PiF$: $\text{space } (PiF I M) = (\bigcup J \in I. (\Pi' j \in J. \text{space } (M j)))$

$\langle \text{proof} \rangle$

lemma $\text{sets-}PiF$:

$\text{sets } (PiF I M) = \text{sigma-sets } (\bigcup J \in I. (\Pi' j \in J. \text{space } (M j)))$
 $\{(\Pi' j \in J. X j) \mid X J. J \in I \wedge X \in (\Pi j \in J. \text{sets } (M j))\}$
 $\langle \text{proof} \rangle$

lemma $\text{sets-}PiF\text{-singleton}$:

$\text{sets } (PiF \{I\} M) = \text{sigma-sets } (\Pi' j \in I. \text{space } (M j))$
 $\{(\Pi' j \in I. X j) \mid X. X \in (\Pi j \in I. \text{sets } (M j))\}$
 $\langle \text{proof} \rangle$

lemma $\text{in-sets-}PiFI$:

assumes $X = (Pi' J S) J \in I \bigwedge i. i \in J \implies S i \in \text{sets } (M i)$
shows $X \in \text{sets } (PiF I M)$
 $\langle \text{proof} \rangle$

lemma $\text{product-in-sets-}PiFI$:

assumes $J \in I \bigwedge i. i \in J \implies S i \in \text{sets } (M i)$
shows $(Pi' J S) \in \text{sets } (PiF I M)$
 $\langle \text{proof} \rangle$

lemma $\text{singleton-space-subset-in-sets}$:

fixes J
assumes $J \in I$
assumes $\text{finite } J$
shows $\text{space } (PiF \{J\} M) \in \text{sets } (PiF I M)$
 $\langle \text{proof} \rangle$

lemma *singleton-subspace-set-in-sets:*

assumes $A: A \in \text{sets } (\text{PiF } \{J\} M)$

assumes *finite J*

assumes $J \in I$

shows $A \in \text{sets } (\text{PiF } I M)$

<proof>

lemma *finite-measurable-singletonI:*

assumes *finite I*

assumes $\bigwedge J. J \in I \implies \text{finite } J$

assumes $MN: \bigwedge J. J \in I \implies A \in \text{measurable } (\text{PiF } \{J\} M) N$

shows $A \in \text{measurable } (\text{PiF } I M) N$

<proof>

lemma *countable-finite-comprehension:*

fixes $f :: 'a::\text{countable set} \Rightarrow -$

assumes $\bigwedge s. P s \implies \text{finite } s$

assumes $\bigwedge s. P s \implies f s \in \text{sets } M$

shows $\bigcup \{f s \mid s. P s\} \in \text{sets } M$

<proof>

lemma *space-subset-in-sets:*

fixes $J::'a::\text{countable set set}$

assumes $J \subseteq I$

assumes $\bigwedge j. j \in J \implies \text{finite } j$

shows $\text{space } (\text{PiF } J M) \in \text{sets } (\text{PiF } I M)$

<proof>

lemma *subspace-set-in-sets:*

fixes $J::'a::\text{countable set set}$

assumes $A: A \in \text{sets } (\text{PiF } J M)$

assumes $J \subseteq I$

assumes $\bigwedge j. j \in J \implies \text{finite } j$

shows $A \in \text{sets } (\text{PiF } I M)$

<proof>

lemma *countable-measurable-PiFI:*

fixes $I::'a::\text{countable set set}$

assumes $MN: \bigwedge J. J \in I \implies \text{finite } J \implies A \in \text{measurable } (\text{PiF } \{J\} M) N$

shows $A \in \text{measurable } (\text{PiF } I M) N$

<proof>

lemma *measurable-PiF:*

assumes $f: \bigwedge x. x \in \text{space } N \implies \text{domain } (f x) \in I \wedge (\forall i \in \text{domain } (f x). (f x) i \in \text{space } (M i))$

assumes $S: \bigwedge J S. J \in I \implies (\bigwedge i. i \in J \implies S i \in \text{sets } (M i)) \implies$

$f - '(\text{Pi } J S) \cap \text{space } N \in \text{sets } N$

shows $f \in \text{measurable } N (\text{PiF } I M)$

<proof>

lemma *restrict-sets-measurable*:

assumes $A: A \in \text{sets } (PiF\ I\ M)$ **and** $J \subseteq I$
shows $A \cap \{m. \text{domain } m \in J\} \in \text{sets } (PiF\ J\ M)$
 $\langle \text{proof} \rangle$

lemma *measurable-finmap-of*:

assumes $f: \bigwedge i. (\exists x \in \text{space } N. i \in J\ x) \implies (\lambda x. f\ x\ i) \in \text{measurable } N\ (M\ i)$
assumes $J: \bigwedge x. x \in \text{space } N \implies J\ x \in I \ \bigwedge x. x \in \text{space } N \implies \text{finite } (J\ x)$
assumes $JN: \bigwedge S. \{x. J\ x = S\} \cap \text{space } N \in \text{sets } N$
shows $(\lambda x. \text{finmap-of } (J\ x)\ (f\ x)) \in \text{measurable } N\ (PiF\ I\ M)$
 $\langle \text{proof} \rangle$

lemma *measurable-PiM-finmap-of*:

assumes $\text{finite } J$
shows $\text{finmap-of } J \in \text{measurable } (PiM\ J\ M)\ (PiF\ \{J\}\ M)$
 $\langle \text{proof} \rangle$

lemma *proj-measurable-singleton*:

assumes $A \in \text{sets } (M\ i)$
shows $(\lambda x. (x)_F\ i) - 'A \cap \text{space } (PiF\ \{I\}\ M) \in \text{sets } (PiF\ \{I\}\ M)$
 $\langle \text{proof} \rangle$

lemma *measurable-proj-singleton*:

assumes $i \in I$
shows $(\lambda x. (x)_F\ i) \in \text{measurable } (PiF\ \{I\}\ M)\ (M\ i)$
 $\langle \text{proof} \rangle$

lemma *measurable-proj-countable*:

fixes $I::'a::\text{countable set set}$
assumes $y \in \text{space } (M\ i)$
shows $(\lambda x. \text{if } i \in \text{domain } x \text{ then } (x)_F\ i \text{ else } y) \in \text{measurable } (PiF\ I\ M)\ (M\ i)$
 $\langle \text{proof} \rangle$

lemma *measurable-restrict-proj*:

assumes $J \in II\ \text{finite } J$
shows $\text{finmap-of } J \in \text{measurable } (PiM\ J\ M)\ (PiF\ II\ M)$
 $\langle \text{proof} \rangle$

lemma *measurable-proj-PiM*:

fixes $J\ K::'a::\text{countable set set}$ **and** $I::'a\ \text{set set}$
assumes $\text{finite } J\ J \in I$
assumes $x \in \text{space } (PiM\ J\ M)$
shows $\text{proj} \in \text{measurable } (PiF\ \{J\}\ M)\ (PiM\ J\ M)$
 $\langle \text{proof} \rangle$

lemma *space-PiF-singleton-eq-product*:

assumes $\text{finite } I$
shows $\text{space } (PiF\ \{I\}\ M) = (\Pi' i \in I. \text{space } (M\ i))$

$\langle \text{proof} \rangle$

adapted from $\text{sets } (Pi_M \ ?I \ ?M) = \text{sigma-sets } (\Pi_E \ i \in ?I. \text{space } (?M \ i)) \ \{\{f \in \Pi_E \ i \in ?I. \text{space } (?M \ i). \ f \ i \in A\} \mid i \ A. \ i \in ?I \wedge A \in \text{sets } (?M \ i)\}$

lemma *sets-PiF-single*:

assumes *finite* $I \ I \neq \{\}$

shows $\text{sets } (PiF \ \{I\} \ M) =$

$\text{sigma-sets } (\Pi' \ i \in I. \text{space } (M \ i))$

$\{\{f \in \Pi' \ i \in I. \text{space } (M \ i). \ f \ i \in A\} \mid i \ A. \ i \in I \wedge A \in \text{sets } (M \ i)\}$

(**is** $= \text{sigma-sets } ?\Omega \ ?R$)

$\langle \text{proof} \rangle$

adapted from $(\bigwedge i. \ i \in ?I \implies ?A \ i = ?B \ i) \implies Pi_E \ ?I \ ?A = Pi_E \ ?I \ ?B$

lemma *Pi'-cong*:

assumes *finite* I

assumes $\bigwedge i. \ i \in I \implies f \ i = g \ i$

shows $Pi' \ I \ f = Pi' \ I \ g$

$\langle \text{proof} \rangle$

adapted from $\llbracket \text{finite } ?I; \bigwedge i \ n \ m. \ \llbracket i \in ?I; \ n \leq m \rrbracket \implies ?A \ n \ i \subseteq ?A \ m \ i \rrbracket \implies (\bigcup_n \ Pi \ ?I \ (?A \ n)) = (\Pi \ i \in ?I. \bigcup_n \ ?A \ n \ i)$

lemma *Pi'-UN*:

fixes $A :: \text{nat} \Rightarrow 'i \Rightarrow 'a \text{ set}$

assumes *finite* I

assumes *mono*: $\bigwedge i \ n \ m. \ i \in I \implies n \leq m \implies A \ n \ i \subseteq A \ m \ i$

shows $(\bigcup n. \ Pi' \ I \ (A \ n)) = Pi' \ I \ (\lambda i. \bigcup n. \ A \ n \ i)$

$\langle \text{proof} \rangle$

adapted from $\llbracket \bigwedge i. \ i \in ?I \implies \exists S \subseteq ?E \ i. \ \text{countable } S \wedge ?\Omega \ i = \bigcup S; \bigwedge i. \ i \in ?I \implies ?E \ i \subseteq \text{Pow } (? \Omega \ i); \bigwedge j. \ j \in ?J \implies \text{finite } j; \bigcup ?J = ?I \rrbracket \implies \text{sets } (Pi_M \ ?I \ (\lambda i. \text{sigma } (? \Omega \ i) \ (?E \ i))) = \text{sets } (\text{sigma } (Pi_E \ ?I \ ?\Omega) \ \{\{f \in Pi_E \ ?I \ ?\Omega. \ \forall i \in j. \ f \ i \in A \ i\} \mid A \ j. \ j \in ?J \wedge A \in Pi \ j \ ?E\})$

lemma *sigma-fprod-algebra-sigma-eq*:

fixes $E :: 'i \Rightarrow 'a \text{ set set}$ **and** $S :: 'i \Rightarrow \text{nat} \Rightarrow 'a \text{ set}$

assumes [*simp*]: *finite* $I \ I \neq \{\}$

and *S-union*: $\bigwedge i. \ i \in I \implies (\bigcup j. \ S \ i \ j) = \text{space } (M \ i)$

and *S-in-E*: $\bigwedge i. \ i \in I \implies \text{range } (S \ i) \subseteq E \ i$

assumes *E-closed*: $\bigwedge i. \ i \in I \implies E \ i \subseteq \text{Pow } (\text{space } (M \ i))$

and *E-generates*: $\bigwedge i. \ i \in I \implies \text{sets } (M \ i) = \text{sigma-sets } (\text{space } (M \ i)) \ (E \ i)$

defines $P == \{ \ Pi' \ I \ F \mid F. \ \forall i \in I. \ F \ i \in E \ i \}$

shows $\text{sets } (PiF \ \{I\} \ M) = \text{sigma-sets } (\text{space } (PiF \ \{I\} \ M)) \ P$

$\langle \text{proof} \rangle$

lemma *product-open-generates-sets-PiF-single*:

assumes $I \neq \{\}$

assumes [*simp*]: *finite* I

shows $\text{sets } (PiF \ \{I\} \ (\lambda -. \text{borel} :: 'b :: \text{second-countable-topology measure})) =$

sigma-sets (*space* (*PiF* {*I*} ($\lambda\cdot$. *borel*))) {*Pi'* *I* *F* | *F*. ($\forall i \in I$. *F* *i* \in *Collect open*)}

<proof>

lemma *finmap-UNIV[simp]*: ($\bigcup J \in \text{Collect finite. } \Pi' j \in J. \text{ UNIV}$) = *UNIV* *<proof>*

lemma *borel-eq-PiF-borel*:

shows (*borel* :: (*'i*::countable \Rightarrow_F *'a*::polish-space) *measure*) =
PiF (*Collect finite*) ($\lambda\cdot$. *borel* :: *'a measure*)
<proof>

22.10 Isomorphism between Functions and Finite Maps

lemma *measurable-finmap-compose*:

shows ($\lambda m. \text{compose } J \text{ m } f$) \in *measurable* (*PiM* (*f* ‘ *J*) ($\lambda\cdot$. *M*)) (*PiM* *J* ($\lambda\cdot$. *M*))
<proof>

lemma *measurable-compose-inv*:

assumes *inj*: $\bigwedge j. j \in J \Rightarrow f' (f j) = j$
shows ($\lambda m. \text{compose } (f' \text{ ' } J) \text{ m } f'$) \in *measurable* (*PiM* *J* ($\lambda\cdot$. *M*)) (*PiM* (*f* ‘ *J*) ($\lambda\cdot$. *M*))
<proof>

locale *function-to-finmap* =

fixes *J*::*'a set* **and** *f* :: *'a* \Rightarrow *'b*::countable **and** *f'*
assumes [*simp*]: *finite J*
assumes *inv*: $i \in J \Rightarrow f' (f i) = i$

begin

to measure finmaps

definition *fm* = (*finmap-of* (*f* ‘ *J*)) *o* ($\lambda g. \text{compose } (f' \text{ ' } J) \text{ g } f'$)

lemma *domain-fm[simp]*: *domain* (*fm* *x*) = *f* ‘ *J*
<proof>

lemma *fm-restrict[simp]*: *fm* (*restrict y J*) = *fm y*
<proof>

lemma *fm-product*:

assumes $\bigwedge i. \text{space } (M i) = \text{UNIV}$
shows *fm* – ‘ *Pi'* (*f* ‘ *J*) *S* \cap *space* (*Pi_M* *J* *M*) = ($\Pi_E j \in J. S (f j)$)
<proof>

lemma *fm-measurable*:

assumes *f* ‘ *J* $\in N$
shows *fm* \in *measurable* (*Pi_M* *J* ($\lambda\cdot$. *M*)) (*Pi_F* *N* ($\lambda\cdot$. *M*))
<proof>

lemma *proj-fm*:
assumes $x \in J$
shows $fm\ m\ (f\ x) = m\ x$
 $\langle proof \rangle$

lemma *inj-on-compose-f'*: $inj\text{-}on\ (\lambda g.\ compose\ (f\ 'J)\ g\ f')\ (extensional\ J)$
 $\langle proof \rangle$

lemma *inj-on-fm*:
assumes $\bigwedge i.\ space\ (M\ i) = UNIV$
shows $inj\text{-}on\ fm\ (space\ (Pi_M\ J\ M))$
 $\langle proof \rangle$

to measure functions

definition $mf = (\lambda g.\ compose\ J\ g\ f)\ o\ proj$

lemma *mf-fm*:
assumes $x \in space\ (Pi_M\ J\ (\lambda\cdot.\ M))$
shows $mf\ (fm\ x) = x$
 $\langle proof \rangle$

lemma *mf-measurable*:
assumes $space\ M = UNIV$
shows $mf \in measurable\ (PiF\ \{f\ 'J\}\ (\lambda\cdot.\ M))\ (PiM\ J\ (\lambda\cdot.\ M))$
 $\langle proof \rangle$

lemma *fm-image-measurable*:
assumes $space\ M = UNIV$
assumes $X \in sets\ (Pi_M\ J\ (\lambda\cdot.\ M))$
shows $fm\ 'X \in sets\ (PiF\ \{f\ 'J\}\ (\lambda\cdot.\ M))$
 $\langle proof \rangle$

lemma *fm-image-measurable-finite*:
assumes $space\ M = UNIV$
assumes $X \in sets\ (Pi_M\ J\ (\lambda\cdot.\ M::'c\ measure))$
shows $fm\ 'X \in sets\ (PiF\ (Collect\ finite)\ (\lambda\cdot.\ M::'c\ measure))$
 $\langle proof \rangle$

measure on finmaps

definition $mapmeasure\ M\ N = distr\ M\ (PiF\ (Collect\ finite)\ N)\ (fm)$

lemma *sets-mapmeasure[simp]*: $sets\ (mapmeasure\ M\ N) = sets\ (PiF\ (Collect\ finite)\ N)$
 $\langle proof \rangle$

lemma *space-mapmeasure[simp]*: $space\ (mapmeasure\ M\ N) = space\ (PiF\ (Collect\ finite)\ N)$
 $\langle proof \rangle$

```

lemma mapmeasure-PiF:
  assumes s1: space M = space (PiM J ( $\lambda$ -. N))
  assumes s2: sets M = sets (PiM J ( $\lambda$ -. N))
  assumes space N = UNIV
  assumes X  $\in$  sets (PiF (Collect finite) ( $\lambda$ -. N))
  shows emeasure (mapmeasure M ( $\lambda$ -. N)) X = emeasure M ((fm -‘ X  $\cap$  extensional J))
  <proof>

lemma mapmeasure-PiM:
  fixes N::'c measure
  assumes s1: space M = space (PiM J ( $\lambda$ -. N))
  assumes s2: sets M = (PiM J ( $\lambda$ -. N))
  assumes N: space N = UNIV
  assumes X: X  $\in$  sets M
  shows emeasure M X = emeasure (mapmeasure M ( $\lambda$ -. N)) (fm -‘ X)
  <proof>

end

end

```

23 Projective Limit

```

theory Projective-Limit
imports
  Fin-Map
  Infinite-Product-Measure
  HOL-Library.Diagonal-Subsequence
begin

```

23.1 Sequences of Finite Maps in Compact Sets

```

locale finmap-seqs-into-compact =
  fixes K::nat  $\Rightarrow$  (nat  $\Rightarrow_F$  'a::metric-space) set and f::nat  $\Rightarrow$  (nat  $\Rightarrow_F$  'a) and
  M
  assumes compact:  $\bigwedge n.$  compact (K n)
  assumes f-in-K:  $\bigwedge n.$  K n  $\neq$  {}
  assumes domain-K:  $\bigwedge n.$  k  $\in$  K n  $\implies$  domain k = domain (f n)
  assumes proj-in-K:
     $\bigwedge t$  n m. m  $\geq$  n  $\implies$  t  $\in$  domain (f n)  $\implies$  (f m)F t  $\in$  ( $\lambda k.$  (k)F t) -‘ K n
begin

lemma proj-in-K': ( $\exists n.$   $\forall m \geq n.$  (f m)F t  $\in$  ( $\lambda k.$  (k)F t) -‘ K n)
  <proof>

lemma proj-in-KE:
  obtains n where  $\bigwedge m.$  m  $\geq$  n  $\implies$  (f m)F t  $\in$  ( $\lambda k.$  (k)F t) -‘ K n
  <proof>

```

lemma *compact-projset*:
shows *compact* $((\lambda k. (k)_F i) \cdot K n)$
 $\langle proof \rangle$

end

lemma *compactE'*:
fixes $S :: 'a :: \text{metric-space set}$
assumes *compact* $S \ \forall n \geq m. f \ n \in S$
obtains $l \ r$ **where** $l \in S$ *strict-mono* $(r :: \text{nat} \Rightarrow \text{nat})$ $((f \circ r) \longrightarrow l)$ *sequentially*
 $\langle proof \rangle$

sublocale *finmap-seqs-into-compact* \subseteq *subseqs* $\lambda n \ s. (\exists l. (\lambda i. ((f \circ s) i)_F n) \longrightarrow l)$
 $\langle proof \rangle$

lemma **(in** *finmap-seqs-into-compact*) *diagonal-tendsto*: $\exists l. (\lambda i. (f \ (diagseq \ i))_F n) \longrightarrow l$
 $\langle proof \rangle$

23.2 Daniell-Kolmogorov Theorem

Existence of Projective Limit

locale *polish-projective* = *projective-family* $I \ P \ \lambda -. \text{ borel} :: 'a :: \text{polish-space measure}$
for $I :: 'i \text{ set}$ **and** P
begin

lemma *emeasure-lim-emb*:
assumes $X: J \subseteq I$ *finite* $J \ X \in \text{sets } (\Pi_M \ i \in J. \text{ borel})$
shows $\lim (\text{emb } I \ J \ X) = P \ J \ X$
 $\langle proof \rangle$

lemma *measure-lim-emb*:
 $J \subseteq I \implies \text{finite } J \implies X \in \text{sets } (\Pi_M \ i \in J. \text{ borel}) \implies \text{measure } \lim (\text{emb } I \ J \ X)$
 $= \text{measure } (P \ J) \ X$
 $\langle proof \rangle$

end

hide-const **(open)** Pi^F
hide-const **(open)** Pi_F
hide-const **(open)** Pi'
hide-const **(open)** *finmap-of*
hide-const **(open)** *proj*
hide-const **(open)** *domain*
hide-const **(open)** *basis-finmap*

sublocale *polish-projective* $\subseteq P$: *prob-space* *lim*

⟨proof⟩

locale *polish-product-prob-space* =
product-prob-space $\lambda\cdot$. *borel::('a::polish-space) measure I* **for** *I::'i set*

sublocale *polish-product-prob-space* \subseteq *P: polish-projective I* λJ . *PiM J* ($\lambda\cdot$. *borel::('a) measure*)
 ⟨proof⟩

lemma (**in** *polish-product-prob-space*) *limP-eq-PiM: lim = PiM I* ($\lambda\cdot$. *borel*)
 ⟨proof⟩

end

24 Random Permutations

theory *Random-Permutations*

imports

HOL-Combinatorics.Multiset-Permutations

Probability-Mass-Function

begin

Choosing a set permutation (i.e. a distinct list with the same elements as the set) uniformly at random is the same as first choosing the first element of the list and then choosing the rest of the list as a permutation of the remaining set.

lemma *random-permutation-of-set:*

assumes *finite A A* $\neq \{\}$
shows *pmf-of-set (permutations-of-set A) =*
do {
x \leftarrow *pmf-of-set A;*
xs \leftarrow *pmf-of-set (permutations-of-set (A - {x}));*
return-pmf (x#xs)
} (is ?lhs = ?rhs)

⟨proof⟩

A generic fold function that takes a function, an initial state, and a set and chooses a random order in which it then traverses the set in the same fashion as a left fold over a list. We first give a recursive definition.

function *fold-random-permutation* $:: ('a \Rightarrow 'b \Rightarrow 'b) \Rightarrow 'b \Rightarrow 'a \text{ set} \Rightarrow 'b \text{ pmf}$

where

fold-random-permutation f x $\{\}$ = *return-pmf x*
 $\mid \neg \text{finite } A \implies \text{fold-random-permutation } f \text{ } x \text{ } A = \text{return-pmf } x$
 $\mid \text{finite } A \implies A \neq \{\} \implies$
fold-random-permutation f x A =
pmf-of-set A $\gg (\lambda a. \text{fold-random-permutation } f \text{ } (f \text{ } a \text{ } x) \text{ } (A - \{a\}))$

⟨proof⟩

termination ⟨proof⟩

We can now show that the above recursive definition is equivalent to choosing a random set permutation and folding over it (in any direction).

lemma *fold-random-permutation-foldl*:

assumes *finite A*

shows $\text{fold-random-permutation } f \ x \ A =$
 $\text{map-pmf } (\text{foldl } (\lambda x \ y. f \ y \ x) \ x) \ (\text{pmf-of-set } (\text{permutations-of-set } A))$

<proof>

lemma *fold-random-permutation-foldr*:

assumes *finite A*

shows $\text{fold-random-permutation } f \ x \ A =$
 $\text{map-pmf } (\lambda xs. \text{foldr } f \ xs \ x) \ (\text{pmf-of-set } (\text{permutations-of-set } A))$

<proof>

lemma *fold-random-permutation-fold*:

assumes *finite A*

shows $\text{fold-random-permutation } f \ x \ A =$
 $\text{map-pmf } (\lambda xs. \text{fold } f \ xs \ x) \ (\text{pmf-of-set } (\text{permutations-of-set } A))$

<proof>

lemma *fold-random-permutation-code* [code]:

$\text{fold-random-permutation } f \ x \ (\text{set } xs) =$
 $\text{map-pmf } (\text{foldl } (\lambda x \ y. f \ y \ x) \ x) \ (\text{pmf-of-set } (\text{permutations-of-set } (\text{set } xs)))$

<proof>

We now introduce a slightly generalised version of the above fold operation that does not simply return the result in the end, but applies a monadic bind to it. This may seem somewhat arbitrary, but it is a common use case, e.g. in the Social Decision Scheme of Random Serial Dictatorship, where voters narrow down a set of possible winners in a random order and the winner is chosen from the remaining set uniformly at random.

function *fold-bind-random-permutation*

$:: ('a \Rightarrow 'b \Rightarrow 'c) \Rightarrow ('b \Rightarrow 'c \text{ pmf}) \Rightarrow 'b \Rightarrow 'a \text{ set} \Rightarrow 'c \text{ pmf}$ **where**

$\text{fold-bind-random-permutation } f \ g \ x \ \{\} = g \ x$

$| \neg \text{finite } A \Rightarrow \text{fold-bind-random-permutation } f \ g \ x \ A = g \ x$

$| \text{finite } A \Rightarrow A \neq \{\} \Rightarrow$

$\text{fold-bind-random-permutation } f \ g \ x \ A =$

$\text{pmf-of-set } A \gg (\lambda a. \text{fold-bind-random-permutation } f \ g \ (f \ a \ x) \ (A - \{a\}))$

<proof>

termination *<proof>*

We now show that the recursive definition is equivalent to a random fold followed by a monadic bind.

lemma *fold-bind-random-permutation-altdef* [code]:

$\text{fold-bind-random-permutation } f \ g \ x \ A = \text{fold-random-permutation } f \ x \ A \gg g$

<proof>

We can now derive the following nice monadic representations of the com-

binned fold-and-bind:

lemma *fold-bind-random-permutation-foldl*:

assumes *finite A*

shows *fold-bind-random-permutation f g x A =*

do {xs ← pmf-of-set (permutations-of-set A); g (foldl (λx y. f y x) x xs)}

<proof>

lemma *fold-bind-random-permutation-foldr*:

assumes *finite A*

shows *fold-bind-random-permutation f g x A =*

do {xs ← pmf-of-set (permutations-of-set A); g (foldr f xs x)}

<proof>

lemma *fold-bind-random-permutation-fold*:

assumes *finite A*

shows *fold-bind-random-permutation f g x A =*

do {xs ← pmf-of-set (permutations-of-set A); g (fold f xs x)}

<proof>

The following useful lemma allows us to swap partitioning a set w.r.t. a predicate and drawing a random permutation of that set.

lemma *partition-random-permutations*:

assumes *finite A*

shows *map-pmf (partition P) (pmf-of-set (permutations-of-set A)) =*

pair-pmf (pmf-of-set (permutations-of-set {x∈A. P x}))

(pmf-of-set (permutations-of-set {x∈A. ¬P x})) (is ?lhs = ?rhs)

<proof>

end

25 Discrete subprobability distribution

theory *SPMF imports*

Probability-Mass-Function

HOL-Library.Complete-Partial-Order2

HOL-Library.Rewrite

begin

25.1 Auxiliary material

lemma *cSUP-singleton [simp]: (SUP x∈{x}. f x :: - :: conditionally-complete-lattice)*

= f x

<proof>

25.1.1 More about extended reals

lemma *[simp]:*

shows *ennreal-max-0: ennreal (max 0 x) = ennreal x*

and *ennreal-max-0'*: $\text{ennreal } (\max x \ 0) = \text{ennreal } x$
 $\langle \text{proof} \rangle$

lemma *e2ennreal-0 [simp]*: $e2\text{ennreal } 0 = 0$
 $\langle \text{proof} \rangle$

lemma *enn2real-bot [simp]*: $\text{enn2real } \perp = 0$
 $\langle \text{proof} \rangle$

lemma *continuous-at-ennreal[continuous-intros]*: $\text{continuous } F \ f \implies \text{continuous } F$
 $(\lambda x. \text{ennreal } (f \ x))$
 $\langle \text{proof} \rangle$

lemma *ennreal-Sup*:
assumes *: $(\text{SUP } a \in A. \text{ennreal } a) \neq \top$
and $A \neq \{\}$
shows $\text{ennreal } (\text{Sup } A) = (\text{SUP } a \in A. \text{ennreal } a)$
 $\langle \text{proof} \rangle$

lemma *ennreal-SUP*:
 $\llbracket (\text{SUP } a \in A. \text{ennreal } (f \ a)) \neq \top; A \neq \{\} \rrbracket \implies \text{ennreal } (\text{SUP } a \in A. f \ a) = (\text{SUP } a \in A. \text{ennreal } (f \ a))$
 $\langle \text{proof} \rangle$

lemma *ennreal-lt-0*: $x < 0 \implies \text{ennreal } x = 0$
 $\langle \text{proof} \rangle$

25.1.2 More about 'a option

lemma *None-in-map-option-image [simp]*: $\text{None} \in \text{map-option } f \ 'A \longleftrightarrow \text{None} \in A$
 $\langle \text{proof} \rangle$

lemma *Some-in-map-option-image [simp]*: $\text{Some } x \in \text{map-option } f \ 'A \longleftrightarrow (\exists y. x = f \ y \wedge \text{Some } y \in A)$
 $\langle \text{proof} \rangle$

lemma *case-option-collapse*: $\text{case-option } x \ (\lambda -. x) = (\lambda -. x)$
 $\langle \text{proof} \rangle$

lemma *case-option-id*: $\text{case-option } \text{None } \text{Some} = \text{id}$
 $\langle \text{proof} \rangle$

inductive *ord-option* :: $('a \Rightarrow 'b \Rightarrow \text{bool}) \Rightarrow 'a \text{ option} \Rightarrow 'b \text{ option} \Rightarrow \text{bool}$
for *ord* :: $'a \Rightarrow 'b \Rightarrow \text{bool}$

where

None: $\text{ord-option } \text{ord } \text{None } x$
 $|$ *Some*: $\text{ord } x \ y \implies \text{ord-option } \text{ord } (\text{Some } x) (\text{Some } y)$

inductive-simps *ord-option-simps* [simp]:

ord-option ord None x
ord-option ord x None
ord-option ord (Some x) (Some y)
ord-option ord (Some x) None

inductive-simps *ord-option-eq-simps* [simp]:

ord-option (=) None y
ord-option (=) (Some x) y

lemma *ord-option-refl*: $(\bigwedge y. y \in \text{set-option } x \implies \text{ord } y \ y) \implies \text{ord-option ord } x$
 <proof>

lemma *reflp-ord-option*: $\text{reflp ord} \implies \text{reflp } (\text{ord-option ord})$
 <proof>

lemma *ord-option-trans*:

$\llbracket \text{ord-option ord } x \ y; \text{ord-option ord } y \ z; \bigwedge a \ b \ c. \llbracket a \in \text{set-option } x; b \in \text{set-option } y; c \in \text{set-option } z; \text{ord } a \ b; \text{ord } b \ c \rrbracket \implies \text{ord } a \ c \rrbracket$
 $\implies \text{ord-option ord } x \ z$
 <proof>

lemma *transp-ord-option*: $\text{transp ord} \implies \text{transp } (\text{ord-option ord})$
 <proof>

lemma *antisymp-ord-option*: $\text{antisymp ord} \implies \text{antisymp } (\text{ord-option ord})$
 <proof>

lemma *ord-option-chainD*:

Complete-Partial-Order.chain (ord-option ord) Y
 $\implies \text{Complete-Partial-Order.chain ord } \{x. \text{Some } x \in Y\}$
 <proof>

definition *lub-option* :: $('a \text{ set} \Rightarrow 'b) \Rightarrow 'a \text{ option set} \Rightarrow 'b \text{ option}$

where *lub-option lub* $Y = (\text{if } Y \subseteq \{\text{None}\} \text{ then None else Some } (\text{lub } \{x. \text{Some } x \in Y\}))$

lemma *map-lub-option*: $\text{map-option } f \ (\text{lub-option lub } Y) = \text{lub-option } (f \circ \text{lub}) \ Y$
 <proof>

lemma *lub-option-upper*:

assumes *Complete-Partial-Order.chain (ord-option ord) Y* $x \in Y$
and *lub-upper*: $\bigwedge Y \ x. \llbracket \text{Complete-Partial-Order.chain ord } Y; x \in Y \rrbracket \implies \text{ord } x \ (\text{lub } Y)$
shows *ord-option ord x (lub-option lub Y)*
 <proof>

lemma *lub-option-least*:

assumes *Y*: *Complete-Partial-Order.chain* (*ord-option ord*) *Y*
and *upper*: $\bigwedge x. x \in Y \implies \text{ord-option } \text{ord } x \ y$
assumes *lub-least*: $\bigwedge Y y. \llbracket \text{Complete-Partial-Order.chain } \text{ord } Y; \bigwedge x. x \in Y \implies \text{ord } x \ y \rrbracket \implies \text{ord } (\text{lub } Y) \ y$
shows *ord-option ord* (*lub-option lub Y*) *y*
 $\langle \text{proof} \rangle$

lemma *lub-map-option*: *lub-option lub* (*map-option f* ‘ *Y*) = *lub-option* (*lub* \circ (*f*) *Y*)
 $\langle \text{proof} \rangle$

lemma *ord-option-mono*: $\llbracket \text{ord-option } A \ x \ y; \bigwedge x \ y. A \ x \ y \implies B \ x \ y \rrbracket \implies \text{ord-option } B \ x \ y$
 $\langle \text{proof} \rangle$

lemma *ord-option-mono'* [*mono*]:

$(\bigwedge x \ y. A \ x \ y \longrightarrow B \ x \ y) \implies \text{ord-option } A \ x \ y \longrightarrow \text{ord-option } B \ x \ y$
 $\langle \text{proof} \rangle$

lemma *ord-option-compp*: *ord-option* (*A OO B*) = *ord-option A OO ord-option B*
 $\langle \text{proof} \rangle$

lemma *ord-option-inf*: *inf* (*ord-option A*) (*ord-option B*) = *ord-option* (*inf A B*)
(is ?lhs = ?rhs)
 $\langle \text{proof} \rangle$

lemma *ord-option-map2*: *ord-option ord x* (*map-option f y*) = *ord-option* ($\lambda x \ y. \text{ord } x \ (f \ y)$) *x y*
 $\langle \text{proof} \rangle$

lemma *ord-option-map1*: *ord-option ord* (*map-option f x*) *y* = *ord-option* ($\lambda x \ y. \text{ord } (f \ x) \ y$) *x y*
 $\langle \text{proof} \rangle$

lemma *option-ord-Some1-iff*: *option-ord* (*Some x*) *y* $\longleftrightarrow y = \text{Some } x$
 $\langle \text{proof} \rangle$

25.1.3 A relator for sets that treats sets like predicates

context includes *lifting-syntax*
begin

definition *rel-pred* :: (*'a* \Rightarrow *'b* \Rightarrow *bool*) \Rightarrow *'a set* \Rightarrow *'b set* \Rightarrow *bool*
where *rel-pred R A B* = (*R ==> (=)*) ($\lambda x. x \in A$) ($\lambda y. y \in B$)

lemma *rel-predI*: (*R ==> (=)*) ($\lambda x. x \in A$) ($\lambda y. y \in B$) $\implies \text{rel-pred } R \ A \ B$
 $\langle \text{proof} \rangle$

lemma *rel-predD*: $\llbracket \text{rel-pred } R \ A \ B; \ R \ x \ y \rrbracket \implies x \in A \longleftrightarrow y \in B$
 $\langle \text{proof} \rangle$

lemma *Collect-parametric*: $((A \implies (=)) \implies \text{rel-pred } A) \text{ Collect Collect}$
 — Declare this rule as *transfer-rule* only locally because it blows up the search space for *transfer* (in combination with *Collect-transfer*)
 $\langle \text{proof} \rangle$

end

25.1.4 Monotonicity rules

lemma *monotone-gfp-eadd1*: $\text{monotone } (\geq) (\geq) (\lambda x. x + y :: \text{enat})$
 $\langle \text{proof} \rangle$

lemma *monotone-gfp-eadd2*: $\text{monotone } (\geq) (\geq) (\lambda y. x + y :: \text{enat})$
 $\langle \text{proof} \rangle$

lemma *mono2mono-gfp-eadd*[*THEN* *gfp.mono2mono2*, *cont-intro*, *simp*]:
shows *monotone-eadd*: $\text{monotone } (\text{rel-prod } (\geq) (\geq)) (\geq) (\lambda(x, y). x + y :: \text{enat})$
 $\langle \text{proof} \rangle$

lemma *eadd-gfp-partial-function-mono* [*partial-function-mono*]:
 $\llbracket \text{monotone } (\text{fun-ord } (\geq)) (\geq) f; \text{monotone } (\text{fun-ord } (\geq)) (\geq) g \rrbracket$
 $\implies \text{monotone } (\text{fun-ord } (\geq)) (\geq) (\lambda x. f \ x + g \ x :: \text{enat})$
 $\langle \text{proof} \rangle$

lemma *mono2mono-ereal*[*THEN* *lfp.mono2mono*]:
shows *monotone-ereal*: $\text{monotone } (\leq) (\leq) \text{ereal}$
 $\langle \text{proof} \rangle$

lemma *mono2mono-ennreal*[*THEN* *lfp.mono2mono*]:
shows *monotone-ennreal*: $\text{monotone } (\leq) (\leq) \text{ennreal}$
 $\langle \text{proof} \rangle$

25.1.5 Bijections

lemma *bi-unique-rel-set-bij-betw*:
assumes *unique*: *bi-unique* *R*
and *rel*: *rel-set* *R* *A* *B*
shows $\exists f. \text{bij-betw } f \ A \ B \wedge (\forall x \in A. \ R \ x \ (f \ x))$
 $\langle \text{proof} \rangle$

lemma *bij-betw-rel-setD*: $\text{bij-betw } f \ A \ B \implies \text{rel-set } (\lambda x \ y. y = f \ x) \ A \ B$
 $\langle \text{proof} \rangle$

25.2 Subprobability mass function

type-synonym *'a* *spmf* = *'a* *option* *pmf*
translations $(\text{type}) \ 'a \ \text{spmf} \leftarrow (\text{type}) \ 'a \ \text{option} \ \text{pmf}$

definition $\text{measure-spmf} :: 'a \text{ spmf} \Rightarrow 'a \text{ measure}$

where $\text{measure-spmf } p = \text{distr } (\text{restrict-space } (\text{measure-pmf } p) (\text{range } \text{Some}))$
 $(\text{count-space } \text{UNIV}) \text{ the}$

abbreviation $\text{spmf} :: 'a \text{ spmf} \Rightarrow 'a \Rightarrow \text{real}$

where $\text{spmf } p \ x \equiv \text{pmf } p \ (\text{Some } x)$

lemma $\text{space-measure-spmf}: \text{space } (\text{measure-spmf } p) = \text{UNIV}$

$\langle \text{proof} \rangle$

lemma $\text{sets-measure-spmf} [\text{simp}, \text{measurable-cong}]: \text{sets } (\text{measure-spmf } p) = \text{sets}$
 $(\text{count-space } \text{UNIV})$

$\langle \text{proof} \rangle$

lemma $\text{measure-spmf-not-bot} [\text{simp}]: \text{measure-spmf } p \neq \perp$

$\langle \text{proof} \rangle$

lemma $\text{measurable-the-measure-pmf-Some} [\text{measurable}, \text{simp}]:$

$\text{the } \in \text{measurable } (\text{restrict-space } (\text{measure-pmf } p) (\text{range } \text{Some})) (\text{count-space } \text{UNIV})$

$\langle \text{proof} \rangle$

lemma $\text{measurable-spmf-measure1} [\text{simp}]: \text{measurable } (\text{measure-spmf } M) \ N = \text{UNIV}$
 $\rightarrow \text{space } N$

$\langle \text{proof} \rangle$

lemma $\text{measurable-spmf-measure2} [\text{simp}]: \text{measurable } N \ (\text{measure-spmf } M) = \text{measurable } N$
 $(\text{count-space } \text{UNIV})$

$\langle \text{proof} \rangle$

lemma $\text{subprob-space-measure-spmf} [\text{simp}, \text{intro!}]: \text{subprob-space } (\text{measure-spmf } p)$

$\langle \text{proof} \rangle$

interpretation $\text{measure-spmf}: \text{subprob-space measure-spmf } p \text{ for } p$

$\langle \text{proof} \rangle$

lemma $\text{finite-measure-spmf} [\text{simp}]: \text{finite-measure } (\text{measure-spmf } p)$

$\langle \text{proof} \rangle$

lemma $\text{spmf-conv-measure-spmf}: \text{spmf } p \ x = \text{measure } (\text{measure-spmf } p) \ \{x\}$

$\langle \text{proof} \rangle$

lemma $\text{emeasure-measure-spmf-conv-measure-pmf}: \text{emeasure } (\text{measure-spmf } p) \ A = \text{emeasure } (\text{measure-pmf } p) \ (\text{Some } 'A)$

$\langle \text{proof} \rangle$

lemma $\text{measure-measure-spmf-conv-measure-pmf}: \text{measure } (\text{measure-spmf } p) \ A = \text{measure } (\text{measure-pmf } p) \ (\text{Some } 'A)$

$\langle \text{proof} \rangle$

$\langle \text{proof} \rangle$

lemma *emeasure-spmf-map-pmf-Some* [simp]:

$\text{emeasure} (\text{measure-spmf} (\text{map-pmf } \text{Some } p)) A = \text{emeasure} (\text{measure-pmf } p) A$
 $\langle \text{proof} \rangle$

lemma *measure-spmf-map-pmf-Some* [simp]:

$\text{measure} (\text{measure-spmf} (\text{map-pmf } \text{Some } p)) A = \text{measure} (\text{measure-pmf } p) A$
 $\langle \text{proof} \rangle$

lemma *nn-integral-measure-spmf*: $(\int^+ x. f x \partial \text{measure-spmf } p) = \int^+ x. \text{ennreal} (\text{spmfp } p x) * f x \partial \text{count-space } \text{UNIV}$

(is ?lhs = ?rhs)

$\langle \text{proof} \rangle$

lemma *integral-measure-spmf*:

assumes *integrable* ($\text{measure-spmf } p$) f

shows $(\int x. f x \partial \text{measure-spmf } p) = \int x. \text{spmfp } p x * f x \partial \text{count-space } \text{UNIV}$

$\langle \text{proof} \rangle$

lemma *emeasure-spmf-single*: $\text{emeasure} (\text{measure-spmf } p) \{x\} = \text{spmfp } p x$

$\langle \text{proof} \rangle$

lemma *measurable-measure-spmf*[*measurable*]:

$(\lambda x. \text{measure-spmf } (M x)) \in \text{measurable} (\text{count-space } \text{UNIV}) (\text{subprob-algebra} (\text{count-space } \text{UNIV}))$

$\langle \text{proof} \rangle$

lemma *nn-integral-measure-spmf-conv-measure-pmf*:

assumes [*measurable*]: $f \in \text{borel-measurable} (\text{count-space } \text{UNIV})$

shows $\text{nn-integral} (\text{measure-spmf } p) f = \text{nn-integral} (\text{restrict-space} (\text{measure-pmf } p) (\text{range } \text{Some})) (f \circ \text{the})$

$\langle \text{proof} \rangle$

lemma *measure-spmf-in-space-subprob-algebra* [simp]:

$\text{measure-spmf } p \in \text{space} (\text{subprob-algebra} (\text{count-space } \text{UNIV}))$

$\langle \text{proof} \rangle$

lemma *nn-integral-spmf-neq-top*: $(\int^+ x. \text{spmfp } p x \partial \text{count-space } \text{UNIV}) \neq \top$

$\langle \text{proof} \rangle$

lemma *SUP-spmf-neq-top'*: $(\text{SUP } p \in Y. \text{ennreal} (\text{spmfp } p x)) \neq \top$

$\langle \text{proof} \rangle$

lemma *SUP-spmf-neq-top*: $(\text{SUP } i. \text{ennreal} (\text{spmfp } (Y i) x)) \neq \top$

$\langle \text{proof} \rangle$

lemma *SUP-emeasure-spmf-neq-top*: $(\text{SUP } p \in Y. \text{emeasure} (\text{measure-spmf } p) A) \neq \top$

$\neq \top$

$\langle \text{proof} \rangle$

25.3 Support

definition $\text{set-spmf} :: 'a \text{ spmf} \Rightarrow 'a \text{ set}$
where $\text{set-spmf } p = \text{set-pmf } p \gg= \text{set-option}$

lemma set-spmf-rep-eq : $\text{set-spmf } p = \{x. \text{measure } (\text{measure-spmf } p) \{x\} \neq 0\}$
 $\langle \text{proof} \rangle$

lemma in-set-spmf : $x \in \text{set-spmf } p \longleftrightarrow \text{Some } x \in \text{set-pmf } p$
 $\langle \text{proof} \rangle$

lemma $\text{AE-measure-spmf-iff}$ [simp]: $(\text{AE } x \text{ in } \text{measure-spmf } p. P \ x) \longleftrightarrow (\forall x \in \text{set-spmf } p. P \ x)$
 $\langle \text{proof} \rangle$

lemma $\text{spmfs-eq-0-set-spmf}$: $\text{spmfs } p \ x = 0 \longleftrightarrow x \notin \text{set-spmf } p$
 $\langle \text{proof} \rangle$

lemma $\text{in-set-spmf-iff-spmf}$: $x \in \text{set-spmf } p \longleftrightarrow \text{spmfs } p \ x \neq 0$
 $\langle \text{proof} \rangle$

lemma $\text{set-spmf-return-pmf-None}$ [simp]: $\text{set-spmf } (\text{return-pmf } \text{None}) = \{\}$
 $\langle \text{proof} \rangle$

lemma $\text{countable-set-spmf}$ [simp]: $\text{countable } (\text{set-spmf } p)$
 $\langle \text{proof} \rangle$

lemma spmfs-eqI :
assumes $\bigwedge i. \text{spmfs } p \ i = \text{spmfs } q \ i$
shows $p = q$
 $\langle \text{proof} \rangle$

lemma $\text{integral-measure-spmf-restrict}$:
fixes $f :: 'a \Rightarrow 'b :: \{\text{banach}, \text{second-countable-topology}\}$
shows $(\int x. f \ x \ \partial \text{measure-spmf } M) = (\int x. f \ x \ \partial \text{restrict-space } (\text{measure-spmf } M) \ (\text{set-spmf } M))$
 $\langle \text{proof} \rangle$

lemma $\text{nn-integral-measure-spmf'}$:
 $(\int^+ x. f \ x \ \partial \text{measure-spmf } p) = \int^+ x. \text{ennreal } (\text{spmfs } p \ x) * f \ x \ \partial \text{count-space } (\text{set-spmf } p)$
 $\langle \text{proof} \rangle$

25.4 Functorial structure

abbreviation $\text{map-spmf} :: ('a \Rightarrow 'b) \Rightarrow 'a \text{ spmf} \Rightarrow 'b \text{ spmf}$
where $\text{map-spmf } f \equiv \text{map-pmf } (\text{map-option } f)$

context begin

$\langle ML \rangle$

lemma *map-comp*: $\text{map-spmf } f \ (\text{map-spmf } g \ p) = \text{map-spmf } (f \circ g) \ p$
 $\langle \text{proof} \rangle$

lemma *map-id0*: $\text{map-spmf } \text{id} = \text{id}$
 $\langle \text{proof} \rangle$

lemma *map-id [simp]*: $\text{map-spmf } \text{id} \ p = p$
 $\langle \text{proof} \rangle$

lemma *map-ident [simp]*: $\text{map-spmf } (\lambda x. \ x) \ p = p$
 $\langle \text{proof} \rangle$

end

lemma *set-map-spmf [simp]*: $\text{set-spmf } (\text{map-spmf } f \ p) = f \ ` \ \text{set-spmf } p$
 $\langle \text{proof} \rangle$

lemma *map-spmf-cong*:
 $\llbracket p = q; \bigwedge x. \ x \in \text{set-spmf } q \implies f \ x = g \ x \rrbracket \implies \text{map-spmf } f \ p = \text{map-spmf } g \ q$
 $\langle \text{proof} \rangle$

lemma *map-spmf-cong-simp*:
 $\llbracket p = q; \bigwedge x. \ x \in \text{set-spmf } q = \text{simp} \implies f \ x = g \ x \rrbracket$
 $\implies \text{map-spmf } f \ p = \text{map-spmf } g \ q$
 $\langle \text{proof} \rangle$

lemma *map-spmf-idI*: $(\bigwedge x. \ x \in \text{set-spmf } p \implies f \ x = x) \implies \text{map-spmf } f \ p = p$
 $\langle \text{proof} \rangle$

lemma *emeasure-map-spmf*:
 $\text{emeasure } (\text{measure-spmf } (\text{map-spmf } f \ p)) \ A = \text{emeasure } (\text{measure-spmf } p) \ (f \ - \ ` \ A)$
 $\langle \text{proof} \rangle$

lemma *measure-map-spmf*: $\text{measure } (\text{measure-spmf } (\text{map-spmf } f \ p)) \ A = \text{measure } (\text{measure-spmf } p) \ (f \ - \ ` \ A)$
 $\langle \text{proof} \rangle$

lemma *measure-map-spmf-conv-distr*:
 $\text{measure-spmf } (\text{map-spmf } f \ p) = \text{distr } (\text{measure-spmf } p) \ (\text{count-space } \text{UNIV}) \ f$
 $\langle \text{proof} \rangle$

lemma *spmf-map-pmf-Some [simp]*: $\text{spmf } (\text{map-pmf } \text{Some } p) \ i = \text{pmf } p \ i$
 $\langle \text{proof} \rangle$

lemma *spmf-map-inj*: $\llbracket \text{inj-on } f \ (\text{set-spmf } M); \ x \in \text{set-spmf } M \rrbracket \implies \text{spmf } (\text{map-spmf } f \ p) \ x = \text{spmf } p \ x$

$f M) (f x) = \text{spmf } M x$
 $\langle \text{proof} \rangle$

lemma *spmf-map-inj'*: $\text{inj } f \implies \text{spmf } (\text{map-spmf } f M) (f x) = \text{spmf } M x$
 $\langle \text{proof} \rangle$

lemma *spmf-map-outside*: $x \notin f \text{ ` } \text{set-spmf } M \implies \text{spmf } (\text{map-spmf } f M) x = 0$
 $\langle \text{proof} \rangle$

lemma *ennreal-spmf-map*: $\text{ennreal } (\text{spmf } (\text{map-spmf } f p) x) = \text{emeasure } (\text{measure-spmf } p) (f \text{ ` } \{x\})$
 $\langle \text{proof} \rangle$

lemma *spmf-map*: $\text{spmf } (\text{map-spmf } f p) x = \text{measure } (\text{measure-spmf } p) (f \text{ ` } \{x\})$
 $\langle \text{proof} \rangle$

lemma *ennreal-spmf-map-conv-nn-integral*:
 $\text{ennreal } (\text{spmf } (\text{map-spmf } f p) x) = \text{integral}^N (\text{measure-spmf } p) (\text{indicator } (f \text{ ` } \{x\}))$
 $\langle \text{proof} \rangle$

25.5 Monad operations

25.5.1 Return

abbreviation *return-spmf* :: $'a \Rightarrow 'a \text{ spmf}$
where $\text{return-spmf } x \equiv \text{return-pmf } (\text{Some } x)$

lemma *pmf-return-spmf*: $\text{pmf } (\text{return-spmf } x) y = \text{indicator } \{y\} (\text{Some } x)$
 $\langle \text{proof} \rangle$

lemma *measure-spmf-return-spmf*: $\text{measure-spmf } (\text{return-spmf } x) = \text{Giry-Monad.return } (\text{count-space UNIV}) x$
 $\langle \text{proof} \rangle$

lemma *measure-spmf-return-pmf-None* [simp]: $\text{measure-spmf } (\text{return-pmf None}) = \text{null-measure } (\text{count-space UNIV})$
 $\langle \text{proof} \rangle$

lemma *set-return-spmf* [simp]: $\text{set-spmf } (\text{return-spmf } x) = \{x\}$
 $\langle \text{proof} \rangle$

25.5.2 Bind

definition *bind-spmf* :: $'a \text{ spmf} \Rightarrow ('a \Rightarrow 'b \text{ spmf}) \Rightarrow 'b \text{ spmf}$
where $\text{bind-spmf } x f = \text{bind-pmf } x (\lambda a. \text{case } a \text{ of None} \Rightarrow \text{return-pmf None} \mid \text{Some } a' \Rightarrow f a')$

adhoc-overloading *Monad-Syntax.bind* $\equiv \text{bind-spmf}$

lemma *return-None-bind-spmf* [simp]: $\text{return-pmf None} \gg= (f :: 'a \Rightarrow -) = \text{return-pmf None}$
 <proof>

lemma *return-bind-spmf* [simp]: $\text{return-spmf } x \gg= f = f \ x$
 <proof>

lemma *bind-return-spmf* [simp]: $x \gg= \text{return-spmf} = x$
 <proof>

lemma *bind-spmf-assoc* [simp]:
 fixes $x :: 'a \text{ spmf}$ and $f :: 'a \Rightarrow 'b \text{ spmf}$ and $g :: 'b \Rightarrow 'c \text{ spmf}$
 shows $(x \gg= f) \gg= g = x \gg= (\lambda y. f \ y \gg= g)$
 <proof>

lemma *pmf-bind-spmf-None*: $\text{pmf } (p \gg= f) \text{ None} = \text{pmf } p \text{ None} + \int x. \text{pmf } (f \ x) \text{ None} \ \partial \text{measure-spmf } p$
 (is ?lhs = ?rhs)
 <proof>

lemma *spmfm-bind*: $\text{spmfm } (p \gg= f) \ y = \int x. \text{spmfm } (f \ x) \ y \ \partial \text{measure-spmfm } p$
 <proof>

lemma *ennreal-spmfm-bind*: $\text{ennreal } (\text{spmfm } (p \gg= f) \ x) = \int^+ y. \text{spmfm } (f \ y) \ x \ \partial \text{measure-spmfm } p$
 <proof>

lemma *measure-spmfm-bind-pmf*: $\text{measure-spmfm } (p \gg= f) = \text{measure-pmf } p \gg= \text{measure-spmfm} \circ f$
 (is ?lhs = ?rhs)
 <proof>

lemma *measure-spmfm-bind*: $\text{measure-spmfm } (p \gg= f) = \text{measure-spmfm } p \gg= \text{measure-spmfm} \circ f$
 (is ?lhs = ?rhs)
 <proof>

lemma *map-spmfm-bind-spmfm*: $\text{map-spmfm } f (\text{bind-spmfm } p \ g) = \text{bind-spmfm } p (\text{map-spmfm } f \circ g)$
 <proof>

lemma *bind-map-spmfm*: $\text{map-spmfm } f \ p \gg= g = p \gg= g \circ f$
 <proof>

lemma *spmfm-bind-leI*:
 assumes $\bigwedge y. y \in \text{set-spmfm } p \implies \text{spmfm } (f \ y) \ x \leq r$
 and $0 \leq r$
 shows $\text{spmfm } (\text{bind-spmfm } p \ f) \ x \leq r$
 <proof>

lemma *map-spmf-conv-bind-spmf*: $\text{map-spmf } f \, p = (p \gg= (\lambda x. \text{return-spmf } (f \, x)))$
 $\langle \text{proof} \rangle$

lemma *bind-spmf-cong*:
 $\llbracket p = q; \bigwedge x. x \in \text{set-spmf } q \implies f \, x = g \, x \rrbracket \implies \text{bind-spmf } p \, f = \text{bind-spmf } q \, g$
 $\langle \text{proof} \rangle$

lemma *bind-spmf-cong-simp*:
 $\llbracket p = q; \bigwedge x. x \in \text{set-spmf } q =_{\text{simp}} \implies f \, x = g \, x \rrbracket$
 $\implies \text{bind-spmf } p \, f = \text{bind-spmf } q \, g$
 $\langle \text{proof} \rangle$

lemma *set-bind-spmf*: $\text{set-spmf } (M \gg= f) = \text{set-spmf } M \gg= (\text{set-spmf } \circ f)$
 $\langle \text{proof} \rangle$

lemma *bind-spmf-const-return-None* [simp]: $\text{bind-spmf } p \, (\lambda \cdot. \text{return-pmf } \text{None}) = \text{return-pmf } \text{None}$
 $\langle \text{proof} \rangle$

lemma *bind-commute-spmf*:
 $\text{bind-spmf } p \, (\lambda x. \text{bind-spmf } q \, (f \, x)) = \text{bind-spmf } q \, (\lambda y. \text{bind-spmf } p \, (\lambda x. f \, x \, y))$
 $(\text{is } ?lhs = ?rhs)$
 $\langle \text{proof} \rangle$

25.6 Relator

abbreviation *rel-spmf* :: $('a \Rightarrow 'b \Rightarrow \text{bool}) \Rightarrow 'a \, \text{spmf} \Rightarrow 'b \, \text{spmf} \Rightarrow \text{bool}$
where $\text{rel-spmf } R \equiv \text{rel-pmf } (\text{rel-option } R)$

lemma *rel-spmf-mono*:
 $\llbracket \text{rel-spmf } A \, f \, g; \bigwedge x \, y. A \, x \, y \implies B \, x \, y \rrbracket \implies \text{rel-spmf } B \, f \, g$
 $\langle \text{proof} \rangle$

lemma *rel-spmf-mono-strong*:
 $\llbracket \text{rel-spmf } A \, f \, g; \bigwedge x \, y. \llbracket A \, x \, y; x \in \text{set-spmf } f; y \in \text{set-spmf } g \rrbracket \implies B \, x \, y \rrbracket \implies \text{rel-spmf } B \, f \, g$
 $\langle \text{proof} \rangle$

lemma *rel-spmf-reflI*: $(\bigwedge x. x \in \text{set-spmf } p \implies P \, x \, x) \implies \text{rel-spmf } P \, p \, p$
 $\langle \text{proof} \rangle$

lemma *rel-spmfI* [intro?]:
 $\llbracket \bigwedge x \, y. (x, y) \in \text{set-spmf } pq \implies P \, x \, y; \text{map-spmf } \text{fst } pq = p; \text{map-spmf } \text{snd } pq = q \rrbracket$
 $\implies \text{rel-spmf } P \, p \, q$
 $\langle \text{proof} \rangle$

lemma *rel-spmfE* [elim?, consumes 1, case-names rel-spmf]:

assumes $rel\text{-}spmf\ P\ p\ q$
obtains pq **where**
 $\bigwedge x\ y. (x, y) \in set\text{-}spmf\ pq \implies P\ x\ y$
 $p = map\text{-}spmf\ fst\ pq$
 $q = map\text{-}spmf\ snd\ pq$
 $\langle proof \rangle$

lemma $rel\text{-}spmf\text{-}simps$:
 $rel\text{-}spmf\ R\ p\ q \longleftrightarrow (\exists pq. (\forall (x, y) \in set\text{-}spmf\ pq. R\ x\ y) \wedge map\text{-}spmf\ fst\ pq = p \wedge map\text{-}spmf\ snd\ pq = q)$
 $\langle proof \rangle$

lemma $spmf\text{-}rel\text{-}map$:
shows $spmf\text{-}rel\text{-}map1$: $\bigwedge R\ f\ x. rel\text{-}spmf\ R\ (map\text{-}spmf\ f\ x) = rel\text{-}spmf\ (\lambda x. R\ (f\ x))\ x$
and $spmf\text{-}rel\text{-}map2$: $\bigwedge R\ x\ g\ y. rel\text{-}spmf\ R\ x\ (map\text{-}spmf\ g\ y) = rel\text{-}spmf\ (\lambda x\ y. R\ x\ (g\ y))\ x\ y$
 $\langle proof \rangle$

lemma $spmf\text{-}rel\text{-}conversep$: $rel\text{-}spmf\ R^{-1-1} = (rel\text{-}spmf\ R)^{-1-1}$
 $\langle proof \rangle$

lemma $spmf\text{-}rel\text{-}eq$: $rel\text{-}spmf\ (=) = (=)$
 $\langle proof \rangle$

context includes $lifting\text{-}syntax$
begin

lemma $bind\text{-}spmf\text{-}parametric$ [transfer-rule]:
 $(rel\text{-}spmf\ A \implies (A \implies rel\text{-}spmf\ B) \implies rel\text{-}spmf\ B)\ bind\text{-}spmf\ bind\text{-}spmf$
 $\langle proof \rangle$

lemma $return\text{-}spmf\text{-}parametric$: $(A \implies rel\text{-}spmf\ A)\ return\text{-}spmf\ return\text{-}spmf$
 $\langle proof \rangle$

lemma $map\text{-}spmf\text{-}parametric$: $((A \implies B) \implies rel\text{-}spmf\ A \implies rel\text{-}spmf\ B)\ map\text{-}spmf\ map\text{-}spmf$
 $\langle proof \rangle$

lemma $rel\text{-}spmf\text{-}parametric$:
 $((A \implies B \implies (=)) \implies rel\text{-}spmf\ A \implies rel\text{-}spmf\ B \implies (=))$
 $rel\text{-}spmf\ rel\text{-}spmf$
 $\langle proof \rangle$

lemma $set\text{-}spmf\text{-}parametric$ [transfer-rule]:
 $(rel\text{-}spmf\ A \implies rel\text{-}set\ A)\ set\text{-}spmf\ set\text{-}spmf$
 $\langle proof \rangle$

lemma $return\text{-}spmf\text{-}None\text{-}parametric$:

$(rel\text{-}spmf\ A)\ (return\text{-}pmf\ None)\ (return\text{-}pmf\ None)$
 $\langle proof \rangle$

end

lemma *rel-spmf-bindI*:

$\llbracket rel\text{-}spmf\ R\ p\ q; \bigwedge x\ y. R\ x\ y \implies rel\text{-}spmf\ P\ (f\ x)\ (g\ y) \rrbracket$
 $\implies rel\text{-}spmf\ P\ (p \ggg f)\ (q \ggg g)$
 $\langle proof \rangle$

lemma *rel-spmf-bind-reflI*:

$(\bigwedge x. x \in set\text{-}spmf\ p \implies rel\text{-}spmf\ P\ (f\ x)\ (g\ x)) \implies rel\text{-}spmf\ P\ (p \ggg f)\ (p \ggg g)$
 $\langle proof \rangle$

lemma *rel-pmf-return-pmfI*: $P\ x\ y \implies rel\text{-}pmf\ P\ (return\text{-}pmf\ x)\ (return\text{-}pmf\ y)$
 $\langle proof \rangle$

context includes *lifting-syntax*
begin

We do not yet have a relator for *'a measure*, so we combine *Sigma-Algebra.measure* and *measure-pmf*

lemma *measure-pmf-parametric*:

$(rel\text{-}pmf\ A \implies rel\text{-}pred\ A \implies (=))\ (\lambda p. measure\ (measure\text{-}pmf\ p))\ (\lambda q. measure\ (measure\text{-}pmf\ q))$
 $\langle proof \rangle$

lemma *measure-spmf-parametric*:

$(rel\text{-}spmf\ A \implies rel\text{-}pred\ A \implies (=))\ (\lambda p. measure\ (measure\text{-}spmf\ p))\ (\lambda q. measure\ (measure\text{-}spmf\ q))$
 $\langle proof \rangle$

end

25.7 From *'a pmf* to *'a spmf*

definition *spmf-of-pmf* :: *'a pmf* \Rightarrow *'a spmf*

where *spmf-of-pmf* = *map-pmf Some*

lemma *set-spmf-spmf-of-pmf [simp]*: *set-spmf* (*spmf-of-pmf* *p*) = *set-pmf* *p*
 $\langle proof \rangle$

lemma *spmf-spmf-of-pmf [simp]*: *spmf* (*spmf-of-pmf* *p*) *x* = *pmf* *p* *x*
 $\langle proof \rangle$

lemma *pmf-spmf-of-pmf-None [simp]*: *pmf* (*spmf-of-pmf* *p*) *None* = 0
 $\langle proof \rangle$

lemma *emeasure-spmf-of-pmf* [simp]: *emeasure* (*measure-spmf* (*spmf-of-pmf* *p*))
A = *emeasure* (*measure-pmf* *p*) *A*
 ⟨proof⟩

lemma *measure-spmf-spmf-of-pmf* [simp]: *measure-spmf* (*spmf-of-pmf* *p*) = *measure-pmf* *p*
 ⟨proof⟩

lemma *map-spmf-of-pmf* [simp]: *map-spmf* (*spmf-of-pmf* *p*) = *spmf-of-pmf* (*map-pmf* *f* *p*)
 ⟨proof⟩

lemma *rel-spmf-spmf-of-pmf* [simp]: *rel-spmf* *R* (*spmf-of-pmf* *p*) (*spmf-of-pmf* *q*)
 = *rel-pmf* *R* *p* *q*
 ⟨proof⟩

lemma *spmf-of-pmf-return-pmf* [simp]: *spmf-of-pmf* (*return-pmf* *x*) = *return-spmf* *x*
 ⟨proof⟩

lemma *bind-spmf-of-pmf* [simp]: *bind-spmf* (*spmf-of-pmf* *p*) *f* = *bind-pmf* *p* *f*
 ⟨proof⟩

lemma *set-spmf-bind-pmf*: *set-spmf* (*bind-pmf* *p* *f*) = *Set.bind* (*set-pmf* *p*) (*set-spmf* *f*)
 ⟨proof⟩

lemma *spmf-of-pmf-bind*: *spmf-of-pmf* (*bind-pmf* *p* *f*) = *bind-pmf* *p* ($\lambda x. \text{spmf-of-pmf } (f\ x)$)
 ⟨proof⟩

lemma *bind-pmf-return-spmf*: *p* $\gg=$ ($\lambda x. \text{return-spmf } (f\ x)$) = *spmf-of-pmf* (*map-pmf* *f* *p*)
 ⟨proof⟩

25.8 Weight of a subprobability

abbreviation *weight-spmf* :: 'a *spmf* \Rightarrow *real*
where *weight-spmf* *p* \equiv *measure* (*measure-spmf* *p*) (*space* (*measure-spmf* *p*))

lemma *weight-spmf-def*: *weight-spmf* *p* = *measure* (*measure-spmf* *p*) *UNIV*
 ⟨proof⟩

lemma *weight-spmf-le-1*: *weight-spmf* *p* ≤ 1
 ⟨proof⟩

lemma *weight-return-spmf* [simp]: *weight-spmf* (*return-spmf* *x*) = 1
 ⟨proof⟩

lemma *weight-return-pmf-None* [simp]: $\text{weight-spmf } (\text{return-pmf } \text{None}) = 0$
 ⟨proof⟩

lemma *weight-map-spmf* [simp]: $\text{weight-spmf } (\text{map-spmf } f \ p) = \text{weight-spmf } p$
 ⟨proof⟩

lemma *weight-spmf-of-pmf* [simp]: $\text{weight-spmf } (\text{spm-f-of-pmf } p) = 1$
 ⟨proof⟩

lemma *weight-spmf-nonneg*: $\text{weight-spmf } p \geq 0$
 ⟨proof⟩

lemma (in *finite-measure*) *integrable-weight-spmf* [simp]:
 $(\lambda x. \text{weight-spmf } (f \ x)) \in \text{borel-measurable } M \implies \text{integrable } M \ (\lambda x. \text{weight-spmf } (f \ x))$
 ⟨proof⟩

lemma *weight-spmf-eq-nn-integral-spmf*: $\text{weight-spmf } p = \int^+ x. \text{spm-f } p \ x \ \partial \text{count-space}$
 UNIV
 ⟨proof⟩

lemma *weight-spmf-eq-nn-integral-support*:
 $\text{weight-spmf } p = \int^+ x. \text{spm-f } p \ x \ \partial \text{count-space } (\text{set-spmf } p)$
 ⟨proof⟩

lemma *pmf-None-eq-weight-spmf*: $\text{pmf } p \ \text{None} = 1 - \text{weight-spmf } p$
 ⟨proof⟩

lemma *weight-spmf-conv-pmf-None*: $\text{weight-spmf } p = 1 - \text{pmf } p \ \text{None}$
 ⟨proof⟩

lemma *weight-spmf-lt-0*: $\neg \text{weight-spmf } p < 0$
 ⟨proof⟩

lemma *spm-f-le-weight*: $\text{spm-f } p \ x \leq \text{weight-spmf } p$
 ⟨proof⟩

lemma *weight-spmf-eq-0*: $\text{weight-spmf } p = 0 \iff p = \text{return-pmf } \text{None}$
 ⟨proof⟩

lemma *weight-bind-spmf*: $\text{weight-spmf } (x \gg f) = \text{lebesgue-integral } (\text{measure-spmf } x) \ (\text{weight-spmf } \circ f)$
 ⟨proof⟩

lemma *rel-spmf-weightD*: $\text{rel-spmf } A \ p \ q \implies \text{weight-spmf } p = \text{weight-spmf } q$
 ⟨proof⟩

lemma *rel-spmf-bij-betw*:
 assumes $f: \text{bij-betw } f \ (\text{set-spmf } p) \ (\text{set-spmf } q)$

and $eq: \bigwedge x. x \in \text{set-spmf } p \implies \text{spm} f p x = \text{spm} f q (f x)$
shows $\text{rel-spmf } (\lambda x y. f x = y) p q$
 $\langle \text{proof} \rangle$

25.9 From density to spmfs

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

definition $\text{embed-spmf} :: 'a \text{ spmf}$

where $\text{embed-spmf} = \text{embed-pmf } (\lambda x. \text{case } x \text{ of } \text{None} \Rightarrow 1 - \text{enn2real } (\int^+ x. \text{ennreal } (f x) \partial \text{count-space } \text{UNIV}) \mid \text{Some } x' \Rightarrow \text{max } 0 (f x'))$

context

assumes $\text{prob}: (\int^+ x. \text{ennreal } (f x) \partial \text{count-space } \text{UNIV}) \leq 1$
begin

lemma $\text{nn-integral-embed-spmf-eq-1}$:

$(\int^+ x. \text{ennreal } (\text{case } x \text{ of } \text{None} \Rightarrow 1 - \text{enn2real } (\int^+ x. \text{ennreal } (f x) \partial \text{count-space } \text{UNIV}) \mid \text{Some } x' \Rightarrow \text{max } 0 (f x')) \partial \text{count-space } \text{UNIV}) = 1$
 $(\text{is ?lhs} = - \text{ is } (\int^+ x. ?f x \partial ?M) = -)$
 $\langle \text{proof} \rangle$

lemma $\text{pmf-embed-spmf-None}$: $\text{pmf } \text{embed-spmf } \text{None} = 1 - \text{enn2real } (\int^+ x. \text{ennreal } (f x) \partial \text{count-space } \text{UNIV})$
 $\langle \text{proof} \rangle$

lemma $\text{spm} f \text{ embed-spmf } x = \text{max } 0 (f x)$
 $\langle \text{proof} \rangle$

end

end

lemma $\text{embed-spmf-K-0[simp]}$: $\text{embed-spmf } (\lambda -. 0) = \text{return-pmf } \text{None}$
 $\langle \text{proof} \rangle$

25.10 Ordering on spmfs

rel-pmf does not preserve a ccpo structure. Counterexample by Saheb-Djahromi: Take prefix order over bool llist and the set $\text{range } (\lambda n :: \text{nat}. \text{uniform } (\text{llist-n } n))$ where llist-n is the set of all llists of length n and uniform returns a uniform distribution over the given set. The set forms a chain in ord-pmf lprefix , but it has not an upper bound. Any upper bound may contain only infinite lists in its support because otherwise it is not greater than the $n+1$ -st element in the chain where n is the length of the finite list. Moreover its support must contain all infinite lists, because otherwise there is a finite list all of whose finite extensions are not in the support - a contradiction to the upper bound property. Hence, the support

is uncountable, but pmf’s only have countable support.

However, if all chains in the ccpo are finite, then it should preserve the ccpo structure.

abbreviation $\text{ord-spmf} :: ('a \Rightarrow 'a \Rightarrow \text{bool}) \Rightarrow 'a \text{ spmf} \Rightarrow 'a \text{ spmf} \Rightarrow \text{bool}$
where $\text{ord-spmf } \text{ord} \equiv \text{rel-pmf } (\text{ord-option } \text{ord})$

locale ord-spmf-syntax **begin**

notation ord-spmf (**infix** $\langle \sqsubseteq_1 \rangle$ 60)

end

lemma $\text{ord-spmf-map-spmf1}$: $\text{ord-spmf } R \ (\text{map-spmf } f \ p) = \text{ord-spmf } (\lambda x. R \ (f \ x))$
 p
 $\langle \text{proof} \rangle$

lemma $\text{ord-spmf-map-spmf2}$: $\text{ord-spmf } R \ p \ (\text{map-spmf } f \ q) = \text{ord-spmf } (\lambda x \ y. R \ x \ (f \ y)) \ p \ q$
 $\langle \text{proof} \rangle$

lemma $\text{ord-spmf-map-spmf12}$: $\text{ord-spmf } R \ (\text{map-spmf } f \ p) \ (\text{map-spmf } f \ q) = \text{ord-spmf } (\lambda x \ y. R \ (f \ x) \ (f \ y)) \ p \ q$
 $\langle \text{proof} \rangle$

lemmas $\text{ord-spmf-map-spmf} = \text{ord-spmf-map-spmf1 } \text{ord-spmf-map-spmf2 } \text{ord-spmf-map-spmf12}$

context **fixes** $\text{ord} :: 'a \Rightarrow 'a \Rightarrow \text{bool}$ (**structure**) **begin**

interpretation ord-spmf-syntax $\langle \text{proof} \rangle$

lemma ord-spmfI :

$\llbracket \bigwedge x \ y. (x, y) \in \text{set-spmf } pq \implies \text{ord } x \ y; \text{map-spmf } \text{fst } pq = p; \text{map-spmf } \text{snd } pq = q \rrbracket$
 $\implies p \sqsubseteq q$
 $\langle \text{proof} \rangle$

lemma ord-spmf-None $[simp]$: $\text{return-pmf } \text{None} \sqsubseteq x$
 $\langle \text{proof} \rangle$

lemma ord-spmf-reflI : $(\bigwedge x. x \in \text{set-spmf } p \implies \text{ord } x \ x) \implies p \sqsubseteq p$
 $\langle \text{proof} \rangle$

lemma rel-spmf-inf :

assumes $p \sqsubseteq q$

and $q \sqsubseteq p$

and refl : $\text{reflp } \text{ord}$

and trans : $\text{transp } \text{ord}$

shows $\text{rel-spmf } (\text{inf } \text{ord } \text{ord}^{-1-1}) \ p \ q$

$\langle \text{proof} \rangle$

end

lemma *ord-spmf-return-spmf2*: $\text{ord-spmf } R \ p \ (\text{return-spmf } y) \longleftrightarrow (\forall x \in \text{set-spmf } p. R \ x \ y)$
 $\langle \text{proof} \rangle$

lemma *ord-spmf-mono*: $\llbracket \text{ord-spmf } A \ p \ q; \bigwedge x \ y. A \ x \ y \implies B \ x \ y \rrbracket \implies \text{ord-spmf } B \ p \ q$
 $\langle \text{proof} \rangle$

lemma *ord-spmf-compp*: $\text{ord-spmf } (A \ OO \ B) = \text{ord-spmf } A \ OO \ \text{ord-spmf } B$
 $\langle \text{proof} \rangle$

lemma *ord-spmf-bindI*:
 assumes $pq: \text{ord-spmf } R \ p \ q$
 and $fg: \bigwedge x \ y. R \ x \ y \implies \text{ord-spmf } P \ (f \ x) \ (g \ y)$
 shows $\text{ord-spmf } P \ (p \ggg f) \ (q \ggg g)$
 $\langle \text{proof} \rangle$

lemma *ord-spmf-bind-reflI*:
 $(\bigwedge x. x \in \text{set-spmf } p \implies \text{ord-spmf } R \ (f \ x) \ (g \ x)) \implies \text{ord-spmf } R \ (p \ggg f) \ (p \ggg g)$
 $\langle \text{proof} \rangle$

lemma *ord-spmf-increaseI*:
 assumes $le: \bigwedge x. \text{spm}f \ p \ x \leq \text{spm}f \ q \ x$
 and $refl: \bigwedge x. x \in \text{set-spmf } p \implies R \ x \ x$
 shows $\text{ord-spmf } R \ p \ q$
 $\langle \text{proof} \rangle$

lemma *ord-spmf-eq-leD*:
 assumes $\text{ord-spmf } (=) \ p \ q$
 shows $\text{spm}f \ p \ x \leq \text{spm}f \ q \ x$
 $\langle \text{proof} \rangle$

lemma *ord-spmf-eqD-set-spmf*: $\text{ord-spmf } (=) \ p \ q \implies \text{set-spmf } p \subseteq \text{set-spmf } q$
 $\langle \text{proof} \rangle$

lemma *ord-spmf-eqD-emeasure*:
 $\text{ord-spmf } (=) \ p \ q \implies \text{emeasure } (\text{measure-spmf } p) \ A \leq \text{emeasure } (\text{measure-spmf } q) \ A$
 $\langle \text{proof} \rangle$

lemma *ord-spmf-eqD-measure-spmf*: $\text{ord-spmf } (=) \ p \ q \implies \text{measure-spmf } p \leq \text{measure-spmf } q$
 $\langle \text{proof} \rangle$

25.11 CCPO structure for the flat ccpo *ord-option* (=)

context fixes $Y :: 'a \ \text{spm}f \ \text{set} \ \text{begin}$

definition $\text{lub-spmf} :: 'a \text{ spmf}$

where $\text{lub-spmf} = \text{embed-spmf } (\lambda x. \text{enn2real } (\text{SUP } p \in Y. \text{ennreal } (\text{spm} f p x)))$

— We go through ennreal to have a sensible definition even if Y is empty.

lemma $\text{lub-spmf-empty [simp]: SPMF.lub-spmf } \{\} = \text{return-pmf None}$
 $\langle \text{proof} \rangle$

context assumes $\text{chain: Complete-Partial-Order.chain } (\text{ord-spmf } (=)) \ Y$
begin

lemma $\text{chain-ord-spmf-eqD: Complete-Partial-Order.chain } (\leq) ((\lambda p x. \text{ennreal } (\text{spm} f p x)) \text{ ' } Y)$
 $(\text{is Complete-Partial-Order.chain - } (?f \text{ ' } -))$
 $\langle \text{proof} \rangle$

lemma $\text{ord-spmf-eq-pmf-None-eq:}$
assumes $\text{le: ord-spmf } (=) \ p \ q$
and $\text{None: pmf } p \ \text{None} = \text{pmf } q \ \text{None}$
shows $p = q$
 $\langle \text{proof} \rangle$

lemma $\text{ord-spmf-eqD-pmf-None:}$
assumes $\text{ord-spmf } (=) \ x \ y$
shows $\text{pmf } x \ \text{None} \geq \text{pmf } y \ \text{None}$
 $\langle \text{proof} \rangle$

Chains on $'a \text{ spmf}$ maintain countable support. Thanks to Johannes Hölzl for the proof idea.

lemma $\text{spm} f\text{-chain-countable: countable } (\bigcup p \in Y. \text{set-spmf } p)$
 $\langle \text{proof} \rangle$

lemma $\text{lub-spmf-subprob: } (\int^+ x. (\text{SUP } p \in Y. \text{ennreal } (\text{spm} f p x)) \ \partial \text{count-space UNIV}) \leq 1$
 $\langle \text{proof} \rangle$

lemma $\text{spm} f\text{-lub-spmf:}$
assumes $Y \neq \{\}$
shows $\text{spm} f \ \text{lub-spmf } x = (\text{SUP } p \in Y. \text{spm} f p x)$
 $\langle \text{proof} \rangle$

lemma $\text{ennreal-spmf-lub-spmf: } Y \neq \{\} \implies \text{ennreal } (\text{spm} f \ \text{lub-spmf } x) = (\text{SUP } p \in Y. \text{ennreal } (\text{spm} f p x))$
 $\langle \text{proof} \rangle$

lemma lub-spmf-upper:
assumes $p: p \in Y$
shows $\text{ord-spmf } (=) \ p \ \text{lub-spmf}$
 $\langle \text{proof} \rangle$

lemma *lub-spmf-least*:

assumes $z: \bigwedge x. x \in Y \implies \text{ord-spmf } (=) x z$

shows $\text{ord-spmf } (=) \text{ lub-spmf } z$

$\langle \text{proof} \rangle$

lemma *set-lub-spmf*: $\text{set-spmf lub-spmf} = (\bigcup p \in Y. \text{set-spmf } p)$ (**is** $?lhs = ?rhs$)

$\langle \text{proof} \rangle$

lemma *emeasure-lub-spmf*:

assumes $Y: Y \neq \{\}$

shows $\text{emeasure } (\text{measure-spmf lub-spmf}) A = (\text{SUP } y \in Y. \text{emeasure } (\text{measure-spmf } y) A)$

(**is** $?lhs = ?rhs$)

$\langle \text{proof} \rangle$

lemma *measure-lub-spmf*:

assumes $Y: Y \neq \{\}$

shows $\text{measure } (\text{measure-spmf lub-spmf}) A = (\text{SUP } y \in Y. \text{measure } (\text{measure-spmf } y) A)$ (**is** $?lhs = ?rhs$)

$\langle \text{proof} \rangle$

lemma *weight-lub-spmf*:

assumes $Y: Y \neq \{\}$

shows $\text{weight-spmf lub-spmf} = (\text{SUP } y \in Y. \text{weight-spmf } y)$

$\langle \text{proof} \rangle$

lemma *measure-spmf-lub-spmf*:

assumes $Y: Y \neq \{\}$

shows $\text{measure-spmf lub-spmf} = (\text{SUP } p \in Y. \text{measure-spmf } p)$ (**is** $?lhs = ?rhs$)

$\langle \text{proof} \rangle$

end

end

lemma *partial-function-definitions-spmf*: $\text{partial-function-definitions } (\text{ord-spmf } (=)) \text{ lub-spmf}$

(**is** $\text{partial-function-definitions } ?R -$)

$\langle \text{proof} \rangle$

lemma *ccpo-spmf*: $\text{class.ccpo lub-spmf } (\text{ord-spmf } (=)) (\text{mk-less } (\text{ord-spmf } (=)))$

$\langle \text{proof} \rangle$

interpretation *spm*: $\text{partial-function-definitions ord-spmf } (=) \text{ lub-spmf}$

rewrites $\text{lub-spmf } \{\} \equiv \text{return-pmf None}$

$\langle \text{proof} \rangle$

$\langle ML \rangle$

declare *spmf.leq-refl*[*simp*]

declare *admissible-leI*[*OF cppo-spmf, cont-intro*]

abbreviation *mono-spmf* \equiv *monotone* (*fun-ord* (*ord-spmf* (=))) (*ord-spmf* (=))

lemma *lub-spmf-const* [*simp*]: *lub-spmf* {*p*} = *p*
 ⟨*proof*⟩

lemma *bind-spmf-mono'*:
 assumes *fg*: *ord-spmf* (=) *f g*
 and *hk*: $\bigwedge x :: 'a. \text{ord-spmf } (=) (h\ x) (k\ x)$
 shows *ord-spmf* (=) (*f* \gg *h*) (*g* \gg *k*)
 ⟨*proof*⟩

lemma *bind-spmf-mono* [*partial-function-mono*]:
 assumes *mf*: *mono-spmf* *B* and *mg*: $\bigwedge y. \text{mono-spmf } (\lambda f. C\ y\ f)$
 shows *mono-spmf* ($\lambda f. \text{bind-spmf } (B\ f) (\lambda y. C\ y\ f)$)
 ⟨*proof*⟩

lemma *monotone-bind-spmf1*: *monotone* (*ord-spmf* (=)) (*ord-spmf* (=)) ($\lambda y. \text{bind-spmf } y\ g$)
 ⟨*proof*⟩

lemma *monotone-bind-spmf2*:
 assumes *g*: $\bigwedge x. \text{monotone ord } (\text{ord-spmf } (=)) (\lambda y. g\ y\ x)$
 shows *monotone ord* (*ord-spmf* (=)) ($\lambda y. \text{bind-spmf } p\ (g\ y)$)
 ⟨*proof*⟩

lemma *bind-lub-spmf*:
 assumes *chain*: *Complete-Partial-Order.chain* (*ord-spmf* (=)) *Y*
 shows *bind-spmf* (*lub-spmf* *Y*) *f* = *lub-spmf* (($\lambda p. \text{bind-spmf } p\ f$) ‘ *Y*) (is ?*lhs* = ?*rhs*)
 ⟨*proof*⟩

lemma *map-lub-spmf*:
Complete-Partial-Order.chain (*ord-spmf* (=)) *Y*
 $\implies \text{map-spmf } f\ (\text{lub-spmf } Y) = \text{lub-spmf } (\text{map-spmf } f\ ‘\ Y)$
 ⟨*proof*⟩

lemma *mcont-bind-spmf1*: *mcont* *lub-spmf* (*ord-spmf* (=)) *lub-spmf* (*ord-spmf* (=)) ($\lambda y. \text{bind-spmf } y\ f$)
 ⟨*proof*⟩

lemma *bind-lub-spmf2*:
 assumes *chain*: *Complete-Partial-Order.chain* *ord* *Y*
 and *g*: $\bigwedge y. \text{monotone ord } (\text{ord-spmf } (=)) (g\ y)$
 shows *bind-spmf* *x* ($\lambda y. \text{lub-spmf } (g\ y\ ‘\ Y)$) = *lub-spmf* (($\lambda p. \text{bind-spmf } x\ (\lambda y. g\ y\ p)$) ‘ *Y*)

(is ?lhs = ?rhs)
 ⟨proof⟩

lemma *mcont-bind-spmf* [*cont-intro*]:
 assumes f : *mcont luba orda lub-spmf* (*ord-spmf* (=)) f
 and g : $\bigwedge y. \text{mcont luba orda lub-spmf } (\text{ord-spmf } (=)) (g\ y)$
 shows *mcont luba orda lub-spmf* (*ord-spmf* (=)) $(\lambda x. \text{bind-spmf } (f\ x) (\lambda y. g\ y\ x))$
 ⟨proof⟩

lemma *bind-pmf-mono* [*partial-function-mono*]:
 $(\bigwedge y. \text{mono-spmf } (\lambda f. C\ y\ f)) \implies \text{mono-spmf } (\lambda f. \text{bind-pmf } p\ (\lambda x. C\ x\ f))$
 ⟨proof⟩

lemma *map-spmf-mono* [*partial-function-mono*]: *mono-spmf* $B \implies \text{mono-spmf } (\lambda g. \text{map-spmf } f\ (B\ g))$
 ⟨proof⟩

lemma *mcont-map-spmf* [*cont-intro*]:
mcont luba orda lub-spmf (*ord-spmf* (=)) g
 $\implies \text{mcont luba orda lub-spmf } (\text{ord-spmf } (=)) (\lambda x. \text{map-spmf } f\ (g\ x))$
 ⟨proof⟩

lemma *monotone-set-spmf*: *monotone* (*ord-spmf* (=)) (\subseteq) *set-spmf*
 ⟨proof⟩

lemma *cont-set-spmf*: *cont lub-spmf* (*ord-spmf* (=)) *Union* (\subseteq) *set-spmf*
 ⟨proof⟩

lemma *mcont2mcont-set-spmf* [*THEN mcont2mcont, cont-intro*]:
 shows *mcont-set-spmf*: *mcont lub-spmf* (*ord-spmf* (=)) *Union* (\subseteq) *set-spmf*
 ⟨proof⟩

lemma *monotone-spmf*: *monotone* (*ord-spmf* (=)) (\leq) $(\lambda p. \text{spmf } p\ x)$
 ⟨proof⟩

lemma *cont-spmf*: *cont lub-spmf* (*ord-spmf* (=)) *Sup* (\leq) $(\lambda p. \text{spmf } p\ x)$
 ⟨proof⟩

lemma *mcont-spmf*: *mcont lub-spmf* (*ord-spmf* (=)) *Sup* (\leq) $(\lambda p. \text{spmf } p\ x)$
 ⟨proof⟩

lemma *cont-ennreal-spmf*: *cont lub-spmf* (*ord-spmf* (=)) *Sup* (\leq) $(\lambda p. \text{ennreal } (\text{spmf } p\ x))$
 ⟨proof⟩

lemma *mcont2mcont-ennreal-spmf* [*THEN mcont2mcont, cont-intro*]:
 shows *mcont-ennreal-spmf*: *mcont lub-spmf* (*ord-spmf* (=)) *Sup* (\leq) $(\lambda p. \text{ennreal } (\text{spmf } p\ x))$

$\langle \text{proof} \rangle$

lemma *nn-integral-map-spmf [simp]: nn-integral (measure-spmf (map-spmf f p))*
 $g = \text{nn-integral (measure-spmf p) (g \circ f)}$
 $\langle \text{proof} \rangle$

25.11.1 Admissibility of *rel-spmf*

lemma *rel-spmf-measureD:*

assumes *rel-spmf R p q*

shows *measure (measure-spmf p) A ≤ measure (measure-spmf q) {y. ∃ x ∈ A. R x y}* **(is ?lhs ≤ ?rhs)**
 $\langle \text{proof} \rangle$

locale *rel-spmf-characterisation =*

assumes *rel-pmf-measureI:*

$\bigwedge (R :: 'a \text{ option} \Rightarrow 'b \text{ option} \Rightarrow \text{bool}) \ p \ q.$

$(\bigwedge A. \text{measure (measure-pmf p) } A \leq \text{measure (measure-pmf q) } \{y. \exists x \in A. R \ x \ y\})$

$\implies \text{rel-pmf } R \ p \ q$

— This assumption is shown to hold in general in the AFP entry *MFMC-Countable*.

begin

context *fixes R :: 'a ⇒ 'b ⇒ bool* **begin**

lemma *rel-spmf-measureI:*

assumes *eq1: $\bigwedge A. \text{measure (measure-spmf p) } A \leq \text{measure (measure-spmf q) } \{y. \exists x \in A. R \ x \ y\}$*

assumes *eq2: weight-spmf q ≤ weight-spmf p*

shows *rel-spmf R p q*

$\langle \text{proof} \rangle$

lemma *admissible-rel-spmf:*

ccpo.admissible (prod-lub lub-spmf lub-spmf) (rel-prod (ord-spmf (=)) (ord-spmf (=))) (case-prod (rel-spmf R))

(is cppo.admissible ?lub ?ord ?P)

$\langle \text{proof} \rangle$

lemma *admissible-rel-spmf-mcont [cont-intro]:*

$\llbracket \text{mcont lub ord lub-spmf (ord-spmf (=)) } f; \text{mcont lub ord lub-spmf (ord-spmf (=)) } g \rrbracket$

$\implies \text{ccpo.admissible lub ord } (\lambda x. \text{rel-spmf } R \ (f \ x) \ (g \ x))$

$\langle \text{proof} \rangle$

context *includes lifting-syntax*

begin

lemma *fixp-spmf-parametric':*

assumes *f: $\bigwedge x. \text{monotone (ord-spmf (=)) (ord-spmf (=)) } F$*

and $g: \bigwedge x. \text{monotone } (\text{ord-spmf } (=)) (\text{ord-spmf } (=)) G$
and $\text{param}: (\text{rel-spmf } R ==> \text{rel-spmf } R) F G$
shows $(\text{rel-spmf } R) (\text{ccpo.fixp } \text{lub-spmf } (\text{ord-spmf } (=)) F) (\text{ccpo.fixp } \text{lub-spmf } (\text{ord-spmf } (=)) G)$
 $\langle \text{proof} \rangle$

lemma *fixp-spmf-parametric*:

assumes $f: \bigwedge x. \text{mono-spmf } (\lambda f. F f x)$
and $g: \bigwedge x. \text{mono-spmf } (\lambda f. G f x)$
and $\text{param}: ((A ==> \text{rel-spmf } R) ==> A ==> \text{rel-spmf } R) F G$
shows $(A ==> \text{rel-spmf } R) (\text{spmfixp-fun } F) (\text{spmfixp-fun } G)$
 $\langle \text{proof} \rangle$

end

end

end

25.12 Restrictions on spmfs

definition *restrict-spmf* :: $'a \text{ spmf} \Rightarrow 'a \text{ set} \Rightarrow 'a \text{ spmf}$ (**infixl** $\langle \upharpoonright \rangle$ 110)

where $p \upharpoonright A = \text{map-pmf } (\lambda x. x \gg= (\lambda y. \text{if } y \in A \text{ then Some } y \text{ else None})) p$

lemma *set-restrict-spmf* [simp]: $\text{set-spmf } (p \upharpoonright A) = \text{set-spmf } p \cap A$
 $\langle \text{proof} \rangle$

lemma *restrict-map-spmf*: $\text{map-spmf } f p \upharpoonright A = \text{map-spmf } f (p \upharpoonright (f^{-1} A))$
 $\langle \text{proof} \rangle$

lemma *restrict-restrict-spmf* [simp]: $p \upharpoonright A \upharpoonright B = p \upharpoonright (A \cap B)$
 $\langle \text{proof} \rangle$

lemma *restrict-spmf-empty* [simp]: $p \upharpoonright \{\} = \text{return-pmf } \text{None}$
 $\langle \text{proof} \rangle$

lemma *restrict-spmf-UNIV* [simp]: $p \upharpoonright \text{UNIV} = p$
 $\langle \text{proof} \rangle$

lemma *spmfixp-restrict-spmf-outside* [simp]: $x \notin A \implies \text{spmfixp } (p \upharpoonright A) x = 0$
 $\langle \text{proof} \rangle$

lemma *emeasure-restrict-spmf* [simp]: $\text{emeasure } (\text{measure-spmf } (p \upharpoonright A)) X = \text{emeasure } (\text{measure-spmf } p) (X \cap A)$
 $\langle \text{proof} \rangle$

lemma *measure-restrict-spmf* [simp]:
 $\text{measure } (\text{measure-spmf } (p \upharpoonright A)) X = \text{measure } (\text{measure-spmf } p) (X \cap A)$
 $\langle \text{proof} \rangle$

lemma *spmf-restrict-spmf*: $\text{spmf } (p \restriction A) x = (\text{if } x \in A \text{ then } \text{spmf } p x \text{ else } 0)$
 $\langle \text{proof} \rangle$

lemma *spmf-restrict-spmf-inside* [simp]: $x \in A \implies \text{spmf } (p \restriction A) x = \text{spmf } p x$
 $\langle \text{proof} \rangle$

lemma *pmf-restrict-spmf-None*: $\text{pmf } (p \restriction A) \text{ None} = \text{pmf } p \text{ None} + \text{measure } (\text{measure-spmf } p) (- A)$
 $\langle \text{proof} \rangle$

lemma *restrict-spmf-trivial*: $(\bigwedge x. x \in \text{set-spmf } p \implies x \in A) \implies p \restriction A = p$
 $\langle \text{proof} \rangle$

lemma *restrict-spmf-trivial'*: $\text{set-spmf } p \subseteq A \implies p \restriction A = p$
 $\langle \text{proof} \rangle$

lemma *restrict-return-spmf*: $\text{return-spmf } x \restriction A = (\text{if } x \in A \text{ then } \text{return-spmf } x \text{ else } \text{return-pmf None})$
 $\langle \text{proof} \rangle$

lemma *restrict-return-spmf-inside* [simp]: $x \in A \implies \text{return-spmf } x \restriction A = \text{return-spmf } x$
 $\langle \text{proof} \rangle$

lemma *restrict-return-spmf-outside* [simp]: $x \notin A \implies \text{return-spmf } x \restriction A = \text{return-pmf None}$
 $\langle \text{proof} \rangle$

lemma *restrict-spmf-return-pmf-None* [simp]: $\text{return-pmf None} \restriction A = \text{return-pmf None}$
 $\langle \text{proof} \rangle$

lemma *restrict-bind-pmf*: $\text{bind-pmf } p g \restriction A = p \gg (\lambda x. g x \restriction A)$
 $\langle \text{proof} \rangle$

lemma *restrict-bind-spmf*: $\text{bind-spmf } p g \restriction A = p \gg (\lambda x. g x \restriction A)$
 $\langle \text{proof} \rangle$

lemma *bind-restrict-pmf*: $\text{bind-pmf } (p \restriction A) g = p \gg (\lambda x. \text{if } x \in \text{Some } A \text{ then } g x \text{ else } g \text{ None})$
 $\langle \text{proof} \rangle$

lemma *bind-restrict-spmf*: $\text{bind-spmf } (p \restriction A) g = p \gg (\lambda x. \text{if } x \in A \text{ then } g x \text{ else } \text{return-pmf None})$
 $\langle \text{proof} \rangle$

lemma *spmf-map-restrict*: $\text{spmf } (\text{map-spmf fst } (p \restriction (\text{snd} - \{y\}))) x = \text{spmf } p (x, y)$

$\langle \text{proof} \rangle$

lemma *measure-eqI-restrict-spmf*:

assumes *rel-spmf* *R* (*restrict-spmf* *p* *A*) (*restrict-spmf* *q* *B*)

shows *measure* (*measure-spmf* *p*) *A* = *measure* (*measure-spmf* *q*) *B*

$\langle \text{proof} \rangle$

25.13 Subprobability distributions of sets

definition *spmf-of-set* :: 'a set \Rightarrow 'a *spmf*

where

spmf-of-set *A* = (if *finite* *A* \wedge *A* \neq {} then *spmf-of-pmf* (*pmf-of-set* *A*) else *return-pmf* *None*)

lemma *spmf-of-set*: *spmf* (*spmf-of-set* *A*) *x* = *indicator* *A* *x* / *card* *A*

$\langle \text{proof} \rangle$

lemma *pmf-spmf-of-set-None* [*simp*]: *pmf* (*spmf-of-set* *A*) *None* = *indicator* {*A*.
infinite *A* \vee *A* = {}} *A*

$\langle \text{proof} \rangle$

lemma *set-spmf-of-set*: *set-spmf* (*spmf-of-set* *A*) = (if *finite* *A* then *A* else {})

$\langle \text{proof} \rangle$

lemma *set-spmf-of-set-finite* [*simp*]: *finite* *A* \Longrightarrow *set-spmf* (*spmf-of-set* *A*) = *A*

$\langle \text{proof} \rangle$

lemma *spmf-of-set-singleton*: *spmf-of-set* {*x*} = *return-spmf* *x*

$\langle \text{proof} \rangle$

lemma *map-spmf-of-set-inj-on* [*simp*]:

inj-on *f* *A* \Longrightarrow *map-spmf* *f* (*spmf-of-set* *A*) = *spmf-of-set* (*f* ‘ *A*)

$\langle \text{proof} \rangle$

lemma *spmf-of-pmf-pmf-of-set* [*simp*]:

$\llbracket \text{finite } A; A \neq \{\} \rrbracket \Longrightarrow \text{spmf-of-pmf } (\text{pmf-of-set } A) = \text{spmf-of-set } A$

$\langle \text{proof} \rangle$

lemma *weight-spmf-of-set*:

weight-spmf (*spmf-of-set* *A*) = (if *finite* *A* \wedge *A* \neq {} then 1 else 0)

$\langle \text{proof} \rangle$

lemma *weight-spmf-of-set-finite* [*simp*]: $\llbracket \text{finite } A; A \neq \{\} \rrbracket \Longrightarrow \text{weight-spmf}$
(*spmf-of-set* *A*) = 1

$\langle \text{proof} \rangle$

lemma *weight-spmf-of-set-infinite* [*simp*]: *infinite* *A* \Longrightarrow *weight-spmf* (*spmf-of-set*
A) = 0

$\langle \text{proof} \rangle$

lemma *measure-spmf-spmf-of-set:*

measure-spmf (spm-f-of-set A) = (if finite A \wedge A \neq {} then measure-pmf (pmf-of-set A) else null-measure (count-space UNIV))
<proof>

lemma *emeasure-spmf-of-set:*

emeasure (measure-spmf (spm-f-of-set S)) A = card (S \cap A) / card S
<proof>

lemma *measure-spmf-of-set:*

measure (measure-spmf (spm-f-of-set S)) A = card (S \cap A) / card S
<proof>

lemma *nn-integral-spmf-of-set:* *nn-integral (measure-spmf (spm-f-of-set A)) f = sum f A / card A*
<proof>

lemma *integral-spmf-of-set:* *integral^L (measure-spmf (spm-f-of-set A)) f = sum f A / card A*
<proof>

notepad begin — *pmf-of-set* is not fully parametric.

<proof>

end

lemma *rel-pmf-of-set-bij:*

assumes f: bij-betw f A B
and A: A \neq {} finite A
and B: B \neq {} finite B
and R: $\bigwedge x. x \in A \implies R\ x\ (f\ x)$
shows rel-pmf R (pmf-of-set A) (pmf-of-set B)
<proof>

lemma *rel-spmf-of-set-bij:*

assumes f: bij-betw f A B
and R: $\bigwedge x. x \in A \implies R\ x\ (f\ x)$
shows rel-spmf R (spm-f-of-set A) (spm-f-of-set B)
<proof>

context includes *lifting-syntax*

begin

lemma *rel-spmf-of-set:*

assumes bi-unique R
shows (rel-set R \implies rel-spmf R) spm-f-of-set spm-f-of-set
<proof>

end

lemma *map-mem-spmf-of-set*:

assumes *finite B B ≠ {}*

shows *map-spmf (λx. x ∈ A) (spmf-of-set B) = spmf-of-pmf (bernoulli-pmf (card (A ∩ B) / card B))*

(is ?lhs = ?rhs)

<proof>

abbreviation *coin-spmf* :: *bool spmf*

where *coin-spmf* ≡ *spmf-of-set UNIV*

lemma *map-eq-const-coin-spmf*: *map-spmf ((=) c) coin-spmf = coin-spmf*

<proof>

lemma *bind-coin-spmf-eq-const*: *coin-spmf ≫= (λx :: bool. return-spmf (b = x))*

= coin-spmf

<proof>

lemma *bind-coin-spmf-eq-const'*: *coin-spmf ≫= (λx :: bool. return-spmf (x = b))*

= coin-spmf

<proof>

25.14 Losslessness

definition *lossless-spmf* :: *'a spmf ⇒ bool*

where *lossless-spmf p* \longleftrightarrow *weight-spmf p = 1*

lemma *lossless-iff-pmf-None*: *lossless-spmf p* \longleftrightarrow *pmf p None = 0*

<proof>

lemma *lossless-return-spmf [iff]*: *lossless-spmf (return-spmf x)*

<proof>

lemma *lossless-return-pmf-None [iff]*: \neg *lossless-spmf (return-pmf None)*

<proof>

lemma *lossless-map-spmf [simp]*: *lossless-spmf (map-spmf f p)* \longleftrightarrow *lossless-spmf p*

<proof>

lemma *lossless-bind-spmf [simp]*:

lossless-spmf (p ≫= f) \longleftrightarrow *lossless-spmf p* \wedge ($\forall x \in \text{set-spmf } p. \text{lossless-spmf (f x)}$)

<proof>

lemma *lossless-weight-spmfD*: *lossless-spmf p* \implies *weight-spmf p = 1*

<proof>

lemma *lossless-iff-set-pmf-None*:

$lossless\text{-}spmf\ p \longleftrightarrow None \notin set\text{-}pmf\ p$
 $\langle proof \rangle$

lemma $lossless\text{-}spmf\text{-}of\text{-}set$ [simp]: $lossless\text{-}spmf\ (spmf\text{-}of\text{-}set\ A) \longleftrightarrow finite\ A \wedge A \neq \{\}$
 $\langle proof \rangle$

lemma $lossless\text{-}spmf\text{-}spmf\text{-}of\text{-}spmf$ [simp]: $lossless\text{-}spmf\ (spmf\text{-}of\text{-}pmf\ p)$
 $\langle proof \rangle$

lemma $lossless\text{-}spmf\text{-}bind\text{-}pmf$ [simp]:
 $lossless\text{-}spmf\ (bind\text{-}pmf\ p\ f) \longleftrightarrow (\forall x \in set\text{-}pmf\ p. lossless\text{-}spmf\ (f\ x))$
 $\langle proof \rangle$

lemma $lossless\text{-}spmf\text{-}conv\text{-}spmf\text{-}of\text{-}pmf$: $lossless\text{-}spmf\ p \longleftrightarrow (\exists p'. p = spmf\text{-}of\text{-}pmf\ p')$
 $\langle proof \rangle$

lemma $spmf\text{-}False\text{-}conv\text{-}True$: $lossless\text{-}spmf\ p \implies spmf\ p\ False = 1 - spmf\ p\ True$
 $\langle proof \rangle$

lemma $spmf\text{-}True\text{-}conv\text{-}False$: $lossless\text{-}spmf\ p \implies spmf\ p\ True = 1 - spmf\ p\ False$
 $\langle proof \rangle$

lemma $bind\text{-}eq\text{-}return\text{-}spmf$:
 $bind\text{-}spmf\ p\ f = return\text{-}spmf\ x \longleftrightarrow (\forall y \in set\text{-}spmf\ p. f\ y = return\text{-}spmf\ x) \wedge lossless\text{-}spmf\ p$
 $\langle proof \rangle$

lemma $rel\text{-}spmf\text{-}return\text{-}spmf2$:
 $rel\text{-}spmf\ R\ p\ (return\text{-}spmf\ x) \longleftrightarrow lossless\text{-}spmf\ p \wedge (\forall a \in set\text{-}spmf\ p. R\ a\ x)$
 $\langle proof \rangle$

lemma $rel\text{-}spmf\text{-}return\text{-}spmf1$:
 $rel\text{-}spmf\ R\ (return\text{-}spmf\ x)\ p \longleftrightarrow lossless\text{-}spmf\ p \wedge (\forall a \in set\text{-}spmf\ p. R\ x\ a)$
 $\langle proof \rangle$

lemma $rel\text{-}spmf\text{-}bindI1$:
assumes $f: \bigwedge x. x \in set\text{-}spmf\ p \implies rel\text{-}spmf\ R\ (f\ x)\ q$
and $p: lossless\text{-}spmf\ p$
shows $rel\text{-}spmf\ R\ (bind\text{-}spmf\ p\ f)\ q$
 $\langle proof \rangle$

lemma $rel\text{-}spmf\text{-}bindI2$:
 $\llbracket \bigwedge x. x \in set\text{-}spmf\ q \implies rel\text{-}spmf\ R\ p\ (f\ x); lossless\text{-}spmf\ q \rrbracket$
 $\implies rel\text{-}spmf\ R\ p\ (bind\text{-}spmf\ q\ f)$
 $\langle proof \rangle$

25.15 Scaling

definition $scale\text{-}spmf :: real \Rightarrow 'a\ spmf \Rightarrow 'a\ spmf$

where

$scale\text{-}spmf\ r\ p = embed\text{-}spmf\ (\lambda x. min\ (inverse\ (weight\text{-}spmf\ p))\ (max\ 0\ r))\ * spmf\ p\ x)$

lemma $scale\text{-}spmf\text{-}le\text{-}1$:

$(\int^+ x. min\ (inverse\ (weight\text{-}spmf\ p))\ (max\ 0\ r))\ * spmf\ p\ x\ \partial count\text{-}space\ UNIV) \leq 1$ (is ?lhs \leq -)
 $\langle proof \rangle$

lemma $spmf\text{-}scale\text{-}spmf$: $spmf\ (scale\text{-}spmf\ r\ p)\ x = max\ 0\ (min\ (inverse\ (weight\text{-}spmf\ p))\ r)\ * spmf\ p\ x$ (is ?lhs = ?rhs)

$\langle proof \rangle$

lemma $real\text{-}inverse\text{-}le\text{-}1\text{-}iff$: **fixes** $x :: real$

shows $\llbracket 0 \leq x; x \leq 1 \rrbracket \implies 1 / x \leq 1 \longleftrightarrow x = 1 \vee x = 0$

$\langle proof \rangle$

lemma $spmf\text{-}scale\text{-}spmf'$: $r \leq 1 \implies spmf\ (scale\text{-}spmf\ r\ p)\ x = max\ 0\ r\ * spmf\ p\ x$

$\langle proof \rangle$

lemma $scale\text{-}spmf\text{-}neg$: $r \leq 0 \implies scale\text{-}spmf\ r\ p = return\text{-}pmf\ None$

$\langle proof \rangle$

lemma $scale\text{-}spmf\text{-}return\text{-}None$ [simp]: $scale\text{-}spmf\ r\ (return\text{-}pmf\ None) = return\text{-}pmf\ None$

$\langle proof \rangle$

lemma $scale\text{-}spmf\text{-}conv\text{-}bind\text{-}bernoulli$:

assumes $r \leq 1$

shows $scale\text{-}spmf\ r\ p = bind\text{-}pmf\ (bernoulli\text{-}pmf\ r)\ (\lambda b. if\ b\ then\ p\ else\ return\text{-}pmf\ None)$ (is ?lhs = ?rhs)

$\langle proof \rangle$

lemma $nn\text{-}integral\text{-}spmf$: $(\int^+ x. spmf\ p\ x\ \partial count\text{-}space\ A) = emeasure\ (measure\text{-}spmf\ p)\ A$

$\langle proof \rangle$

lemma $measure\text{-}spmf\text{-}scale\text{-}spmf$: $measure\text{-}spmf\ (scale\text{-}spmf\ r\ p) = scale\text{-}measure\ (min\ (inverse\ (weight\text{-}spmf\ p))\ r)\ (measure\text{-}spmf\ p)$

$\langle proof \rangle$

lemma $measure\text{-}spmf\text{-}scale\text{-}spmf'$:

assumes $r \leq 1$

shows $measure\text{-}spmf\ (scale\text{-}spmf\ r\ p) = scale\text{-}measure\ r\ (measure\text{-}spmf\ p)$

$\langle proof \rangle$

lemma *scale-spmf-1* [simp]: *scale-spmf 1 p = p*
 ⟨proof⟩

lemma *scale-spmf-0* [simp]: *scale-spmf 0 p = return-pmf None*
 ⟨proof⟩

lemma *bind-scale-spmf*:
 assumes $r: r \leq 1$
 shows *bind-spmf (scale-spmf r p) f = bind-spmf p ($\lambda x. \text{scale-spmf } r \text{ (f } x)$)*
 (is ?lhs = ?rhs)
 ⟨proof⟩

lemma *scale-bind-spmf*:
 assumes $r \leq 1$
 shows *scale-spmf r (bind-spmf p f) = bind-spmf p ($\lambda x. \text{scale-spmf } r \text{ (f } x)$)*
 (is ?lhs = ?rhs)
 ⟨proof⟩

lemma *bind-spmf-const*: *bind-spmf p ($\lambda x. q$) = scale-spmf (weight-spmf p) q* (is ?lhs = ?rhs)
 ⟨proof⟩

lemma *map-scale-spmf*: *map-spmf f (scale-spmf r p) = scale-spmf r (map-spmf f p)* (is ?lhs = ?rhs)
 ⟨proof⟩

lemma *set-scale-spmf*: *set-spmf (scale-spmf r p) = (if $r > 0$ then set-spmf p else {})*
 ⟨proof⟩

lemma *set-scale-spmf'* [simp]: $0 < r \implies \text{set-spmf (scale-spmf } r \text{ p)} = \text{set-spmf p}$
 ⟨proof⟩

lemma *rel-spmf-scaleI*:
 assumes $r > 0 \implies \text{rel-spmf } A \text{ p } q$
 shows *rel-spmf A (scale-spmf r p) (scale-spmf r q)*
 ⟨proof⟩

lemma *weight-scale-spmf*: *weight-spmf (scale-spmf r p) = min 1 (max 0 r * weight-spmf p)*
 ⟨proof⟩

lemma *weight-scale-spmf'* [simp]:
 $\llbracket 0 \leq r; r \leq 1 \rrbracket \implies \text{weight-spmf (scale-spmf } r \text{ p)} = r * \text{weight-spmf p}$
 ⟨proof⟩

lemma *pmf-scale-spmf-None*:
*pmf (scale-spmf k p) None = 1 - min 1 (max 0 k * (1 - pmf p None))*
 ⟨proof⟩

lemma *scale-scale-spmf*:

$\text{scale-spmf } r \ (\text{scale-spmf } r' \ p) = \text{scale-spmf } (r * \max 0 \ (\min (\text{inverse } (\text{weight-spmf } p)) \ r')) \ p$
 (is ?lhs = ?rhs)
 <proof>

lemma *scale-scale-spmf' [simp]*:

assumes $0 \leq r \ r \leq 1 \ 0 \leq r' \ r' \leq 1$
shows $\text{scale-spmf } r \ (\text{scale-spmf } r' \ p) = \text{scale-spmf } (r * r') \ p$
 <proof>

lemma *scale-spmf-eq-same*: $\text{scale-spmf } r \ p = p \longleftrightarrow \text{weight-spmf } p = 0 \vee r = 1 \vee r \geq 1 \wedge \text{weight-spmf } p = 1$

(is ?lhs \longleftrightarrow ?rhs)
 <proof>

lemma *map-const-spmf-of-set*:

$\llbracket \text{finite } A; A \neq \{\} \rrbracket \implies \text{map-spmf } (\lambda \cdot. c) \ (\text{spm-f-of-set } A) = \text{return-spmf } c$
 <proof>

25.16 Conditional spmfs

lemma *set-pmf-Int-Some*: $\text{set-pmf } p \cap \text{Some } 'A = \{\} \longleftrightarrow \text{set-spmf } p \cap A = \{\}$
 <proof>

lemma *measure-spmf-zero-iff*: $\text{measure } (\text{measure-spmf } p) \ A = 0 \longleftrightarrow \text{set-spmf } p \cap A = \{\}$
 <proof>

definition *cond-spmf* :: $'a \ \text{spm-f} \Rightarrow 'a \ \text{set} \Rightarrow 'a \ \text{spm-f}$

where $\text{cond-spmf } p \ A = (\text{if } \text{set-spmf } p \cap A = \{\} \text{ then } \text{return-pmf } \text{None} \text{ else } \text{cond-pmf } p \ (\text{Some } 'A))$

lemma *set-cond-spmf [simp]*: $\text{set-spmf } (\text{cond-spmf } p \ A) = \text{set-spmf } p \cap A$
 <proof>

lemma *cond-map-spmf [simp]*: $\text{cond-spmf } (\text{map-spmf } f \ p) \ A = \text{map-spmf } f \ (\text{cond-spmf } p \ (f \ - 'A))$
 <proof>

lemma *spm-f-cond-spmf [simp]*:

$\text{spm-f } (\text{cond-spmf } p \ A) \ x = (\text{if } x \in A \text{ then } \text{spm-f } p \ x / \text{measure } (\text{measure-spmf } p) \ A \text{ else } 0)$
 <proof>

lemma *bind-eq-return-pmf-None*:

$\text{bind-spmf } p \ f = \text{return-pmf } \text{None} \longleftrightarrow (\forall x \in \text{set-spmf } p. f \ x = \text{return-pmf } \text{None})$
 <proof>

lemma *return-pmf-None-eq-bind*:

$\text{return-pmf None} = \text{bind-spmf } p \ f \longleftrightarrow (\forall x \in \text{set-spmf } p. f \ x = \text{return-pmf None})$
 $\langle \text{proof} \rangle$

25.17 Product spmf

definition *pair-spmf* :: $'a \text{ spmf} \Rightarrow 'b \text{ spmf} \Rightarrow ('a \times 'b) \text{ spmf}$

where $\text{pair-spmf } p \ q = \text{bind-pmf } (\text{pair-pmf } p \ q) \ (\lambda xy. \text{case } xy \text{ of } (\text{Some } x, \text{Some } y) \Rightarrow \text{return-spmf } (x, y) \mid - \Rightarrow \text{return-pmf None})$

lemma *map-fst-pair-spmf* [simp]: $\text{map-spmf fst } (\text{pair-spmf } p \ q) = \text{scale-spmf } (\text{weight-spmf } q) \ p$
 $\langle \text{proof} \rangle$

lemma *map-snd-pair-spmf* [simp]: $\text{map-spmf snd } (\text{pair-spmf } p \ q) = \text{scale-spmf } (\text{weight-spmf } p) \ q$
 $\langle \text{proof} \rangle$

lemma *set-pair-spmf* [simp]: $\text{set-spmf } (\text{pair-spmf } p \ q) = \text{set-spmf } p \times \text{set-spmf } q$
 $\langle \text{proof} \rangle$

lemma *spm-f-pair* [simp]: $\text{spm-f } (\text{pair-spmf } p \ q) \ (x, y) = \text{spm-f } p \ x * \text{spm-f } q \ y$ (is ?lhs = ?rhs)
 $\langle \text{proof} \rangle$

lemma *pair-map-spmf2*: $\text{pair-spmf } p \ (\text{map-spmf } f \ q) = \text{map-spmf } (\text{apsnd } f) \ (\text{pair-spmf } p \ q)$
 $\langle \text{proof} \rangle$

lemma *pair-map-spmf1*: $\text{pair-spmf } (\text{map-spmf } f \ p) \ q = \text{map-spmf } (\text{apfst } f) \ (\text{pair-spmf } p \ q)$
 $\langle \text{proof} \rangle$

lemma *pair-map-spmf*: $\text{pair-spmf } (\text{map-spmf } f \ p) \ (\text{map-spmf } g \ q) = \text{map-spmf } (\text{map-prod } f \ g) \ (\text{pair-spmf } p \ q)$
 $\langle \text{proof} \rangle$

lemma *pair-spmf-alt-def*: $\text{pair-spmf } p \ q = \text{bind-spmf } p \ (\lambda x. \text{bind-spmf } q \ (\lambda y. \text{return-spmf } (x, y)))$
 $\langle \text{proof} \rangle$

lemma *weight-pair-spmf* [simp]: $\text{weight-spmf } (\text{pair-spmf } p \ q) = \text{weight-spmf } p * \text{weight-spmf } q$
 $\langle \text{proof} \rangle$

lemma *pair-scale-spmf1*:

$r \leq 1 \implies \text{pair-spmf } (\text{scale-spmf } r \ p) \ q = \text{scale-spmf } r \ (\text{pair-spmf } p \ q)$
 $\langle \text{proof} \rangle$

lemma *pair-scale-spmf2*:

$$r \leq 1 \implies \text{pair-spmf } p \text{ (scale-spmf } r \text{ } q) = \text{scale-spmf } r \text{ (pair-spmf } p \text{ } q)$$

<proof>

lemma *pair-spmf-return-None1* [simp]: *pair-spmf (return-pmf None) p = return-pmf None*

<proof>

lemma *pair-spmf-return-None2* [simp]: *pair-spmf p (return-pmf None) = return-pmf None*

<proof>

lemma *pair-spmf-return-spmf1*: *pair-spmf (return-spmf x) q = map-spmf (Pair x) q*

<proof>

lemma *pair-spmf-return-spmf2*: *pair-spmf p (return-spmf y) = map-spmf ($\lambda x. (x, y)$) p*

<proof>

lemma *pair-spmf-return-spmf* [simp]: *pair-spmf (return-spmf x) (return-spmf y) = return-spmf (x, y)*

<proof>

lemma *rel-pair-spmf-prod*:

$$\begin{aligned} & \text{rel-spmf (rel-prod } A \text{ } B) \text{ (pair-spmf } p \text{ } q) \text{ (pair-spmf } p' \text{ } q') \longleftrightarrow \\ & \text{rel-spmf } A \text{ (scale-spmf (weight-spmf } q) \text{ } p) \text{ (scale-spmf (weight-spmf } q') \text{ } p') \wedge \\ & \text{rel-spmf } B \text{ (scale-spmf (weight-spmf } p) \text{ } q) \text{ (scale-spmf (weight-spmf } p') \text{ } q')} \\ & \text{(is ?lhs } \longleftrightarrow \text{ ?rhs is - } \longleftrightarrow \text{ ?} A \wedge \text{ ?} B \text{ is - } \longleftrightarrow \text{ rel-spmf - ?} p \text{ ?} p' \wedge \text{ rel-spmf - ?} q \text{ ?} q') \end{aligned}$$

<proof>

lemma *pair-pair-spmf*:

$$\text{pair-spmf (pair-spmf } p \text{ } q) \text{ } r = \text{map-spmf } (\lambda(x, (y, z)). ((x, y), z)) \text{ (pair-spmf } p \text{ (pair-spmf } q \text{ } r))$$

<proof>

lemma *pair-commute-spmf*:

$$\text{pair-spmf } p \text{ } q = \text{map-spmf } (\lambda(y, x). (x, y)) \text{ (pair-spmf } q \text{ } p)$$

<proof>

25.18 Assertions

definition *assert-spmf* :: *bool* \Rightarrow *unit* *spmf*

where *assert-spmf* *b* = (*if* *b* *then* *return-spmf* () *else* *return-pmf* None)

lemma *assert-spmf-simps* [simp]:

$$\text{assert-spmf True} = \text{return-spmf } ()$$

assert-spmf False = return-pmf None
<proof>

lemma *in-set-assert-spmf* [simp]: $x \in \text{set-spmf } (\text{assert-spmf } p) \longleftrightarrow p$
<proof>

lemma *set-spmf-assert-spmf-eq-empty* [simp]: $\text{set-spmf } (\text{assert-spmf } b) = \{\} \longleftrightarrow \neg b$
<proof>

lemma *lossless-assert-spmf* [iff]: $\text{lossless-spmf } (\text{assert-spmf } b) \longleftrightarrow b$
<proof>

25.19 Try

definition *try-spmf* :: $'a \text{ spmf} \Rightarrow 'a \text{ spmf} \Rightarrow 'a \text{ spmf}$
 ($\langle \langle \text{open-block notation} = \langle \text{mixfix try-spmf} \rangle \rangle \text{TRY - ELSE -} \rangle [0,60] 59$)

where *TRY* p *ELSE* $q = \text{bind-pmf } p (\lambda x. \text{case } x \text{ of None} \Rightarrow q \mid \text{Some } y \Rightarrow \text{return-spmf } y)$

lemma *try-spmf-lossless* [simp]:
assumes *lossless-spmf* p
shows *TRY* p *ELSE* $q = p$
<proof>

lemma *try-spmf-return-spmf1*: *TRY* *return-spmf* x *ELSE* $q = \text{return-spmf } x$
<proof>

lemma *try-spmf-return-None* [simp]: *TRY* *return-pmf* *None* *ELSE* $q = q$
<proof>

lemma *try-spmf-return-pmf-None2* [simp]: *TRY* p *ELSE* *return-pmf* *None* = p
<proof>

lemma *map-try-spmf*: $\text{map-spmf } f (\text{try-spmf } p \ q) = \text{try-spmf } (\text{map-spmf } f \ p)$
 ($\text{map-spmf } f \ q$)
<proof>

lemma *try-spmf-bind-pmf*: *TRY* (*bind-pmf* $p \ f$) *ELSE* $q = \text{bind-pmf } p (\lambda x. \text{TRY } (f \ x) \text{ ELSE } q)$
<proof>

lemma *try-spmf-bind-spmf-lossless*:
 $\text{lossless-spmf } p \Longrightarrow \text{TRY } (\text{bind-spmf } p \ f) \text{ ELSE } q = \text{bind-spmf } p (\lambda x. \text{TRY } (f \ x) \text{ ELSE } q)$
<proof>

lemma *try-spmf-bind-out*:
 $\text{lossless-spmf } p \Longrightarrow \text{bind-spmf } p (\lambda x. \text{TRY } (f \ x) \text{ ELSE } q) = \text{TRY } (\text{bind-spmf } p \ f) \text{ ELSE } q$

f) *ELSE* q
 $\langle \text{proof} \rangle$

lemma *lossless-try-spmf* [simp]:
 $\text{lossless-spmf } (\text{TRY } p \text{ ELSE } q) \longleftrightarrow \text{lossless-spmf } p \vee \text{lossless-spmf } q$
 $\langle \text{proof} \rangle$

context includes *lifting-syntax*
begin

lemma *try-spmf-parametric* [transfer-rule]:
 $(\text{rel-spmf } A \Longrightarrow \text{rel-spmf } A \Longrightarrow \text{rel-spmf } A) \text{ try-spmf try-spmf}$
 $\langle \text{proof} \rangle$

end

lemma *try-spmf-cong*:
 $\llbracket p = p'; \neg \text{lossless-spmf } p' \Longrightarrow q = q' \rrbracket \Longrightarrow \text{TRY } p \text{ ELSE } q = \text{TRY } p' \text{ ELSE } q'$
 $\langle \text{proof} \rangle$

lemma *rel-spmf-try-spmf*:
 $\llbracket \text{rel-spmf } R \text{ } p \text{ } p'; \neg \text{lossless-spmf } p' \Longrightarrow \text{rel-spmf } R \text{ } q \text{ } q' \rrbracket$
 $\Longrightarrow \text{rel-spmf } R \text{ } (\text{TRY } p \text{ ELSE } q) \text{ } (\text{TRY } p' \text{ ELSE } q')$
 $\langle \text{proof} \rangle$

lemma *spmf-try-spmf*:
 $\text{spmf } (\text{TRY } p \text{ ELSE } q) \text{ } x = \text{spmf } p \text{ } x + \text{pmf } p \text{ None} * \text{spmf } q \text{ } x$
 $\langle \text{proof} \rangle$

lemma *try-scale-spmf-same* [simp]: $\text{lossless-spmf } p \Longrightarrow \text{TRY } \text{scale-spmf } k \text{ } p \text{ ELSE } p = p$
 $\langle \text{proof} \rangle$

lemma *pmf-try-spmf-None* [simp]: $\text{pmf } (\text{TRY } p \text{ ELSE } q) \text{ None} = \text{pmf } p \text{ None} * \text{pmf } q \text{ None}$ (is ?lhs = ?rhs)
 $\langle \text{proof} \rangle$

lemma *try-bind-spmf-lossless2*:
 $\text{lossless-spmf } q \Longrightarrow \text{TRY } (\text{bind-spmf } p \text{ } f) \text{ ELSE } q = \text{TRY } (p \gg (\lambda x. \text{TRY } (f \text{ } x) \text{ ELSE } q)) \text{ ELSE } q$
 $\langle \text{proof} \rangle$

lemma *try-bind-spmf-lossless2'*:
fixes $f :: 'a \Rightarrow 'b \text{ spmf}$ **shows**
 $\llbracket \text{NO-MATCH } (\lambda x :: 'a. \text{try-spmf } (g \text{ } x :: 'b \text{ spmf}) \text{ } (h \text{ } x)) \text{ } f; \text{lossless-spmf } q \rrbracket$
 $\Longrightarrow \text{TRY } (\text{bind-spmf } p \text{ } f) \text{ ELSE } q = \text{TRY } (p \gg (\lambda x :: 'a. \text{TRY } (f \text{ } x) \text{ ELSE } q))$
 $\text{ELSE } q$
 $\langle \text{proof} \rangle$

lemma *try-bind-assert-spmf*:

TRY (*assert-spmf* $b \gg= f$) *ELSE* $q = (\text{if } b \text{ then } \text{TRY } (f \ ()) \text{ ELSE } q \text{ else } q)$
 $\langle \text{proof} \rangle$

25.20 Miscellaneous

lemma *assumes* *rel-spmf* $(\lambda x y. \text{bad1 } x = \text{bad2 } y \wedge (\neg \text{bad2 } y \longrightarrow A \ x \longleftrightarrow B \ y)) \ p \ q$ (**is** *rel-spmf* $?A \ - \ -$)

shows *fundamental-lemma-bad*: $\text{measure } (\text{measure-spmf } p) \ \{x. \text{bad1 } x\} = \text{measure } (\text{measure-spmf } q) \ \{y. \text{bad2 } y\}$ (**is** $?bad$)

and *fundamental-lemma*: $|\text{measure } (\text{measure-spmf } p) \ \{x. A \ x\} - \text{measure } (\text{measure-spmf } q) \ \{y. B \ y\}| \leq$

$\text{measure } (\text{measure-spmf } p) \ \{x. \text{bad1 } x\}$ (**is** $?fundamental$)
 $\langle \text{proof} \rangle$

end

26 Indexed products of PMFs

theory *Product-PMF*

imports *Probability-Mass-Function Independent-Family*
begin

Conflicting notation from *HOL–Analysis.Infinite-Sum*

no-notation *Infinite-Sum.abs-summable-on* (**infixr** $\langle \text{abs}'\text{-summable}'\text{-on} \rangle \ 46$)

26.1 Preliminaries

lemma *pmf-expectation-eq-infsetsum*: $\text{measure-pmf.expectation } p \ f = \text{infsetsum } (\lambda x. \text{pmf } p \ x * f \ x) \ \text{UNIV}$
 $\langle \text{proof} \rangle$

lemma *measure-pmf-prob-product*:

assumes *countable* A *countable* B

shows $\text{measure-pmf.prob } (\text{pair-pmf } M \ N) \ (A \times B) = \text{measure-pmf.prob } M \ A * \text{measure-pmf.prob } N \ B$

$\langle \text{proof} \rangle$

26.2 Definition

In analogy to Pi_M , we define an indexed product of PMFs. In the literature, this is typically called taking a vector of independent random variables. Note that the components do not have to be identically distributed.

The operation takes an explicit index set A and a function f that maps each element from A to a PMF and defines the product measure $\bigotimes_{i \in A} f(i)$, which is represented as a $(\text{'a} \Rightarrow \text{'b}) \text{ pmf}$.

Note that unlike Pi_M , this only works for *finite* index sets. It could be extended to countable sets and beyond, but the construction becomes somewhat more involved.

definition $Pi\text{-}pmf :: 'a \text{ set} \Rightarrow 'b \Rightarrow ('a \Rightarrow 'b \text{ pmf}) \Rightarrow ('a \Rightarrow 'b) \text{ pmf}$ **where**

$Pi\text{-}pmf \ A \ dflt \ p =$
 $\text{embed-pmf } (\lambda f. \text{ if } (\forall x. x \notin A \longrightarrow f \ x = dflt) \text{ then } \prod_{x \in A}. \text{ pmf } (p \ x) (f \ x)$
 $\text{ else } 0)$

A technical subtlety that needs to be addressed is this: Intuitively, the functions in the support of a product distribution have domain A . However, since HOL is a total logic, these functions must still return *some* value for inputs outside A . The product measure Pi_M simply lets these functions return *undefined* in these cases. We chose a different solution here, which is to supply a default value $dflt$ that is returned in these cases.

As one possible application, one could model the result of n different independent coin tosses as $Pi\text{-}pmf \ \{0..<n\} \ False \ (\lambda-. \text{ bernoulli-pmf } (1 / 2))$. This returns a function of type $\text{nat} \Rightarrow \text{bool}$ that maps every natural number below n to the result of the corresponding coin toss, and every other natural number to *False*.

lemma pmf-Pi :

assumes $A: \text{finite } A$

shows $\text{pmf } (Pi\text{-}pmf \ A \ dflt \ p) \ f =$

$(\text{if } (\forall x. x \notin A \longrightarrow f \ x = dflt) \text{ then } \prod_{x \in A}. \text{ pmf } (p \ x) (f \ x) \text{ else } 0)$

$\langle \text{proof} \rangle$

lemma $Pi\text{-}pmf\text{-}cong$:

assumes $A = A' \ dflt = dflt' \ \bigwedge x. x \in A \implies f \ x = f' \ x$

shows $Pi\text{-}pmf \ A \ dflt \ f = Pi\text{-}pmf \ A' \ dflt' \ f'$

$\langle \text{proof} \rangle$

lemma $\text{pmf-Pi}'$:

assumes $\text{finite } A \ \bigwedge x. x \notin A \implies f \ x = dflt$

shows $\text{pmf } (Pi\text{-}pmf \ A \ dflt \ p) \ f = (\prod_{x \in A}. \text{ pmf } (p \ x) (f \ x))$

$\langle \text{proof} \rangle$

lemma pmf-Pi-outside :

assumes $\text{finite } A \ \exists x. x \notin A \ \wedge \ f \ x \neq dflt$

shows $\text{pmf } (Pi\text{-}pmf \ A \ dflt \ p) \ f = 0$

$\langle \text{proof} \rangle$

lemma $\text{pmf-Pi-empty [simp]}$: $Pi\text{-}pmf \ \{\} \ dflt \ p = \text{return-pmf } (\lambda-. \ dflt)$

$\langle \text{proof} \rangle$

lemma set-Pi-pmf-subset : $\text{finite } A \implies \text{set-pmf } (Pi\text{-}pmf \ A \ dflt \ p) \subseteq \{f. \forall x. x \notin A \longrightarrow f \ x = dflt\}$

$\langle \text{proof} \rangle$

26.3 Dependent product sets with a default

The following describes a dependent product of sets where the functions are required to return the default value *dflt* outside their domain, in analogy to *PiE*, which uses *undefined*.

definition *PiE-dflt*

where $PiE\text{-}dflt\ A\ dflt\ B = \{f. \forall x. (x \in A \longrightarrow f\ x \in B\ x) \wedge (x \notin A \longrightarrow f\ x = dflt)\}$

lemma *restrict-PiE-dflt*: $(\lambda h. restrict\ h\ A) \text{ ‘ } PiE\text{-}dflt\ A\ dflt\ B = PiE\ A\ B$

<proof>

lemma *dflt-image-PiE*: $(\lambda h\ x. if\ x \in A\ then\ h\ x\ else\ dflt) \text{ ‘ } PiE\ A\ B = PiE\text{-}dflt\ A\ dflt\ B$

(**is** $?f \text{ ‘ } ?X = ?Y$)

<proof>

lemma *finite-PiE-dflt [intro]*:

assumes $finite\ A \wedge x. x \in A \implies finite\ (B\ x)$

shows $finite\ (PiE\text{-}dflt\ A\ d\ B)$

<proof>

lemma *card-PiE-dflt*:

assumes $finite\ A \wedge x. x \in A \implies finite\ (B\ x)$

shows $card\ (PiE\text{-}dflt\ A\ d\ B) = (\prod_{x \in A}. card\ (B\ x))$

<proof>

lemma *PiE-dflt-empty-iff [simp]*: $PiE\text{-}dflt\ A\ dflt\ B = \{\} \longleftrightarrow (\exists x \in A. B\ x = \{\})$

<proof>

lemma *set-Pi-pmf-subset'*:

assumes $finite\ A$

shows $set\text{-}pmf\ (Pi\text{-}pmf\ A\ dflt\ p) \subseteq PiE\text{-}dflt\ A\ dflt\ (set\text{-}pmf \circ p)$

<proof>

lemma *set-Pi-pmf*:

assumes $finite\ A$

shows $set\text{-}pmf\ (Pi\text{-}pmf\ A\ dflt\ p) = PiE\text{-}dflt\ A\ dflt\ (set\text{-}pmf \circ p)$

<proof>

The probability of an independent combination of events is precisely the product of the probabilities of each individual event.

lemma *measure-Pi-pmf-PiE-dflt*:

assumes $[simp]: finite\ A$

shows $measure\text{-}pmf.\text{prob}\ (Pi\text{-}pmf\ A\ dflt\ p)\ (PiE\text{-}dflt\ A\ dflt\ B) =$
 $(\prod_{x \in A}. measure\text{-}pmf.\text{prob}\ (p\ x)\ (B\ x))$

<proof>

lemma *measure-Pi-pmf-Pi*:

```

fixes  $t::nat$ 
assumes  $[simp]: finite\ A$ 
shows  $measure\text{-}pmf.\text{prob}\ (Pi\text{-}pmf\ A\ dflt\ p)\ (Pi\ A\ B) =$ 
 $(\prod_{x \in A}. measure\text{-}pmf.\text{prob}\ (p\ x)\ (B\ x))\ (\text{is}\ ?lhs = ?rhs)$ 
 $\langle proof \rangle$ 

```

26.4 Common PMF operations on products

$Pi\text{-}pmf$ distributes over the ‘bind’ operation in the Girmonad:

```

lemma  $Pi\text{-}pmf\text{-}bind$ :
assumes  $finite\ A$ 
shows  $Pi\text{-}pmf\ A\ d\ (\lambda x. bind\text{-}pmf\ (p\ x)\ (q\ x)) =$ 
 $do\ \{f \leftarrow Pi\text{-}pmf\ A\ d'\ p; Pi\text{-}pmf\ A\ d\ (\lambda x. q\ x\ (f\ x))\}\ (\text{is}\ ?lhs = ?rhs)$ 
 $\langle proof \rangle$ 

```

```

lemma  $Pi\text{-}pmf\text{-}return\text{-}pmf\ [simp]$ :
assumes  $finite\ A$ 
shows  $Pi\text{-}pmf\ A\ dflt\ (\lambda x. return\text{-}pmf\ (f\ x)) = return\text{-}pmf\ (\lambda x. \text{if } x \in A \text{ then } f$ 
 $x \text{ else } dflt)$ 
 $\langle proof \rangle$ 

```

Analogously any componentwise mapping can be pulled outside the product:

```

lemma  $Pi\text{-}pmf\text{-}map$ :
assumes  $[simp]: finite\ A$  and  $f\ dflt = dflt'$ 
shows  $Pi\text{-}pmf\ A\ dflt'\ (\lambda x. map\text{-}pmf\ f\ (g\ x)) = map\text{-}pmf\ (\lambda h. f \circ h)\ (Pi\text{-}pmf\ A$ 
 $dflt\ g)$ 
 $\langle proof \rangle$ 

```

We can exchange the default value in a product of PMFs like this:

```

lemma  $Pi\text{-}pmf\text{-}default\text{-}swap$ :
assumes  $finite\ A$ 
shows  $map\text{-}pmf\ (\lambda f\ x. \text{if } x \in A \text{ then } f\ x \text{ else } dflt')\ (Pi\text{-}pmf\ A\ dflt\ p) =$ 
 $Pi\text{-}pmf\ A\ dflt'\ p\ (\text{is}\ ?lhs = ?rhs)$ 
 $\langle proof \rangle$ 

```

The following rule allows reindexing the product:

```

lemma  $Pi\text{-}pmf\text{-}bij\text{-}betw$ :
assumes  $finite\ A\ bij\text{-}betw\ h\ A\ B\ \wedge x. x \notin A \implies h\ x \notin B$ 
shows  $Pi\text{-}pmf\ A\ dflt\ (\lambda -. f) = map\text{-}pmf\ (\lambda g. g \circ h)\ (Pi\text{-}pmf\ B\ dflt\ (\lambda -. f))$ 
 $(\text{is}\ ?lhs = ?rhs)$ 
 $\langle proof \rangle$ 

```

A product of uniform random choices is again a uniform distribution.

```

lemma  $Pi\text{-}pmf\text{-}of\text{-}set$ :
assumes  $finite\ A\ \wedge x. x \in A \implies finite\ (B\ x)\ \wedge x. x \in A \implies B\ x \neq \{\}$ 
shows  $Pi\text{-}pmf\ A\ d\ (\lambda x. pmf\text{-}of\text{-}set\ (B\ x)) = pmf\text{-}of\text{-}set\ (PiE\text{-}dflt\ A\ d\ B)\ (\text{is}$ 
 $?lhs = ?rhs)$ 
 $\langle proof \rangle$ 

```

26.5 Merging and splitting PMF products

The following lemma shows that we can add a single PMF to a product:

lemma *Pi-pmf-insert:*

assumes *finite A x ∉ A*
shows $Pi\text{-}pmf\ (insert\ x\ A)\ dflt\ p = map\text{-}pmf\ (\lambda(y,f). f(x:=y))\ (pair\text{-}pmf\ (p\ x)\ (Pi\text{-}pmf\ A\ dflt\ p))$
 $\langle proof \rangle$

lemma *Pi-pmf-insert':*

assumes *finite A x ∉ A*
shows $Pi\text{-}pmf\ (insert\ x\ A)\ dflt\ p =$
 $do\ \{y \leftarrow p\ x; f \leftarrow Pi\text{-}pmf\ A\ dflt\ p; return\text{-}pmf\ (f(x := y))\}$
 $\langle proof \rangle$

lemma *Pi-pmf-singleton:*

$Pi\text{-}pmf\ \{x\}\ dflt\ p = map\text{-}pmf\ (\lambda a\ b. if\ b = x\ then\ a\ else\ dflt)\ (p\ x)$
 $\langle proof \rangle$

Projecting a product of PMFs onto a component yields the expected result:

lemma *Pi-pmf-component:*

assumes *finite A*
shows $map\text{-}pmf\ (\lambda f. f\ x)\ (Pi\text{-}pmf\ A\ dflt\ p) = (if\ x \in A\ then\ p\ x\ else\ return\text{-}pmf\ dflt)$
 $\langle proof \rangle$

We can take merge two PMF products on disjoint sets like this:

lemma *Pi-pmf-union:*

assumes *finite A finite B A ∩ B = {}*
shows $Pi\text{-}pmf\ (A \cup B)\ dflt\ p =$
 $map\text{-}pmf\ (\lambda(f,g)\ x. if\ x \in A\ then\ f\ x\ else\ g\ x)$
 $(pair\text{-}pmf\ (Pi\text{-}pmf\ A\ dflt\ p)\ (Pi\text{-}pmf\ B\ dflt\ p))\ (is_ =\ map\text{-}pmf\ (?h\ A)\$
 $(?q\ A))$
 $\langle proof \rangle$

We can also project a product to a subset of the indices by mapping all the other indices to the default value:

lemma *Pi-pmf-subset:*

assumes *finite A A' ⊆ A*
shows $Pi\text{-}pmf\ A'\ dflt\ p = map\text{-}pmf\ (\lambda f\ x. if\ x \in A'\ then\ f\ x\ else\ dflt)\ (Pi\text{-}pmf\ A\ dflt\ p)$
 $\langle proof \rangle$

lemma *Pi-pmf-subset':*

fixes $f :: 'a \Rightarrow 'b\ pmf$
assumes *finite A B ⊆ A ∧ x. x ∈ A − B ⇒ f x = return-pmf dflt*
shows $Pi\text{-}pmf\ A\ dflt\ f = Pi\text{-}pmf\ B\ dflt\ f$
 $\langle proof \rangle$

lemma *Pi-pmf-if-set*:

assumes *finite A*

shows $Pi\text{-}pmf\ A\ dflt\ (\lambda x. \text{if } b\ x \text{ then } f\ x \text{ else return-pmf } dflt) =$
 $Pi\text{-}pmf\ \{x \in A. b\ x\}\ dflt\ f$

<proof>

lemma *Pi-pmf-if-set'*:

assumes *finite A*

shows $Pi\text{-}pmf\ A\ dflt\ (\lambda x. \text{if } b\ x \text{ then return-pmf } dflt \text{ else } f\ x) =$
 $Pi\text{-}pmf\ \{x \in A. \neg b\ x\}\ dflt\ f$

<proof>

Lastly, we can delete a single component from a product:

lemma *Pi-pmf-remove*:

assumes *finite A*

shows $Pi\text{-}pmf\ (A - \{x\})\ dflt\ p = map\text{-}pmf\ (\lambda f. f(x := dflt))\ (Pi\text{-}pmf\ A\ dflt\ p)$

<proof>

26.6 Additional properties

lemma *nn-integral-prod-Pi-pmf*:

assumes *finite A*

shows $nn\text{-}integral\ (Pi\text{-}pmf\ A\ dflt\ p)\ (\lambda y. \prod_{x \in A}. f\ x\ (y\ x)) = (\prod_{x \in A}. nn\text{-}integral\ (p\ x)\ (f\ x))$

<proof>

lemma *integrable-prod-Pi-pmf*:

fixes $f :: 'a \Rightarrow 'b \Rightarrow 'c :: \{\text{real-normed-field, second-countable-topology, banach}\}$

assumes *finite A* **and** $\bigwedge x. x \in A \implies integrable\ (measure\text{-}pmf\ (p\ x))\ (f\ x)$

shows $integrable\ (measure\text{-}pmf\ (Pi\text{-}pmf\ A\ dflt\ p))\ (\lambda h. \prod_{x \in A}. f\ x\ (h\ x))$

<proof>

lemma *expectation-prod-Pi-pmf*:

fixes $f :: - \Rightarrow - \Rightarrow \text{real}$

assumes *finite A*

assumes $\bigwedge x. x \in A \implies integrable\ (measure\text{-}pmf\ (p\ x))\ (f\ x)$

assumes $\bigwedge x\ y. x \in A \implies y \in set\text{-}pmf\ (p\ x) \implies f\ x\ y \geq 0$

shows $measure\text{-}pmf.\text{expectation}\ (Pi\text{-}pmf\ A\ dflt\ p)\ (\lambda y. \prod_{x \in A}. f\ x\ (y\ x)) =$
 $(\prod_{x \in A}. measure\text{-}pmf.\text{expectation}\ (p\ x)\ (\lambda v. f\ x\ v))$

<proof>

lemma *indep-vars-Pi-pmf*:

assumes *fin: finite I*

shows $prob\text{-}space.indep\text{-}vars\ (measure\text{-}pmf\ (Pi\text{-}pmf\ I\ dflt\ p))$
 $(\lambda -. count\text{-}space\ UNIV)\ (\lambda x\ f. f\ x)\ I$

<proof>

lemma

fixes $h :: 'a :: \text{comm-monoid-add} \Rightarrow 'b :: \{\text{banach}, \text{second-countable-topology}\}$
assumes $\text{fin}: \text{finite } I$
assumes $\text{integrable}: \bigwedge i. i \in I \implies \text{integrable } (\text{measure-pmf } (D \ i)) \ h$
shows $\text{integrable-sum-Pi-pmf}: \text{integrable } (\text{Pi-pmf } I \ \text{dflt } D) \ (\lambda g. \sum i \in I. h \ (g \ i))$
and $\text{expectation-sum-Pi-pmf}: \text{measure-pmf.expectation } (\text{Pi-pmf } I \ \text{dflt } D) \ (\lambda g. \sum i \in I. h \ (g \ i)) =$
 $(\sum i \in I. \text{measure-pmf.expectation } (D \ i) \ h)$
 $\langle \text{proof} \rangle$

26.7 Applications

Choosing a subset of a set uniformly at random is equivalent to tossing a fair coin independently for each element and collecting all the elements that came up heads.

lemma $\text{pmf-of-set-Pow-conv-bernoulli}$:

assumes $\text{finite } (A :: 'a \text{ set})$
shows $\text{map-pmf } (\lambda b. \{x \in A. b \ x\}) \ (\text{Pi-pmf } A \ P \ (\lambda -. \text{bernoulli-pmf } (1/2))) =$
 $\text{pmf-of-set } (\text{Pow } A)$
 $\langle \text{proof} \rangle$

A binomial distribution can be seen as the number of successes in n independent coin tosses.

lemma $\text{binomial-pmf-altdef}'$:

fixes $A :: 'a \text{ set}$
assumes $\text{finite } A$ **and** $\text{card } A = n$ **and** $p: p \in \{0..1\}$
shows $\text{binomial-pmf } n \ p =$
 $\text{map-pmf } (\lambda f. \text{card } \{x \in A. f \ x\}) \ (\text{Pi-pmf } A \ \text{dflt } (\lambda -. \text{bernoulli-pmf } p)) \ (\text{is } ?lhs = ?rhs)$
 $\langle \text{proof} \rangle$

end

27 Hoeffding’s Lemma and Hoeffding’s Inequality

theory Hoeffding

imports $\text{Product-PMF Independent-Family}$

begin

Hoeffding’s inequality shows that a sum of bounded independent random variables is concentrated around its mean, with an exponential decay of the tail probabilities.

27.1 Hoeffding’s Lemma

lemma convex-on-exp :

fixes $l :: \text{real}$

assumes $l \geq 0$
shows *convex-on UNIV* $(\lambda x. \exp(l * x))$
 $\langle \text{proof} \rangle$

lemma *mult-const-minus-self-real-le*:

fixes $x :: \text{real}$
shows $x * (c - x) \leq c^2 / 4$
 $\langle \text{proof} \rangle$

lemma *Hoeffdings-lemma-aux*:

fixes $h \ p :: \text{real}$
assumes $h \geq 0$ **and** $p \geq 0$
defines $L \equiv (\lambda h. -h * p + \ln (1 + p * (\exp h - 1)))$
shows $L \ h \leq h^2 / 8$
 $\langle \text{proof} \rangle$

locale *interval-bounded-random-variable* = *prob-space* +

fixes $f :: 'a \Rightarrow \text{real}$ **and** $a \ b :: \text{real}$
assumes *random-variable* [measurable]: *random-variable borel* f
assumes *AE-in-interval*: *AE* x *in* $M. f \ x \in \{a..b\}$
begin

lemma *integrable [intro]: integrable* $M \ f$
 $\langle \text{proof} \rangle$

We first show Hoeffding’s lemma for distributions whose expectation is 0.
The general case will easily follow from this later.

lemma *Hoeffdings-lemma-nn-integral-0*:

assumes $l > 0$ **and** $E0$: *expectation* $f = 0$
shows *nn-integral* $M \ (\lambda x. \exp (l * f \ x)) \leq \text{ennreal} (\exp (l^2 * (b - a)^2 / 8))$
 $\langle \text{proof} \rangle$

context

begin

interpretation *shift: interval-bounded-random-variable* $M \ \lambda x. f \ x - \mu \ a - \mu \ b - \mu$

rewrites $b - \mu - (a - \mu) \equiv b - a$
 $\langle \text{proof} \rangle$

lemma *expectation-shift: expectation* $(\lambda x. f \ x - \text{expectation } f) = 0$
 $\langle \text{proof} \rangle$

lemmas *Hoeffdings-lemma-nn-integral* = *shift.Hoeffdings-lemma-nn-integral-0*[*OF* - *expectation-shift*]

end

end

27.2 Hoeffding’s Inequality

Consider n independent real random variables X_1, \dots, X_n that each almost surely lie in a compact interval $[a_i, b_i]$. Hoeffding’s inequality states that the distribution of the sum of the X_i is tightly concentrated around the sum of the expected values: the probability of it being above or below the sum of the expected values by more than some ε decreases exponentially with ε .

```

locale indep-interval-bounded-random-variables = prob-space +
  fixes  $I :: 'b$  set and  $X :: 'b \Rightarrow 'a \Rightarrow \text{real}$ 
  fixes  $a\ b :: 'b \Rightarrow \text{real}$ 
  assumes  $\text{fin: finite } I$ 
  assumes  $\text{indep: indep-vars } (\lambda\cdot. \text{borel})\ X\ I$ 
  assumes  $\text{AE-in-interval: } \bigwedge i. i \in I \implies \text{AE } x \text{ in } M. X\ i\ x \in \{a\ i..b\ i\}$ 
begin

```

```

lemma random-variable [measurable]:
  assumes  $i: i \in I$ 
  shows  $\text{random-variable borel } (X\ i)$ 
   $\langle \text{proof} \rangle$ 

```

```

lemma bounded-random-variable [intro]:
  assumes  $i: i \in I$ 
  shows  $\text{interval-bounded-random-variable } M\ (X\ i)\ (a\ i)\ (b\ i)$ 
   $\langle \text{proof} \rangle$ 

```

end

```

locale Hoeffding-ineq = indep-interval-bounded-random-variables +
  fixes  $\mu :: \text{real}$ 
  defines  $\mu \equiv (\sum i \in I. \text{expectation } (X\ i))$ 
begin

```

```

theorem Hoeffding-ineq-ge:
  assumes  $\varepsilon \geq 0$ 
  assumes  $(\sum i \in I. (b\ i - a\ i)^2) > 0$ 
  shows  $\text{prob } \{x \in \text{space } M. (\sum i \in I. X\ i\ x) \geq \mu + \varepsilon\} \leq \exp (-2 * \varepsilon^2 / (\sum i \in I. (b\ i - a\ i)^2))$ 
proof (cases  $\varepsilon = 0$ )
  case [simp]:  $\text{True}$ 
  have  $\text{prob } \{x \in \text{space } M. (\sum i \in I. X\ i\ x) \geq \mu + \varepsilon\} \leq 1$ 
  by simp
  thus ?thesis by simp
next
  case  $\text{False}$ 
  with  $\langle \varepsilon \geq 0 \rangle$  have  $\varepsilon: \varepsilon > 0$ 

```



```

by auto

define d where d = (∑ i∈I. (b i - a i)²)
define l :: real where l = 4 * ε / d
have d: d > 0
  using assms by (simp add: d-def)
have l: l > 0
  using ε d by (simp add: l-def)
define μ' where μ' = (λi. expectation (X i))

have {x∈space M. (∑ i∈I. X i x) ≥ μ + ε} = {x∈space M. (∑ i∈I. X i x) -
μ ≥ ε}
  by (simp add: algebra-simps)
hence ennreal (prob {x∈space M. (∑ i∈I. X i x) ≥ μ + ε}) = emeasure M ...
  by (simp add: emeasure-eq-measure)
also have ... ≤ ennreal (exp (-l*ε)) * (∫⁺ x∈space M. exp (l * ((∑ i∈I. X i
x) - μ)) ∂M)
  by (intro Chernoff-ineq-nn-integral-ge l) auto
also have (λx. (∑ i∈I. X i x) - μ) = (λx. (∑ i∈I. X i x - μ' i))
  by (simp add: μ-def sum-subtractf μ'-def)
also have (∫⁺ x∈space M. exp (l * ((∑ i∈I. X i x - μ' i))) ∂M) =
  (∫⁺ x. (∏ i∈I. ennreal (exp (l * (X i x - μ' i)))) ∂M)
  by (intro nn-integral-cong)
  (simp-all add: sum-distrib-left ring-distrib exp-diff exp-sum fin prod-ennreal)
also have ... = (∏ i∈I. ∫⁺ x. ennreal (exp (l * (X i x - μ' i))) ∂M)
  by (intro indep-vars-nn-integral fin indep-vars-compose2[OF indep]) auto
also have ennreal (exp (-l * ε)) * ... ≤
  ennreal (exp (-l * ε)) * (∏ i∈I. ennreal (exp (l² * (b i - a i)² / 8)))
proof (intro mult-left-mono prod-mono-ennreal)
  fix i assume i: i ∈ I
  from i interpret interval-bounded-random-variable M X i a i b i ..
  show (∫⁺ x. ennreal (exp (l * (X i x - μ' i))) ∂M) ≤ ennreal (exp (l² * (b i
- a i)² / 8))
    unfolding μ'-def by (rule Hoeffdings-lemma-nn-integral) fact+
qed auto
also have ... = ennreal (exp (-l*ε)) * (∏ i∈I. exp (l² * (b i - a i)² / 8)))
  by (simp add: prod-ennreal prod-nonneg flip: ennreal-mult)
also have exp (-l*ε) * (∏ i∈I. exp (l² * (b i - a i)² / 8)) = exp (d * l² / 8 -
l * ε)
  by (simp add: exp-diff exp-minus sum-divide-distrib sum-distrib-left
sum-distrib-right exp-sum fin divide-simps mult-ac d-def)
also have d * l² / 8 - l * ε = -2 * ε² / d
  using d by (simp add: l-def field-simps power2-eq-square)
finally show ?thesis
  by (subst (asm) ennreal-le-iff) (simp-all add: d-def)
qed

corollary Hoeffding-ineq-le:
  assumes ε: ε ≥ 0

```

assumes $(\sum_{i \in I}. (b \ i - a \ i)^2) > 0$
shows $\text{prob } \{x \in \text{space } M. (\sum_{i \in I}. X \ i \ x) \leq \mu - \varepsilon\} \leq \exp (-2 * \varepsilon^2 / (\sum_{i \in I}. (b \ i - a \ i)^2))$
 $\langle \text{proof} \rangle$

corollary *Hoeffding-ineq-abs-ge*:

assumes $\varepsilon: \varepsilon \geq 0$
assumes $(\sum_{i \in I}. (b \ i - a \ i)^2) > 0$
shows $\text{prob } \{x \in \text{space } M. |(\sum_{i \in I}. X \ i \ x) - \mu| \geq \varepsilon\} \leq 2 * \exp (-2 * \varepsilon^2 / (\sum_{i \in I}. (b \ i - a \ i)^2))$
 $\langle \text{proof} \rangle$

end

27.3 Hoeffding’s inequality for i.i.d. bounded random variables

If we have n even identically-distributed random variables, the statement of Hoeffding’s lemma simplifies a bit more: it shows that the probability that the average of the X_i is more than ε above the expected value is no greater than $e^{\frac{-2n\varepsilon^2}{(b-a)^2}}$.

This essentially gives us a more concrete version of the weak law of large numbers: the law states that the probability vanishes for $n \rightarrow \infty$ for any $\varepsilon > 0$. Unlike Hoeffding’s inequality, it does not assume the variables to have bounded support, but it does not provide concrete bounds.

locale *iid-interval-bounded-random-variables* = *prob-space* +
fixes $I :: 'b \text{ set}$ **and** $X :: 'b \Rightarrow 'a \Rightarrow \text{real}$ **and** $Y :: 'a \Rightarrow \text{real}$
fixes $a \ b :: \text{real}$
assumes *fin*: *finite* I
assumes *indep*: *indep-vars* $(\lambda-. \text{borel}) \ X \ I$
assumes *distr-X*: $i \in I \implies \text{distr } M \ \text{borel} \ (X \ i) = \text{distr } M \ \text{borel} \ Y$
assumes *rv-Y* [*measurable*]: *random-variable* $\text{borel } Y$
assumes *AE-in-interval*: $\text{AE } x \text{ in } M. Y \ x \in \{a..b\}$
begin

lemma *random-variable* [*measurable*]:
assumes $i: i \in I$
shows *random-variable* $\text{borel} \ (X \ i)$
 $\langle \text{proof} \rangle$

sublocale $X: \text{indep-interval-bounded-random-variables } M \ I \ X \ \lambda-. \ a \ \lambda-. \ b$
 $\langle \text{proof} \rangle$

lemma *expectation-X* [*simp*]:
assumes $i: i \in I$
shows *expectation* $(X \ i) = \text{expectation } Y$
 $\langle \text{proof} \rangle$

end

locale *Hoeffding-ineq-iid* = *iid-interval-bounded-random-variables* +
fixes $\mu :: \text{real}$
defines $\mu \equiv \text{expectation } Y$
begin

sublocale *X*: *Hoeffding-ineq* $M \ I \ X \ \lambda \cdot. \ a \ \lambda \cdot. \ b \ \text{real} \ (\text{card } I) * \mu$
 $\langle \text{proof} \rangle$

corollary

assumes $\varepsilon: \varepsilon \geq 0$
assumes $a < b \ I \neq \{\}$
defines $n \equiv \text{card } I$
shows *Hoeffding-ineq-ge*:
 $\text{prob } \{x \in \text{space } M. (\sum i \in I. X \ i \ x) \geq n * \mu + \varepsilon\} \leq$
 $\exp (-2 * \varepsilon^2 / (n * (b - a)^2)) \ (\text{is } ?le)$
and *Hoeffding-ineq-le*:
 $\text{prob } \{x \in \text{space } M. (\sum i \in I. X \ i \ x) \leq n * \mu - \varepsilon\} \leq$
 $\exp (-2 * \varepsilon^2 / (n * (b - a)^2)) \ (\text{is } ?ge)$
and *Hoeffding-ineq-abs-ge*:
 $\text{prob } \{x \in \text{space } M. |(\sum i \in I. X \ i \ x) - n * \mu| \geq \varepsilon\} \leq$
 $2 * \exp (-2 * \varepsilon^2 / (n * (b - a)^2)) \ (\text{is } ?abs-ge)$
 $\langle \text{proof} \rangle$

lemma

assumes $\varepsilon: \varepsilon \geq 0$
assumes $a < b \ I \neq \{\}$
defines $n \equiv \text{card } I$
shows *Hoeffding-ineq-ge'*:
 $\text{prob } \{x \in \text{space } M. (\sum i \in I. X \ i \ x) / n \geq \mu + \varepsilon\} \leq$
 $\exp (-2 * n * \varepsilon^2 / (b - a)^2) \ (\text{is } ?ge)$
and *Hoeffding-ineq-le'*:
 $\text{prob } \{x \in \text{space } M. (\sum i \in I. X \ i \ x) / n \leq \mu - \varepsilon\} \leq$
 $\exp (-2 * n * \varepsilon^2 / (b - a)^2) \ (\text{is } ?le)$
and *Hoeffding-ineq-abs-ge'*:
 $\text{prob } \{x \in \text{space } M. |(\sum i \in I. X \ i \ x) / n - \mu| \geq \varepsilon\} \leq$
 $2 * \exp (-2 * n * \varepsilon^2 / (b - a)^2) \ (\text{is } ?abs-ge)$
 $\langle \text{proof} \rangle$

end

27.4 Hoeffding’s Inequality for the Binomial distribution

We can now apply Hoeffding’s inequality to the Binomial distribution, which can be seen as the sum of n i.i.d. coin flips (the support of each of which is contained in $[0, 1]$).

locale *binomial-distribution* =
 fixes $n :: \text{nat}$ and $p :: \text{real}$
 assumes $p: p \in \{0..1\}$
begin

context
 fixes $\text{coins} :: (\text{nat} \Rightarrow \text{bool}) \text{ pmf}$ and μ
 assumes $n: n > 0$
 defines $\text{coins} \equiv \text{Pi-pmf } \{..<n\} \text{ False } (\lambda-. \text{bernoulli-pmf } p)$
begin

lemma *coins-component*:
 assumes $i: i < n$
 shows $\text{distr } (\text{measure-pmf coins}) \text{ borel } (\lambda f. \text{if } f \text{ } i \text{ then } 1 \text{ else } 0) =$
 $\text{distr } (\text{measure-pmf } (\text{bernoulli-pmf } p)) \text{ borel } (\lambda b. \text{if } b \text{ then } 1 \text{ else } 0)$
 $\langle \text{proof} \rangle$

lemma *prob-binomial-pmf-conv-coins*:
 $\text{measure-pmf.prob } (\text{binomial-pmf } n \text{ } p) \{x. P \text{ (real } x)\} =$
 $\text{measure-pmf.prob coins } \{x. P (\sum i < n. \text{if } x \text{ } i \text{ then } 1 \text{ else } 0)\}$
 $\langle \text{proof} \rangle$

interpretation *Hoeffding-ineq-iid*
 $\text{coins } \{..<n\} \lambda i f. \text{if } f \text{ } i \text{ then } 1 \text{ else } 0 \lambda f. \text{if } f \text{ } 0 \text{ then } 1 \text{ else } 0 \text{ } 0 \text{ } 1 \text{ } p$
 $\langle \text{proof} \rangle$

corollary
 fixes $\varepsilon :: \text{real}$
 assumes $\varepsilon: \varepsilon \geq 0$
 shows $\text{prob-ge: measure-pmf.prob } (\text{binomial-pmf } n \text{ } p) \{x. x \geq n * p + \varepsilon\} \leq \exp$
 $(-2 * \varepsilon^2 / n)$
 and $\text{prob-le: measure-pmf.prob } (\text{binomial-pmf } n \text{ } p) \{x. x \leq n * p - \varepsilon\} \leq \exp$
 $(-2 * \varepsilon^2 / n)$
 and prob-abs-ge:
 $\text{measure-pmf.prob } (\text{binomial-pmf } n \text{ } p) \{x. |x - n * p| \geq \varepsilon\} \leq 2 * \exp (-2$
 $* \varepsilon^2 / n)$
 $\langle \text{proof} \rangle$

corollary
 fixes $\varepsilon :: \text{real}$
 assumes $\varepsilon: \varepsilon \geq 0$
 shows $\text{prob-ge': measure-pmf.prob } (\text{binomial-pmf } n \text{ } p) \{x. x / n \geq p + \varepsilon\} \leq \exp$
 $(-2 * n * \varepsilon^2)$
 and $\text{prob-le': measure-pmf.prob } (\text{binomial-pmf } n \text{ } p) \{x. x / n \leq p - \varepsilon\} \leq \exp$
 $(-2 * n * \varepsilon^2)$
 and prob-abs-ge':
 $\text{measure-pmf.prob } (\text{binomial-pmf } n \text{ } p) \{x. |x / n - p| \geq \varepsilon\} \leq 2 * \exp (-2$
 $* n * \varepsilon^2)$
 $\langle \text{proof} \rangle$

end

end

27.5 Tail bounds for the negative binomial distribution

Since the tail probabilities of a negative Binomial distribution are equal to the tail probabilities of some Binomial distribution, we can obtain tail bounds for the negative Binomial distribution through the Hoeffding tail bounds for the Binomial distribution.

context

fixes $p\ q :: \text{real}$

assumes $p: p \in \{0 < \dots < 1\}$

defines $q \equiv 1 - p$

begin

lemma *prob-neg-binomial-pmf-ge-bound:*

fixes $n :: \text{nat}$ **and** $k :: \text{real}$

defines $\mu \equiv \text{real } n * q / p$

assumes $k: k \geq 0$

shows $\text{measure-pmf.prob } (\text{neg-binomial-pmf } n\ p) \{x. \text{real } x \geq \mu + k\}$
 $\leq \exp (-2 * p \wedge 3 * k^2 / (n + p * k))$

<proof>

lemma *prob-neg-binomial-pmf-le-bound:*

fixes $n :: \text{nat}$ **and** $k :: \text{real}$

defines $\mu \equiv \text{real } n * q / p$

assumes $k: k \geq 0$

shows $\text{measure-pmf.prob } (\text{neg-binomial-pmf } n\ p) \{x. \text{real } x \leq \mu - k\}$
 $\leq \exp (-2 * p \wedge 3 * k^2 / (n - p * k))$

<proof>

Due to the function $\exp(-l/x)$ being concave for $x \geq \frac{l}{2}$, the above two bounds can be combined into the following one for moderate values of k . (cf. <https://math.stackexchange.com/questions/1565559>)

lemma *prob-neg-binomial-pmf-abs-ge-bound:*

fixes $n :: \text{nat}$ **and** $k :: \text{real}$

defines $\mu \equiv \text{real } n * q / p$

assumes $k \geq 0$ **and** $n\text{-ge}: n \geq p * k * (p^2 * k + 1)$

shows $\text{measure-pmf.prob } (\text{neg-binomial-pmf } n\ p) \{x. |\text{real } x - \mu| \geq k\} \leq$
 $2 * \exp (-2 * p \wedge 3 * k \wedge 2 / n)$

<proof>

end

end

theory *Stream-Space*

imports

Infinite-Product-Measure

HOL-Library.Stream

HOL-Library.Linear-Temporal-Logic-on-Streams

begin

lemma *stream-eq-Stream-iff*: $s = x \#\# t \longleftrightarrow (\text{shd } s = x \wedge \text{stl } s = t)$
 $\langle \text{proof} \rangle$

lemma *Stream-snth*: $(x \#\# s) !! n = (\text{case } n \text{ of } 0 \Rightarrow x \mid \text{Suc } n \Rightarrow s !! n)$
 $\langle \text{proof} \rangle$

definition *to-stream* :: $(\text{nat} \Rightarrow 'a) \Rightarrow 'a \text{ stream}$ **where**
to-stream $X = \text{smap } X \text{ nats}$

lemma *to-stream-nat-case*: $\text{to-stream } (\text{case-nat } x \text{ } X) = x \#\# \text{to-stream } X$
 $\langle \text{proof} \rangle$

lemma *to-stream-in-streams*: $\text{to-stream } X \in \text{streams } S \longleftrightarrow (\forall n. X \ n \in S)$
 $\langle \text{proof} \rangle$

definition *stream-space* :: $'a \text{ measure} \Rightarrow 'a \text{ stream measure}$ **where**
stream-space $M =$
 $\text{distr } (\Pi_M i \in \text{UNIV}. M) (\text{vimage-algebra } (\text{streams } (\text{space } M)) \text{ snth } (\Pi_M i \in \text{UNIV}. M)) \text{ to-stream}$

lemma *space-stream-space*: $\text{space } (\text{stream-space } M) = \text{streams } (\text{space } M)$
 $\langle \text{proof} \rangle$

lemma *streams-stream-space[intro]*: $\text{streams } (\text{space } M) \in \text{sets } (\text{stream-space } M)$
 $\langle \text{proof} \rangle$

lemma *stream-space-Stream*:
 $x \#\# \omega \in \text{space } (\text{stream-space } M) \longleftrightarrow x \in \text{space } M \wedge \omega \in \text{space } (\text{stream-space } M)$
 $\langle \text{proof} \rangle$

lemma *stream-space-eq-distr*: $\text{stream-space } M = \text{distr } (\Pi_M i \in \text{UNIV}. M) (\text{stream-space } M) \text{ to-stream}$
 $\langle \text{proof} \rangle$

lemma *sets-stream-space-cong[measurable-cong]*:
 $\text{sets } M = \text{sets } N \implies \text{sets } (\text{stream-space } M) = \text{sets } (\text{stream-space } N)$
 $\langle \text{proof} \rangle$

lemma *measurable-snth-PiM*: $(\lambda \omega n. \omega !! n) \in \text{measurable } (\text{stream-space } M) (\Pi_M i \in \text{UNIV}. M)$

<proof>

lemma *measurable-snth*[*measurable*]: $(\lambda\omega. \omega !! n) \in \text{measurable } (\text{stream-space } M)$
M
<proof>

lemma *measurable-shd*[*measurable*]: *shd* $\in \text{measurable } (\text{stream-space } M)$ *M*
<proof>

lemma *measurable-stream-space2*:
assumes *f-snth*: $\bigwedge n. (\lambda x. f\ x !! n) \in \text{measurable } N\ M$
shows *f* $\in \text{measurable } N\ (\text{stream-space } M)$
<proof>

lemma *measurable-stream-coinduct*[*consumes 1, case-names shd stl, coinduct set: measurable*]:
assumes *F f*
assumes *h*: $\bigwedge f. F\ f \implies (\lambda x. \text{shd } (f\ x)) \in \text{measurable } N\ M$
assumes *t*: $\bigwedge f. F\ f \implies F\ (\lambda x. \text{stl } (f\ x))$
shows *f* $\in \text{measurable } N\ (\text{stream-space } M)$
<proof>

lemma *measurable-sdrop*[*measurable*]: *sdrop n* $\in \text{measurable } (\text{stream-space } M)$ (*stream-space M*)
<proof>

lemma *measurable-stl*[*measurable*]: $(\lambda\omega. \text{stl } \omega) \in \text{measurable } (\text{stream-space } M)$
(*stream-space M*)
<proof>

lemma *measurable-to-stream*[*measurable*]: *to-stream* $\in \text{measurable } (\Pi_M\ i \in \text{UNIV}. M)$ (*stream-space M*)
<proof>

lemma *measurable-Stream*[*measurable (raw)*]:
assumes *f*[*measurable*]: *f* $\in \text{measurable } N\ M$
assumes *g*[*measurable*]: *g* $\in \text{measurable } N\ (\text{stream-space } M)$
shows $(\lambda x. f\ x \#\#\ g\ x) \in \text{measurable } N\ (\text{stream-space } M)$
<proof>

lemma *measurable-smap*[*measurable*]:
assumes *X*[*measurable*]: *X* $\in \text{measurable } N\ M$
shows *smap X* $\in \text{measurable } (\text{stream-space } N)$ (*stream-space M*)
<proof>

lemma *measurable-stake*[*measurable*]:
stake i $\in \text{measurable } (\text{stream-space } (\text{count-space UNIV}))$ (*count-space (UNIV :: 'a::countable list set)*)
<proof>

lemma *measurable-shift*[*measurable*]:

assumes $f: f \in \text{measurable } N \text{ (stream-space } M)$

assumes [*measurable*]: $g \in \text{measurable } N \text{ (stream-space } M)$

shows $(\lambda x. \text{stake } n \text{ (} f \text{ } x) @- g \text{ } x) \in \text{measurable } N \text{ (stream-space } M)$

<proof>

lemma *measurable-case-stream-replace*[*measurable* (*raw*)]:

$(\lambda x. f \text{ } x \text{ (shd (} g \text{ } x)) (stl (} g \text{ } x))) \in \text{measurable } M \text{ } N \implies (\lambda x. \text{case-stream (} f \text{ } x) (g \text{ } x)) \in \text{measurable } M \text{ } N$

<proof>

lemma *measurable-ev-at*[*measurable*]:

assumes [*measurable*]: $\text{Measurable.pred (stream-space } M) \text{ } P$

shows $\text{Measurable.pred (stream-space } M) \text{ (ev-at } P \text{ } n)$

<proof>

lemma *measurable-alw*[*measurable*]:

$\text{Measurable.pred (stream-space } M) \text{ } P \implies \text{Measurable.pred (stream-space } M) \text{ (alw } P)$

<proof>

lemma *measurable-ev*[*measurable*]:

$\text{Measurable.pred (stream-space } M) \text{ } P \implies \text{Measurable.pred (stream-space } M) \text{ (ev } P)$

<proof>

lemma *measurable-until*:

assumes [*measurable*]: $\text{Measurable.pred (stream-space } M) \text{ } \varphi \text{ Measurable.pred (stream-space } M) \text{ } \psi$

shows $\text{Measurable.pred (stream-space } M) \text{ (} \varphi \text{ until } \psi)$

<proof>

lemma *measurable-holds* [*measurable*]: $\text{Measurable.pred } M \text{ } P \implies \text{Measurable.pred (stream-space } M) \text{ (holds } P)$

<proof>

lemma *measurable-hld*[*measurable*]: **assumes** [*measurable*]: $t \in \text{sets } M$ **shows** $\text{Measurable.pred (stream-space } M) \text{ (HLD } t)$

<proof>

lemma *measurable-nxt*[*measurable* (*raw*)]:

$\text{Measurable.pred (stream-space } M) \text{ } P \implies \text{Measurable.pred (stream-space } M) \text{ (nxt } P)$

<proof>

lemma *measurable-suntil*[*measurable*]:

assumes [*measurable*]: $\text{Measurable.pred (stream-space } M) \text{ } Q \text{ Measurable.pred (stream-space } M) \text{ } P$

shows *Measurable.pred* (*stream-space* *M*) (*Q* *suntil* *P*)
 ⟨*proof*⟩

lemma *measurable-szip*:

($\lambda(\omega 1, \omega 2). \text{ szip } \omega 1 \ \omega 2$) \in *measurable* (*stream-space* $M \otimes_M \text{ stream-space } N$)
 (*stream-space* ($M \otimes_M N$))
 ⟨*proof*⟩

lemma (*in prob-space*) *prob-space-stream-space*: *prob-space* (*stream-space* *M*)
 ⟨*proof*⟩

lemma (*in prob-space*) *nn-integral-stream-space*:

assumes [*measurable*]: $f \in \text{borel-measurable } (\text{stream-space } M)$
shows $(\int^+ X. f \ X \ \partial \text{stream-space } M) = (\int^+ x. (\int^+ X. f \ (x \ \#\# \ X) \ \partial \text{stream-space } M) \ \partial M)$
 ⟨*proof*⟩

lemma (*in prob-space*) *emeasure-stream-space*:

assumes $X[\text{measurable}]$: $X \in \text{sets } (\text{stream-space } M)$
shows $\text{emeasure } (\text{stream-space } M) \ X = (\int^+ t. \text{emeasure } (\text{stream-space } M) \ \{x \in \text{space } (\text{stream-space } M). \ t \ \#\# \ x \in X\} \ \partial M)$
 ⟨*proof*⟩

lemma (*in prob-space*) *prob-stream-space*:

assumes $P[\text{measurable}]$: $\{x \in \text{space } (\text{stream-space } M). \ P \ x\} \in \text{sets } (\text{stream-space } M)$
shows $\mathcal{P}(x \text{ in } \text{stream-space } M. \ P \ x) = (\int^+ t. \mathcal{P}(x \text{ in } \text{stream-space } M. \ P \ (t \ \#\# \ x)) \ \partial M)$
 ⟨*proof*⟩

lemma (*in prob-space*) *AE-stream-space*:

assumes [*measurable*]: *Measurable.pred* (*stream-space* *M*) *P*
shows $(\text{AE } X \text{ in } \text{stream-space } M. \ P \ X) = (\text{AE } x \text{ in } M. \ \text{AE } X \text{ in } \text{stream-space } M. \ P \ (x \ \#\# \ X))$
 ⟨*proof*⟩

lemma (*in prob-space*) *AE-stream-all*:

assumes [*measurable*]: *Measurable.pred* *M* *P* **and** *P*: *AE* *x* *in* *M*. *P* *x*
shows *AE* *x* *in* *stream-space* *M*. *stream-all* *P* *x*
 ⟨*proof*⟩

lemma *streams-sets*:

assumes $X[\text{measurable}]$: $X \in \text{sets } M$ **shows** *streams* $X \in \text{sets } (\text{stream-space } M)$
 ⟨*proof*⟩

lemma *sets-stream-space-in-sets*:

assumes *space*: *space* *N* = *streams* (*space* *M*)
assumes *sets*: $\bigwedge i. (\lambda x. x \ \#\# \ i) \in \text{measurable } N \ M$
shows *sets* (*stream-space* *M*) $\subseteq \text{sets } N$

$\langle \text{proof} \rangle$

lemma *sets-stream-space-eq*: $\text{sets} (\text{stream-space } M) =$
 $\text{sets} (\text{SUP } i \in \text{UNIV}. \text{vimage-algebra} (\text{streams} (\text{space } M)) (\lambda s. s !! i) M)$
 $\langle \text{proof} \rangle$

lemma *sets-restrict-stream-space*:
assumes $S[\text{measurable}]$: $S \in \text{sets } M$
shows $\text{sets} (\text{restrict-space} (\text{stream-space } M) (\text{streams } S)) = \text{sets} (\text{stream-space} (\text{restrict-space } M S))$
 $\langle \text{proof} \rangle$

primrec *sstart* :: $'a \text{ set} \Rightarrow 'a \text{ list} \Rightarrow 'a \text{ stream set}$ **where**
 $\text{sstart } S [] = \text{streams } S$
 $| [\text{simp del}]: \text{sstart } S (x \# xs) = (\#\#) x ` \text{sstart } S xs$

lemma *in-sstart[simp]*: $s \in \text{sstart } S (x \# xs) \longleftrightarrow \text{shd } s = x \wedge \text{stl } s \in \text{sstart } S xs$
 $\langle \text{proof} \rangle$

lemma *sstart-in-streams*: $xs \in \text{lists } S \Longrightarrow \text{sstart } S xs \subseteq \text{streams } S$
 $\langle \text{proof} \rangle$

lemma *sstart-eq*: $x \in \text{streams } S \Longrightarrow x \in \text{sstart } S xs = (\forall i < \text{length } xs. x !! i = xs !! i)$
 $\langle \text{proof} \rangle$

lemma *sstart-sets*: $\text{sstart } S xs \in \text{sets} (\text{stream-space} (\text{count-space } \text{UNIV}))$
 $\langle \text{proof} \rangle$

lemma *sigma-sets-singletons*:
assumes $\text{countable } S$
shows $\text{sigma-sets } S ((\lambda s. \{s\})'S) = \text{Pow } S$
 $\langle \text{proof} \rangle$

lemma *sets-count-space-eq-sigma*:
 $\text{countable } S \Longrightarrow \text{sets} (\text{count-space } S) = \text{sets} (\text{sigma } S ((\lambda s. \{s\})'S))$
 $\langle \text{proof} \rangle$

lemma *sets-stream-space-sstart*:
assumes $S[\text{simp}]$: $\text{countable } S$
shows $\text{sets} (\text{stream-space} (\text{count-space } S)) = \text{sets} (\text{sigma} (\text{streams } S) (\text{sstart } S' \text{lists } S \cup \{\{\}\}))$
 $\langle \text{proof} \rangle$

lemma *Int-stable-sstart*: $\text{Int-stable} (\text{sstart } S' \text{lists } S \cup \{\{\}\})$
 $\langle \text{proof} \rangle$

lemma *stream-space-eq-sstart*:
assumes $S[\text{simp}]$: $\text{countable } S$

assumes P : *prob-space* M *prob-space* N
assumes ae : $AE\ x\ in\ M. x \in streams\ S\ AE\ x\ in\ N. x \in streams\ S$
assumes $sets-M$: $sets\ M = sets\ (stream-space\ (count-space\ UNIV))$
assumes $sets-N$: $sets\ N = sets\ (stream-space\ (count-space\ UNIV))$
assumes $*$: $\bigwedge xs. xs \neq [] \implies xs \in lists\ S \implies emeasure\ M\ (sstart\ S\ xs) =$
 $emeasure\ N\ (sstart\ S\ xs)$
shows $M = N$
 $\langle proof \rangle$

lemma $sets-sstart[measurable]$: $sstart\ \Omega\ xs \in sets\ (stream-space\ (count-space\ UNIV))$
 $\langle proof \rangle$

primrec $scylinder :: 'a\ set \Rightarrow 'a\ set\ list \Rightarrow 'a\ stream\ set$
where
 $scylinder\ S\ [] = streams\ S$
 $| scylinder\ S\ (A\ \#\ As) = \{\omega \in streams\ S. shd\ \omega \in A \wedge stl\ \omega \in scylinder\ S\ As\}$

lemma $scylinder-streams$: $scylinder\ S\ xs \subseteq streams\ S$
 $\langle proof \rangle$

lemma $sets-scylinder$: $(\forall x \in set\ xs. x \in sets\ S) \implies scylinder\ (space\ S)\ xs \in sets\ (stream-space\ S)$
 $\langle proof \rangle$

lemma $stream-space-eq-scylinder$:
assumes P : *prob-space* M *prob-space* N
assumes $Int-stable\ G$ **and** S : $sets\ S = sets\ (sigma\ (space\ S)\ G)$
and C : $countable\ C\ C \subseteq G \cup C = space\ S$ **and** G : $G \subseteq Pow\ (space\ S)$
assumes $sets-M$: $sets\ M = sets\ (stream-space\ S)$
assumes $sets-N$: $sets\ N = sets\ (stream-space\ S)$
assumes $*$: $\bigwedge xs. xs \neq [] \implies xs \in lists\ G \implies emeasure\ M\ (scylinder\ (space\ S)\ xs) =$
 $emeasure\ N\ (scylinder\ (space\ S)\ xs)$
shows $M = N$
 $\langle proof \rangle$

lemma $stream-space-coinduct$:
fixes $R :: 'a\ stream\ measure \Rightarrow 'a\ stream\ measure \Rightarrow bool$
assumes $R\ A\ B$
assumes R : $\bigwedge A\ B. R\ A\ B \implies \exists K \in space\ (prob-algebra\ M). \exists A' \in M \rightarrow_M prob-algebra\ (stream-space\ M). \exists B' \in M \rightarrow_M prob-algebra\ (stream-space\ M).$
 $(AE\ y\ in\ K. R\ (A'\ y)\ (B'\ y) \vee A'\ y = B'\ y) \wedge$
 $A = do\ \{ y \leftarrow K; \omega \leftarrow A'\ y; return\ (stream-space\ M)\ (y\ \#\#\ \omega) \} \wedge$
 $B = do\ \{ y \leftarrow K; \omega \leftarrow B'\ y; return\ (stream-space\ M)\ (y\ \#\#\ \omega) \}$
shows $A = B$
 $\langle proof \rangle$

end

theory *Tree-Space*

imports *HOL–Analysis.Analysis HOL–Library.Tree*
begin

lemma *countable-lfp*:

assumes *step*: $\bigwedge Y. \text{countable } Y \implies \text{countable } (F \ Y)$
and *cont*: *Order-Continuity.sup-continuous* *F*
shows *countable* (*lfp F*)
 $\langle \text{proof} \rangle$

lemma *countable-lfp-apply*:

assumes *step*: $\bigwedge Y \ x. (\bigwedge x. \text{countable } (Y \ x)) \implies \text{countable } (F \ Y \ x)$
and *cont*: *Order-Continuity.sup-continuous* *F*
shows *countable* (*lfp F x*)
 $\langle \text{proof} \rangle$

inductive-set *trees* :: 'a set \Rightarrow 'a tree set **for** *S* :: 'a set **where**

$[intro!]: \text{Leaf} \in \text{trees } S$
 $| \ l \in \text{trees } S \implies r \in \text{trees } S \implies v \in S \implies \text{Node } l \ v \ r \in \text{trees } S$

lemma *Node-in-trees-iff[simp]*: $\text{Node } l \ v \ r \in \text{trees } S \longleftrightarrow (l \in \text{trees } S \wedge v \in S \wedge r \in \text{trees } S)$
 $\langle \text{proof} \rangle$

lemma *trees-sub-lfp*: $\text{trees } S \subseteq \text{lfp } (\lambda T. T \cup \{\text{Leaf}\} \cup (\bigcup l \in T. (\bigcup v \in S. (\bigcup r \in T. \{\text{Node } l \ v \ r\}))))$
 $\langle \text{proof} \rangle$

lemma *countable-trees*: *countable* *A* \implies *countable* (*trees A*)
 $\langle \text{proof} \rangle$

lemma *trees-UNIV[simp]*: *trees UNIV* = *UNIV*
 $\langle \text{proof} \rangle$

instance *tree* :: (*countable*) *countable*
 $\langle \text{proof} \rangle$

lemma *map-in-trees[intro]*: $(\bigwedge x. x \in \text{set-tree } t \implies f \ x \in S) \implies \text{map-tree } f \ t \in \text{trees } S$
 $\langle \text{proof} \rangle$

primrec *trees-cyl* :: 'a set tree \Rightarrow 'a tree set **where**

trees-cyl Leaf = $\{\text{Leaf}\}$
 $| \ \text{trees-cyl } (\text{Node } l \ v \ r) = (\bigcup l' \in \text{trees-cyl } l. (\bigcup v' \in v. (\bigcup r' \in \text{trees-cyl } r. \{\text{Node } l' \ v' \ r'\})))$

definition *tree-sigma* :: 'a measure \Rightarrow 'a tree measure
where

$$\text{tree-sigma } M = \text{sigma } (\text{trees } (\text{space } M)) (\text{trees-cyl } ' \text{ trees } (\text{sets } M))$$

lemma *Node-in-trees-cyl*: $\text{Node } l' \ v' \ r' \in \text{trees-cyl } t \longleftrightarrow$
 $(\exists l \ v \ r. t = \text{Node } l \ v \ r \wedge l' \in \text{trees-cyl } l \wedge r' \in \text{trees-cyl } r \wedge v' \in v)$
 $\langle \text{proof} \rangle$

lemma *trees-cyl-sub-trees*:
assumes $t \in \text{trees } A \ A \subseteq \text{Pow } B$ **shows** $\text{trees-cyl } t \subseteq \text{trees } B$
 $\langle \text{proof} \rangle$

lemma *trees-cyl-sets-in-space*: $\text{trees-cyl } ' \text{ trees } (\text{sets } M) \subseteq \text{Pow } (\text{trees } (\text{space } M))$
 $\langle \text{proof} \rangle$

lemma *space-tree-sigma*: $\text{space } (\text{tree-sigma } M) = \text{trees } (\text{space } M)$
 $\langle \text{proof} \rangle$

lemma *sets-tree-sigma-eq*: $\text{sets } (\text{tree-sigma } M) = \text{sigma-sets } (\text{trees } (\text{space } M))$
 $(\text{trees-cyl } ' \text{ trees } (\text{sets } M))$
 $\langle \text{proof} \rangle$

lemma *Leaf-in-space-tree-sigma* [*measurable, simp, intro*]: $\text{Leaf} \in \text{space } (\text{tree-sigma } M)$
 $\langle \text{proof} \rangle$

lemma *Leaf-in-tree-sigma* [*measurable, simp, intro*]: $\{\text{Leaf}\} \in \text{sets } (\text{tree-sigma } M)$
 $\langle \text{proof} \rangle$

lemma *trees-cyl-map-treeI*: $t \in \text{trees-cyl } (\text{map-tree } (\lambda x. A) \ t) \text{ if } *: t \in \text{trees } A$
 $\langle \text{proof} \rangle$

lemma *trees-cyl-map-in-sets*:
 $(\bigwedge x. x \in \text{set-tree } t \implies f \ x \in \text{sets } M) \implies \text{trees-cyl } (\text{map-tree } f \ t) \in \text{sets } (\text{tree-sigma } M)$
 $\langle \text{proof} \rangle$

lemma *Node-in-tree-sigma*:
assumes $L: X \in \text{sets } (M \otimes_M (\text{tree-sigma } M \otimes_M \text{tree-sigma } M))$
shows $\{\text{Node } l \ v \ r \mid l \ v \ r. (v, l, r) \in X\} \in \text{sets } (\text{tree-sigma } M)$
 $\langle \text{proof} \rangle$

lemma *measurable-left*[*measurable*]: $\text{left} \in \text{tree-sigma } M \rightarrow_M \text{tree-sigma } M$
 $\langle \text{proof} \rangle$

lemma *measurable-right*[*measurable*]: $\text{right} \in \text{tree-sigma } M \rightarrow_M \text{tree-sigma } M$
 $\langle \text{proof} \rangle$

lemma *measurable-value'*: $\text{value} \in \text{restrict-space } (\text{tree-sigma } M) \ (-\{\text{Leaf}\}) \rightarrow_M M$
 $\langle \text{proof} \rangle$

lemma *measurable-value*[*measurable (raw)*]:

assumes $f \in X \rightarrow_M \text{tree-sigma } M$

and $\bigwedge x. x \in \text{space } X \implies f\ x \neq \text{Leaf}$

shows $(\lambda \omega. \text{value } (f\ \omega)) \in X \rightarrow_M M$

<proof>

lemma *measurable-Node* [*measurable*]:

$(\lambda(l,x,r). \text{Node } l\ x\ r) \in \text{tree-sigma } M \otimes_M M \otimes_M \text{tree-sigma } M \rightarrow_M \text{tree-sigma } M$

<proof>

lemma *measurable-Node'* [*measurable (raw)*]:

assumes [*measurable*]: $l \in B \rightarrow_M \text{tree-sigma } A$

assumes [*measurable*]: $x \in B \rightarrow_M A$

assumes [*measurable*]: $r \in B \rightarrow_M \text{tree-sigma } A$

shows $(\lambda y. \text{Node } (l\ y)\ (x\ y)\ (r\ y)) \in B \rightarrow_M \text{tree-sigma } A$

<proof>

lemma *measurable-rec-tree*[*measurable (raw)*]:

assumes $t: t \in B \rightarrow_M \text{tree-sigma } M$

assumes $l: l \in B \rightarrow_M A$

assumes $n: (\lambda(x, l, v, r, al, ar). n\ x\ l\ v\ r\ al\ ar) \in$

$(B \otimes_M \text{tree-sigma } M \otimes_M M \otimes_M \text{tree-sigma } M \otimes_M A \otimes_M A) \rightarrow_M A$ (**is** $?N \in ?M \rightarrow_M A$)

shows $(\lambda x. \text{rec-tree } (l\ x)\ (n\ x)\ (t\ x)) \in B \rightarrow_M A$

<proof>

lemma *measurable-case-tree* [*measurable (raw)*]:

assumes $t \in B \rightarrow_M \text{tree-sigma } M$

assumes $l \in B \rightarrow_M A$

assumes $(\lambda(x, l, v, r). n\ x\ l\ v\ r)$

$\in B \otimes_M \text{tree-sigma } M \otimes_M M \otimes_M \text{tree-sigma } M \rightarrow_M A$

shows $(\lambda x. \text{case-tree } (l\ x)\ (n\ x)\ (t\ x)) \in B \rightarrow_M (A :: 'a\ \text{measure})$

<proof>

hide-const (**open**) *left*

hide-const (**open**) *right*

end

28 Conditional Expectation

theory *Conditional-Expectation*

imports *Probability-Measure*

begin

28.1 Restricting a measure to a sub-sigma-algebra

definition *subalgebra*::'a measure \Rightarrow 'a measure \Rightarrow bool **where**
 $\text{subalgebra } M \ F = ((\text{space } F = \text{space } M) \wedge (\text{sets } F \subseteq \text{sets } M))$

lemma *sub-measure-space*:

assumes *i*: *subalgebra* *M F*

shows *measure-space* (*space M*) (*sets F*) (*emeasure M*)

<proof>

definition *restr-to-subalg*::'a measure \Rightarrow 'a measure \Rightarrow 'a measure **where**
 $\text{restr-to-subalg } M \ F = \text{measure-of } (\text{space } M) \ (\text{sets } F) \ (\text{emeasure } M)$

lemma *space-restr-to-subalg*:

$\text{space } (\text{restr-to-subalg } M \ F) = \text{space } M$

<proof>

lemma *sets-restr-to-subalg [measurable-cong]*:

assumes *subalgebra M F*

shows *sets* (*restr-to-subalg M F*) = *sets F*

<proof>

lemma *emeasure-restr-to-subalg*:

assumes *subalgebra M F*

$A \in \text{sets } F$

shows *emeasure* (*restr-to-subalg M F*) *A* = *emeasure M A*

<proof>

lemma *null-sets-restr-to-subalg*:

assumes *subalgebra M F*

shows *null-sets* (*restr-to-subalg M F*) = *null-sets M* \cap *sets F*

<proof>

lemma *AE-restr-to-subalg*:

assumes *subalgebra M F*

$AE \ x \text{ in } (\text{restr-to-subalg } M \ F). \ P \ x$

shows $AE \ x \text{ in } M. \ P \ x$

<proof>

lemma *AE-restr-to-subalg2*:

assumes *subalgebra M F*

$AE \ x \text{ in } M. \ P \ x$ **and** [*measurable*]: $P \in \text{measurable } F \ (\text{count-space } UNIV)$

shows $AE \ x \text{ in } (\text{restr-to-subalg } M \ F). \ P \ x$

<proof>

lemma *prob-space-restr-to-subalg*:

assumes *subalgebra M F*

prob-space M

shows *prob-space* (*restr-to-subalg M F*)

<proof>

lemma *finite-measure-restr-to-subalg*:
assumes *subalgebra M F*
finite-measure M
shows *finite-measure (restr-to-subalg M F)*
 $\langle \text{proof} \rangle$

lemma *measurable-in-subalg*:
assumes *subalgebra M F*
f ∈ measurable F N
shows *f ∈ measurable (restr-to-subalg M F) N*
 $\langle \text{proof} \rangle$

lemma *measurable-in-subalg'*:
assumes *subalgebra M F*
f ∈ measurable (restr-to-subalg M F) N
shows *f ∈ measurable F N*
 $\langle \text{proof} \rangle$

lemma *measurable-from-subalg*:
assumes *subalgebra M F*
f ∈ measurable F N
shows *f ∈ measurable M N*
 $\langle \text{proof} \rangle$

The following is the direct transposition of *nn_integral_subalgebra* (from *Nonnegative_Lebesgue_Integration*) in the current notations, with the removal of the useless assumption $f \geq 0$.

lemma *nn-integral-subalgebra2*:
assumes *subalgebra M F* **and** *[measurable]: f ∈ borel-measurable F*
shows $(\int^+ x. f x \partial(\text{restr-to-subalg } M F)) = (\int^+ x. f x \partial M)$
 $\langle \text{proof} \rangle$

The following is the direct transposition of *integral_subalgebra* (from *Bochner_Integration*) in the current notations.

lemma *integral-subalgebra2*:
fixes *f :: 'a ⇒ 'b::{\banach, second-countable-topology}*
assumes *subalgebra M F* **and**
[measurable]: f ∈ borel-measurable F
shows $(\int x. f x \partial(\text{restr-to-subalg } M F)) = (\int x. f x \partial M)$
 $\langle \text{proof} \rangle$

lemma *integrable-from-subalg*:
fixes *f :: 'a ⇒ 'b::{\banach, second-countable-topology}*
assumes *subalgebra M F*
integrable (restr-to-subalg M F) f
shows *integrable M f*
 $\langle \text{proof} \rangle$

lemma *integrable-in-subalg*:
fixes $f :: 'a \Rightarrow 'b :: \{\text{banach}, \text{second-countable-topology}\}$
assumes $[\text{measurable}]$: *subalgebra* $M F$
 $f \in \text{borel-measurable } F$
 $\text{integrable } M f$
shows *integrable* (*restr-to-subalg* $M F$) f
 $\langle \text{proof} \rangle$

28.2 Nonnegative conditional expectation

The conditional expectation of a function f , on a measure space M , with respect to a sub sigma algebra F , should be a function g which is F -measurable whose integral on any F -set coincides with the integral of f . Such a function is uniquely defined almost everywhere. The most direct construction is to use the measure $f dM$, restrict it to the sigma-algebra F , and apply the Radon-Nikodym theorem to write it as $g dM|_F$ for some F -measurable function g . Another classical construction for L^2 functions is done by orthogonal projection on F -measurable functions, and then extending by density to L^1 . The Radon-Nikodym point of view avoids the L^2 machinery, and works for all positive functions.

In this paragraph, we develop the definition and basic properties for nonnegative functions, as the basics of the general case. As in the definition of integrals, the nonnegative case is done with ennreal-valued functions, without any integrability assumption.

definition *nn-cond-exp* :: $'a \text{ measure} \Rightarrow 'a \text{ measure} \Rightarrow ('a \Rightarrow \text{ennreal}) \Rightarrow ('a \Rightarrow \text{ennreal})$

where

$\text{nn-cond-exp } M F f =$
 $(\text{if } f \in \text{borel-measurable } M \wedge \text{subalgebra } M F$
 $\text{then RN-deriv } (\text{restr-to-subalg } M F) (\text{restr-to-subalg } (\text{density } M f) F)$
 $\text{else } (\lambda \cdot 0))$

lemma

shows *borel-measurable-nn-cond-exp* $[\text{measurable}]$: *nn-cond-exp* $M F f \in \text{borel-measurable } F$

and *borel-measurable-nn-cond-exp2* $[\text{measurable}]$: *nn-cond-exp* $M F f \in \text{borel-measurable } M$

$\langle \text{proof} \rangle$

The good setting for conditional expectations is the situation where the subalgebra F gives rise to a sigma-finite measure space. To see what goes wrong if it is not sigma-finite, think of \mathbb{R} with the trivial sigma-algebra $\{\emptyset, \mathbb{R}\}$. In this case, conditional expectations have to be constant functions, so they have integral 0 or ∞ . This means that a positive integrable function can have no meaningful conditional expectation.

locale *sigma-finite-subalgebra* =

fixes $M F :: 'a \text{ measure}$
assumes $\text{subalg}: \text{subalgebra } M F$
and $\text{sigma-fin-subalg}: \text{sigma-finite-measure } (\text{restr-to-subalg } M F)$

lemma $\text{sigma-finite-subalgebra-is-sigma-finite}$:
assumes $\text{sigma-finite-subalgebra } M F$
shows $\text{sigma-finite-measure } M$
 $\langle \text{proof} \rangle$

sublocale $\text{sigma-finite-subalgebra} \subseteq \text{sigma-finite-measure}$
 $\langle \text{proof} \rangle$

Conditional expectations are very often used in probability spaces. This is a special case of the previous one, as we prove now.

locale $\text{finite-measure-subalgebra} = \text{finite-measure} +$
fixes $F :: 'a \text{ measure}$
assumes $\text{subalg}: \text{subalgebra } M F$

lemma $\text{finite-measure-subalgebra-is-sigma-finite}$:
assumes $\text{finite-measure-subalgebra } M F$
shows $\text{sigma-finite-subalgebra } M F$
 $\langle \text{proof} \rangle$

sublocale $\text{finite-measure-subalgebra} \subseteq \text{sigma-finite-subalgebra}$
 $\langle \text{proof} \rangle$

context $\text{sigma-finite-subalgebra}$
begin

The next lemma is arguably the most fundamental property of conditional expectation: when computing an expectation against an F -measurable function, it is equivalent to work with a function or with its F -conditional expectation.

This property (even for bounded test functions) characterizes conditional expectations, as the second lemma below shows. From this point on, we will only work with it, and forget completely about the definition using Radon-Nikodym derivatives.

lemma nn-cond-exp-intg :
assumes $[\text{measurable}]: f \in \text{borel-measurable } F \ g \in \text{borel-measurable } M$
shows $(\int^+ x. f x * \text{nn-cond-exp } M F g x \partial M) = (\int^+ x. f x * g x \partial M)$
 $\langle \text{proof} \rangle$

lemma $\text{nn-cond-exp-charact}$:
assumes $\bigwedge A. A \in \text{sets } F \implies (\int^+ x \in A. f x \partial M) = (\int^+ x \in A. g x \partial M)$ **and**
 $[\text{measurable}]: f \in \text{borel-measurable } M \ g \in \text{borel-measurable } F$
shows $A E x \text{ in } M. g x = \text{nn-cond-exp } M F f x$
 $\langle \text{proof} \rangle$

lemma *nn-cond-exp-F-meas*:

assumes $f \in \text{borel-measurable } F$

shows $AE\ x\ \text{in } M. f\ x = \text{nn-cond-exp } M\ F\ f\ x$

$\langle \text{proof} \rangle$

lemma *nn-cond-exp-prod*:

assumes $[measurable]: f \in \text{borel-measurable } F\ g \in \text{borel-measurable } M$

shows $AE\ x\ \text{in } M. f\ x * \text{nn-cond-exp } M\ F\ g\ x = \text{nn-cond-exp } M\ F\ (\lambda x. f\ x * g\ x)\ x$

$\langle \text{proof} \rangle$

lemma *nn-cond-exp-sum*:

assumes $[measurable]: f \in \text{borel-measurable } M\ g \in \text{borel-measurable } M$

shows $AE\ x\ \text{in } M. \text{nn-cond-exp } M\ F\ f\ x + \text{nn-cond-exp } M\ F\ g\ x = \text{nn-cond-exp } M\ F\ (\lambda x. f\ x + g\ x)\ x$

$\langle \text{proof} \rangle$

lemma *nn-cond-exp-cong*:

assumes $AE\ x\ \text{in } M. f\ x = g\ x$

and $[measurable]: f \in \text{borel-measurable } M\ g \in \text{borel-measurable } M$

shows $AE\ x\ \text{in } M. \text{nn-cond-exp } M\ F\ f\ x = \text{nn-cond-exp } M\ F\ g\ x$

$\langle \text{proof} \rangle$

lemma *nn-cond-exp-mono*:

assumes $AE\ x\ \text{in } M. f\ x \leq g\ x$

and $[measurable]: f \in \text{borel-measurable } M\ g \in \text{borel-measurable } M$

shows $AE\ x\ \text{in } M. \text{nn-cond-exp } M\ F\ f\ x \leq \text{nn-cond-exp } M\ F\ g\ x$

$\langle \text{proof} \rangle$

lemma *nested-subalg-is-sigma-finite*:

assumes $\text{subalgebra } M\ G\ \text{subalgebra } G\ F$

shows $\text{sigma-finite-subalgebra } M\ G$

$\langle \text{proof} \rangle$

lemma *nn-cond-exp-nested-subalg*:

assumes $\text{subalgebra } M\ G\ \text{subalgebra } G\ F$

and $[measurable]: f \in \text{borel-measurable } M$

shows $AE\ x\ \text{in } M. \text{nn-cond-exp } M\ F\ f\ x = \text{nn-cond-exp } M\ F\ (\text{nn-cond-exp } M\ G\ f)\ x$

$\langle \text{proof} \rangle$

end

28.3 Real conditional expectation

Once conditional expectations of positive functions are defined, the definition for real-valued functions follows readily, by taking the difference of positive and negative parts. One could also define a conditional expectation of vector-space valued functions, as in `Bochner_Integral`, but since

the real-valued case is the most important, and quicker to formalize, I concentrate on it. (It is also essential for the case of the most general Pettis integral.)

definition *real-cond-exp* :: 'a measure \Rightarrow 'a measure \Rightarrow ('a \Rightarrow real) \Rightarrow ('a \Rightarrow real)
where

real-cond-exp $M F f =$
 $(\lambda x. \text{enn2real}(\text{nn-cond-exp } M F (\lambda x. \text{ennreal } (f x)) x) - \text{enn2real}(\text{nn-cond-exp } M F (\lambda x. \text{ennreal } (-f x)) x))$

lemma

shows *borel-measurable-cond-exp* [*measurable*]: *real-cond-exp* $M F f \in \text{borel-measurable } F$

and *borel-measurable-cond-exp2* [*measurable*]: *real-cond-exp* $M F f \in \text{borel-measurable } M$

<proof>

context *sigma-finite-subalgebra*

begin

lemma *real-cond-exp-abs*:

assumes [*measurable*]: $f \in \text{borel-measurable } M$

shows $\text{AE } x \text{ in } M. \text{abs}(\text{real-cond-exp } M F f x) \leq \text{nn-cond-exp } M F (\lambda x. \text{ennreal } (\text{abs}(f x))) x$

<proof>

The next lemma shows that the conditional expectation is an F -measurable function whose average against an F -measurable function f coincides with the average of the original function against f . It is obtained as a consequence of the same property for the positive conditional expectation, taking the difference of the positive and the negative part. The proof is given first assuming $f \geq 0$ for simplicity, and then extended to the general case in the subsequent lemma. The idea of the proof is essentially trivial, but the implementation is slightly tedious as one should check all the integrability properties of the different parts, and go back and forth between positive integral and signed integrals, and between real-valued functions and ennreal-valued functions.

Once this lemma is available, we will use it to characterize the conditional expectation, and never come back to the original technical definition, as we did in the case of the nonnegative conditional expectation.

lemma *real-cond-exp-intg-fpos*:

assumes *integrable* $M (\lambda x. f x * g x)$ **and** *f-pos[simp]*: $\bigwedge x. f x \geq 0$ **and**
 $[\text{measurable}]: f \in \text{borel-measurable } F g \in \text{borel-measurable } M$

shows *integrable* $M (\lambda x. f x * \text{real-cond-exp } M F g x)$

$(\int x. f x * \text{real-cond-exp } M F g x \partial M) = (\int x. f x * g x \partial M)$

<proof>

lemma *real-cond-exp-intg*:

assumes *integrable* M $(\lambda x. f\ x * g\ x)$ **and**
 $[measurable]: f \in \text{borel-measurable } F\ g \in \text{borel-measurable } M$
shows *integrable* M $(\lambda x. f\ x * \text{real-cond-exp } M\ F\ g\ x)$
 $(\int x. f\ x * \text{real-cond-exp } M\ F\ g\ x\ \partial M) = (\int x. f\ x * g\ x\ \partial M)$
 $\langle \text{proof} \rangle$

lemma *real-cond-exp-intA*:
assumes $[measurable]: \text{integrable } M\ f\ A \in \text{sets } F$
shows $(\int x \in A. f\ x\ \partial M) = (\int x \in A. \text{real-cond-exp } M\ F\ f\ x\ \partial M)$
 $\langle \text{proof} \rangle$

lemma *real-cond-exp-int [intro]*:
assumes *integrable* $M\ f$
shows *integrable* M $(\text{real-cond-exp } M\ F\ f)$ $(\int x. \text{real-cond-exp } M\ F\ f\ x\ \partial M) =$
 $(\int x. f\ x\ \partial M)$
 $\langle \text{proof} \rangle$

lemma *real-cond-exp-charact*:
assumes $\bigwedge A. A \in \text{sets } F \implies (\int x \in A. f\ x\ \partial M) = (\int x \in A. g\ x\ \partial M)$
and $[measurable]: \text{integrable } M\ f\ \text{integrable } M\ g$
 $g \in \text{borel-measurable } F$
shows $\text{AE } x \text{ in } M. \text{real-cond-exp } M\ F\ f\ x = g\ x$
 $\langle \text{proof} \rangle$

lemma *real-cond-exp-F-meas [intro, simp]*:
assumes *integrable* $M\ f$
 $f \in \text{borel-measurable } F$
shows $\text{AE } x \text{ in } M. \text{real-cond-exp } M\ F\ f\ x = f\ x$
 $\langle \text{proof} \rangle$

lemma *real-cond-exp-mult*:
assumes $[measurable]: f \in \text{borel-measurable } F\ g \in \text{borel-measurable } M\ \text{integrable}$
 M $(\lambda x. f\ x * g\ x)$
shows $\text{AE } x \text{ in } M. \text{real-cond-exp } M\ F\ (\lambda x. f\ x * g\ x)\ x = f\ x * \text{real-cond-exp } M$
 $F\ g\ x$
 $\langle \text{proof} \rangle$

lemma *real-cond-exp-add [intro]*:
assumes $[measurable]: \text{integrable } M\ f\ \text{integrable } M\ g$
shows $\text{AE } x \text{ in } M. \text{real-cond-exp } M\ F\ (\lambda x. f\ x + g\ x)\ x = \text{real-cond-exp } M\ F\ f\ x$
 $+ \text{real-cond-exp } M\ F\ g\ x$
 $\langle \text{proof} \rangle$

lemma *real-cond-exp-cong*:
assumes $\text{ae}: \text{AE } x \text{ in } M. f\ x = g\ x$ **and** $[measurable]: f \in \text{borel-measurable } M\ g$
 $\in \text{borel-measurable } M$
shows $\text{AE } x \text{ in } M. \text{real-cond-exp } M\ F\ f\ x = \text{real-cond-exp } M\ F\ g\ x$
 $\langle \text{proof} \rangle$

lemma *real-cond-exp-cmult* [*intro*, *simp*]:
 fixes $c::\text{real}$
 assumes *integrable* $M\ f$
 shows $AE\ x\ in\ M. \text{real-cond-exp}\ M\ F\ (\lambda x. c * f\ x)\ x = c * \text{real-cond-exp}\ M\ F\ f\ x$
 $\langle proof \rangle$

lemma *real-cond-exp-cdiv* [*intro*, *simp*]:
 fixes $c::\text{real}$
 assumes *integrable* $M\ f$
 shows $AE\ x\ in\ M. \text{real-cond-exp}\ M\ F\ (\lambda x. f\ x / c)\ x = \text{real-cond-exp}\ M\ F\ f\ x / c$
 $\langle proof \rangle$

lemma *real-cond-exp-diff* [*intro*, *simp*]:
 assumes [*measurable*]: *integrable* $M\ f$ *integrable* $M\ g$
 shows $AE\ x\ in\ M. \text{real-cond-exp}\ M\ F\ (\lambda x. f\ x - g\ x)\ x = \text{real-cond-exp}\ M\ F\ f\ x - \text{real-cond-exp}\ M\ F\ g\ x$
 $\langle proof \rangle$

lemma *real-cond-exp-pos* [*intro*]:
 assumes $AE\ x\ in\ M. f\ x \geq 0$ **and** [*measurable*]: $f \in \text{borel-measurable}\ M$
 shows $AE\ x\ in\ M. \text{real-cond-exp}\ M\ F\ f\ x \geq 0$
 $\langle proof \rangle$

lemma *real-cond-exp-mono*:
 assumes $AE\ x\ in\ M. f\ x \leq g\ x$ **and** [*measurable*]: *integrable* $M\ f$ *integrable* $M\ g$
 shows $AE\ x\ in\ M. \text{real-cond-exp}\ M\ F\ f\ x \leq \text{real-cond-exp}\ M\ F\ g\ x$
 $\langle proof \rangle$

lemma (*in* $-$) *measurable-P-restriction* [*measurable* (*raw*)]:
 assumes [*measurable*]: *Measurable.pred* $M\ P\ A \in \text{sets}\ M$
 shows $\{x \in A. P\ x\} \in \text{sets}\ M$
 $\langle proof \rangle$

lemma *real-cond-exp-gr-c*:
 assumes [*measurable*]: *integrable* $M\ f$
and $AE: AE\ x\ in\ M. f\ x > c$
 shows $AE\ x\ in\ M. \text{real-cond-exp}\ M\ F\ f\ x > c$
 $\langle proof \rangle$

lemma *real-cond-exp-less-c*:
 assumes [*measurable*]: *integrable* $M\ f$
and $AE\ x\ in\ M. f\ x < c$
 shows $AE\ x\ in\ M. \text{real-cond-exp}\ M\ F\ f\ x < c$
 $\langle proof \rangle$

lemma *real-cond-exp-ge-c*:
 assumes [*measurable*]: *integrable* $M\ f$

and $AE\ x\ in\ M. f\ x \geq c$
shows $AE\ x\ in\ M. real\ cond\ exp\ M\ F\ f\ x \geq c$
 $\langle proof \rangle$

lemma *real-cond-exp-le-c*:
assumes $[measurable]: integrable\ M\ f$
and $AE\ x\ in\ M. f\ x \leq c$
shows $AE\ x\ in\ M. real\ cond\ exp\ M\ F\ f\ x \leq c$
 $\langle proof \rangle$

lemma *real-cond-exp-mono-strict*:
assumes $AE\ x\ in\ M. f\ x < g\ x$ **and** $[measurable]: integrable\ M\ f\ integrable\ M\ g$
shows $AE\ x\ in\ M. real\ cond\ exp\ M\ F\ f\ x < real\ cond\ exp\ M\ F\ g\ x$
 $\langle proof \rangle$

lemma *real-cond-exp-nested-subalg* $[intro, simp]$:
assumes *subalgebra* $M\ G\ subalgebra\ G\ F$
and $[measurable]: integrable\ M\ f$
shows $AE\ x\ in\ M. real\ cond\ exp\ M\ F\ (real\ cond\ exp\ M\ G\ f)\ x = real\ cond\ exp\ M\ F\ f\ x$
 $\langle proof \rangle$

lemma *real-cond-exp-sum* $[intro, simp]$:
fixes $f::'b \Rightarrow 'a \Rightarrow real$
assumes $[measurable]: \bigwedge i. integrable\ M\ (f\ i)$
shows $AE\ x\ in\ M. real\ cond\ exp\ M\ F\ (\lambda x. \sum_{i \in I}. f\ i\ x)\ x = (\sum_{i \in I}. real\ cond\ exp\ M\ F\ (f\ i)\ x)$
 $\langle proof \rangle$

Jensen’s inequality, describing the behavior of the integral under a convex function, admits a version for the conditional expectation, as follows.

theorem *real-cond-exp-jensens-inequality*:
fixes $q :: real \Rightarrow real$
assumes $X: integrable\ M\ X\ AE\ x\ in\ M. X\ x \in I$
assumes $I: I = \{a <..< b\} \vee I = \{a <..\} \vee I = \{..< b\} \vee I = UNIV$
assumes $q: integrable\ M\ (\lambda x. q\ (X\ x))\ convex\ on\ I\ q\ q \in borel\ measurable\ borel$
shows $AE\ x\ in\ M. real\ cond\ exp\ M\ F\ X\ x \in I$
 $AE\ x\ in\ M. q\ (real\ cond\ exp\ M\ F\ X\ x) \leq real\ cond\ exp\ M\ F\ (\lambda x. q\ (X\ x))\ x$
 $\langle proof \rangle$

Jensen’s inequality does not imply that $q(E(X|F))$ is integrable, as it only proves an upper bound for it. Indeed, this is not true in general, as the following counterexample shows:

on $[1, \infty)$ with Lebesgue measure, let F be the sigma-algebra generated by the intervals $[n, n+1)$ for integer n . Let $q(x) = -\sqrt{x}$ for $x \geq 0$. Define X which is equal to $1/n$ over $[n, n+1/n)$ and 2^{-n} on $[n+1/n, n+1)$. Then X is integrable as $\sum 1/n^2 < \infty$, and $q(X)$ is integrable as $\sum 1/n^{3/2} < \infty$. On the other hand, $E(X|F)$ is essentially equal to $1/n^2$ on $[n, n+1)$ (we

neglect the term 2^{-n} , we only put it there because X should take its values in $I = (0, \infty)$. Hence, $q(E(X|F))$ is equal to $-1/n$ on $[n, n+1)$, hence it is not integrable.

However, this counterexample is essentially the only situation where this function is not integrable, as shown by the next lemma.

lemma *integrable-convex-cond-exp*:

fixes $q :: \text{real} \Rightarrow \text{real}$

assumes X : *integrable* M X *AE* x *in* M . X $x \in I$

assumes I : $I = \{a <..< b\} \vee I = \{a <..\} \vee I = \{..< b\} \vee I = \text{UNIV}$

assumes q : *integrable* M $(\lambda x. q (X x))$ *convex-on* I q $q \in \text{borel-measurable borel}$

assumes H : *emeasure* M $(\text{space } M) = \infty \implies 0 \in I$

shows *integrable* M $(\lambda x. q (\text{real-cond-exp } M F X x))$

<proof>

end

end

theory *Essential-Supremum*

imports *HOL-Analysis.Analysis*

begin

lemma *ae-filter-eq-bot-iff*: *ae-filter* $M = \text{bot} \iff \text{emeasure } M (\text{space } M) = 0$

<proof>

29 The essential supremum

In this paragraph, we define the essential supremum and give its basic properties. The essential supremum of a function is its maximum value if one is allowed to throw away a set of measure 0. It is convenient to define it to be infinity for non-measurable functions, as it allows for neater statements in general. This is a prerequisite to define the space L^∞ .

definition *esssup*:: $'a \text{ measure} \Rightarrow ('a \Rightarrow 'b :: \{\text{second-countable-topology, dense-linorder, linorder-topology, complete-linorder}\}) \Rightarrow 'b$

where *esssup* M $f = (\text{if } f \in \text{borel-measurable } M \text{ then } \text{Limsup } (\text{ae-filter } M) f \text{ else } \text{top})$

lemma *esssup-non-measurable*: $f \notin M \rightarrow_M \text{borel} \implies \text{esssup } M f = \text{top}$

<proof>

lemma *esssup-eq-AE*:

assumes f : $f \in M \rightarrow_M \text{borel}$ **shows** *esssup* M $f = \text{Inf } \{z. \text{AE } x \text{ in } M. f x \leq z\}$

<proof>

lemma *esssup-eq*: $f \in M \rightarrow_M \text{borel} \implies \text{esssup } M f = \text{Inf } \{z. \text{emeasure } M \{x \in$

$space\ M.\ f\ x > z\} = 0\}$
 $\langle proof \rangle$

lemma *esssup-zero-measure*:

$emeasure\ M\ \{x \in space\ M.\ f\ x > esssup\ M\ f\} = 0$
 $\langle proof \rangle$

lemma *esssup-AE*: $AE\ x\ in\ M.\ f\ x \leq esssup\ M\ f$
 $\langle proof \rangle$

lemma *esssup-pos-measure*:

$f \in borel\text{-}measurable\ M \implies z < esssup\ M\ f \implies emeasure\ M\ \{x \in space\ M.\ f\ x > z\} > 0$
 $\langle proof \rangle$

lemma *esssup-I [intro]*: $f \in borel\text{-}measurable\ M \implies AE\ x\ in\ M.\ f\ x \leq c \implies esssup\ M\ f \leq c$
 $\langle proof \rangle$

lemma *esssup-AE-mono*: $f \in borel\text{-}measurable\ M \implies AE\ x\ in\ M.\ f\ x \leq g\ x \implies esssup\ M\ f \leq esssup\ M\ g$
 $\langle proof \rangle$

lemma *esssup-mono*: $f \in borel\text{-}measurable\ M \implies (\bigwedge x.\ f\ x \leq g\ x) \implies esssup\ M\ f \leq esssup\ M\ g$
 $\langle proof \rangle$

lemma *esssup-AE-cong*:

$f \in borel\text{-}measurable\ M \implies g \in borel\text{-}measurable\ M \implies AE\ x\ in\ M.\ f\ x = g\ x \implies esssup\ M\ f = esssup\ M\ g$
 $\langle proof \rangle$

lemma *esssup-const*: $emeasure\ M\ (space\ M) \neq 0 \implies esssup\ M\ (\lambda x.\ c) = c$
 $\langle proof \rangle$

lemma *esssup-cmult*: **assumes** $c > (0::real)$ **shows** $esssup\ M\ (\lambda x.\ c * f\ x::ereal) = c * esssup\ M\ f$
 $\langle proof \rangle$

lemma *esssup-add*:

$esssup\ M\ (\lambda x.\ f\ x + g\ x::ereal) \leq esssup\ M\ f + esssup\ M\ g$
 $\langle proof \rangle$

lemma *esssup-zero-space*:

$emeasure\ M\ (space\ M) = 0 \implies f \in borel\text{-}measurable\ M \implies esssup\ M\ f = (-\infty::ereal)$
 $\langle proof \rangle$

end

30 Stopping times

```
theory Stopping-Time
  imports HOL-Analysis.Analysis
begin
```

30.1 Stopping Time

This is also called strong stopping time. Then stopping time is T with alternative is $T x < t$ measurable.

definition *stopping-time* :: $('t::\text{linorder} \Rightarrow 'a \text{ measure}) \Rightarrow ('a \Rightarrow 't) \Rightarrow \text{bool}$
where

stopping-time $F T = (\forall t. \text{Measurable.pred } (F t) (\lambda x. T x \leq t))$

lemma *stopping-time-cong*: $(\bigwedge t x. x \in \text{space } (F t) \implies T x = S x) \implies \text{stopping-time } F T = \text{stopping-time } F S$
 $\langle \text{proof} \rangle$

lemma *stopping-timeD*: $\text{stopping-time } F T \implies \text{Measurable.pred } (F t) (\lambda x. T x \leq t)$
 $\langle \text{proof} \rangle$

lemma *stopping-timeD2*: $\text{stopping-time } F T \implies \text{Measurable.pred } (F t) (\lambda x. t < T x)$
 $\langle \text{proof} \rangle$

lemma *stopping-timeI*[intro?]: $(\bigwedge t. \text{Measurable.pred } (F t) (\lambda x. T x \leq t)) \implies \text{stopping-time } F T$
 $\langle \text{proof} \rangle$

lemma *measurable-stopping-time*:
fixes $T :: 'a \Rightarrow 't::\{\text{linorder-topology, second-countable-topology}\}$
assumes $T: \text{stopping-time } F T$
and $M: \bigwedge t. \text{sets } (F t) \subseteq \text{sets } M \bigwedge t. \text{space } (F t) = \text{space } M$
shows $T \in M \rightarrow_M \text{borel}$
 $\langle \text{proof} \rangle$

lemma *stopping-time-const*: $\text{stopping-time } F (\lambda x. c)$
 $\langle \text{proof} \rangle$

lemma *stopping-time-min*:
 $\text{stopping-time } F T \implies \text{stopping-time } F S \implies \text{stopping-time } F (\lambda x. \min (T x) (S x))$
 $\langle \text{proof} \rangle$

lemma *stopping-time-max*:
 $\text{stopping-time } F T \implies \text{stopping-time } F S \implies \text{stopping-time } F (\lambda x. \max (T x) (S x))$
 $\langle \text{proof} \rangle$

31 Filtration

locale *filtration* =

fixes $\Omega :: 'a \text{ set}$ **and** $F :: 't :: \{\text{linorder-topology, second-countable-topology}\} \Rightarrow 'a \text{ measure}$

assumes *space-F*: $\bigwedge i. \text{space } (F \ i) = \Omega$

assumes *sets-F-mono*: $\bigwedge i \ j. i \leq j \implies \text{sets } (F \ i) \leq \text{sets } (F \ j)$

begin

31.1 σ -algebra of a Stopping Time

definition *pre-sigma* :: $('a \Rightarrow 't) \Rightarrow 'a \text{ measure}$

where

pre-sigma $T = \text{sigma } \Omega \ \{A. \forall t. \{\omega \in A. T \ \omega \leq t\} \in \text{sets } (F \ t)\}$

lemma *space-pre-sigma*: $\text{space } (\text{pre-sigma } T) = \Omega$

<proof>

lemma *measure-pre-sigma[simp]*: $\text{emeasure } (\text{pre-sigma } T) = (\lambda \cdot. 0)$

<proof>

lemma *sigma-algebra-pre-sigma*:

assumes T : *stopping-time* $F \ T$

shows *sigma-algebra* $\Omega \ \{A. \forall t. \{\omega \in A. T \ \omega \leq t\} \in \text{sets } (F \ t)\}$

<proof>

lemma *sets-pre-sigma*: $\text{stopping-time } F \ T \implies \text{sets } (\text{pre-sigma } T) = \{A. \forall t. \{\omega \in A. T \ \omega \leq t\} \in \text{sets } (F \ t)\}$

<proof>

lemma *sets-pre-sigmaI*: $\text{stopping-time } F \ T \implies (\bigwedge t. \{\omega \in A. T \ \omega \leq t\} \in \text{sets } (F \ t)) \implies A \in \text{sets } (\text{pre-sigma } T)$

<proof>

lemma *pred-pre-sigmaI*:

assumes T : *stopping-time* $F \ T$

shows $(\bigwedge t. \text{Measurable.pred } (F \ t) \ (\lambda \omega. P \ \omega \wedge T \ \omega \leq t)) \implies \text{Measurable.pred } (\text{pre-sigma } T) \ P$

<proof>

lemma *sets-pre-sigmaD*: $\text{stopping-time } F \ T \implies A \in \text{sets } (\text{pre-sigma } T) \implies \{\omega \in A. T \ \omega \leq t\} \in \text{sets } (F \ t)$

<proof>

lemma *stopping-time-le-const*: $\text{stopping-time } F \ T \implies s \leq t \implies \text{Measurable.pred } (F \ t) \ (\lambda \omega. T \ \omega \leq s)$

<proof>

lemma *measurable-stopping-time-pre-sigma*:

assumes T : *stopping-time* $F \ T$ **shows** $T \in \text{pre-sigma } T \rightarrow_M \text{borel}$

$\langle \text{proof} \rangle$

lemma *mono-pre-sigma*:

assumes T : *stopping-time* F T **and** S : *stopping-time* F S

and le : $\bigwedge \omega. \omega \in \Omega \implies T \ \omega \leq S \ \omega$

shows $\text{sets } (\text{pre-sigma } T) \subseteq \text{sets } (\text{pre-sigma } S)$

$\langle \text{proof} \rangle$

lemma *stopping-time-less-const*:

assumes T : *stopping-time* F T **shows** $\text{Measurable.pred } (F \ t) \ (\lambda \omega. T \ \omega < t)$

$\langle \text{proof} \rangle$

lemma *stopping-time-eq-const*: *stopping-time* F $T \implies \text{Measurable.pred } (F \ t) \ (\lambda \omega. T \ \omega = t)$

$\langle \text{proof} \rangle$

lemma *stopping-time-less*:

assumes T : *stopping-time* F T **and** S : *stopping-time* F S

shows $\text{Measurable.pred } (\text{pre-sigma } T) \ (\lambda \omega. T \ \omega < S \ \omega)$

$\langle \text{proof} \rangle$

end

lemma *stopping-time-SUP-enat*:

fixes $T :: \text{nat} \Rightarrow 'a \Rightarrow \text{enat}$

shows $(\bigwedge i. \text{stopping-time } F \ (T \ i)) \implies \text{stopping-time } F \ (\text{SUP } i. T \ i)$

$\langle \text{proof} \rangle$

lemma *less-eSuc-iff*: $a < \text{eSuc } b \longleftrightarrow (a \leq b \wedge a \neq \infty)$

$\langle \text{proof} \rangle$

lemma *stopping-time-Inf-enat*:

fixes $F :: \text{enat} \Rightarrow 'a \text{ measure}$

assumes F : *filtration* Ω F

assumes P : $\bigwedge i. \text{Measurable.pred } (F \ i) \ (P \ i)$

shows *stopping-time* $F \ (\lambda \omega. \text{Inf } \{i. P \ i \ \omega\})$

$\langle \text{proof} \rangle$

lemma *stopping-time-Inf-nat*:

fixes $F :: \text{nat} \Rightarrow 'a \text{ measure}$

assumes F : *filtration* Ω F

assumes P : $\bigwedge i. \text{Measurable.pred } (F \ i) \ (P \ i)$ **and** wf : $\bigwedge i \ \omega. \omega \in \Omega \implies \exists n. P \ n \ \omega$

shows *stopping-time* $F \ (\lambda \omega. \text{Inf } \{i. P \ i \ \omega\})$

$\langle \text{proof} \rangle$

end

```
theory Probability
imports
  Central-Limit-Theorem
  Discrete-Topology
  PMF-Impl
  Projective-Limit
  Random-Permutations
  SPMF
  Product-PMF
  Hoeffding
  Stream-Space
  Tree-Space
  Conditional-Expectation
  Essential-Supremum
  Stopping-Time
begin

end
```