Quasi-Borel Spaces

Michikazu Hirata, Yasuhiko Minamide, Tetsuya Sato

March 17, 2025

Abstract

The notion of quasi-Borel spaces was introduced by Heunen et al. [1]. The theory provides a suitable denotational model for higher-order probabilistic programming languages with continuous distributions.

This entry is a formalization of the theory of quasi-Borel spaces, including construction of quasi-Borel spaces (product, coproduct, function spaces), the adjunction between the category of measurable spaces and the category of quasi-Borel spaces, and the probability monad on quasi-Borel spaces. This entry also contains the formalization of the Bayesian regression presented in the work of Heunen et al.

This work is a part of the work by same authors, *Program Logic for Higher-Order Probabilistic Programs in Isabelle/HOL*, which will be published in proceedings of the 16th International Symposium on Functional and Logic Programming (FLOPS 2022).

Contents

1 Standard Borel Spaces

theory *StandardBorel*
imports *HOL−Probability.Probability*
begin

A standard Borel space is the Borel space associated with a Polish space. Here, we define standard Borel spaces in another, but equivallent, way. See [1] Proposition 5.

abbreviation *real-borel* ≡ *borel* :: *real measure*
abbreviation *nat-borel* ≡ *borel* :: *nat measure*
abbreviation *ennreal-borel* ≡ *borel* :: *ennreal measure*
abbreviation *bool-borel* ≡ *borel* :: *bool measure*

1.1 Definition

locale *standard-borel* =
  fixes M :: *'a measure*
  assumes *exist-fg*: $\exists f \in M \rightarrow_M$ *real-borel*. $\exists g \in$ *real-borel* $\rightarrow_M M$.
    $\forall x \in$ *space* M. $(g \circ f)\ x = x$
begin

abbreviation *fg* ≡ (*SOME k*. (*fst k*) $\in M \rightarrow_M$ *real-borel* $\wedge$
  (*snd k*) $\in$ *real-borel* $\rightarrow_M M \wedge$
  ($\forall x \in$ *space* M. ((*snd k*) $\circ$ (*fst k*)) $x = x$))

definition f ≡ (*fst fg*)
definition g ≡ (*snd fg*)

lemma
  shows *f-meas*[*simp*,*measurable*] : $f \in M \rightarrow_M$ *real-borel*
    and *g-meas*[*simp*,*measurable*] : $g \in$ *real-borel* $\rightarrow_M M$
    and *gf-comp-id*[*simp*]: $\bigwedge x.\ x \in$ *space* $M \Longrightarrow (g \circ f)\ x = x$
      $\bigwedge x.\ x \in$ *space* $M \Longrightarrow g\ (f\ x) = x$
⟨*proof*⟩

lemma *standard-borel-sets*[*simp*]:
  assumes *sets* M = *sets* Y
  shows *standard-borel* Y
  ⟨*proof*⟩

lemma *f-inj*:
  inj-on f (*space* M)
  ⟨*proof*⟩

lemma *singleton-sets*:
  assumes $x \in$ *space* M
    shows $\{x\} \in$ *sets* M
⟨*proof*⟩

lemma *countable-space-discrete*:
  assumes *countable* (*space* M)
  shows *sets* M = *sets* (*count-space* (*space* M))
⟨*proof*⟩

end

lemma *standard-borelI*:
assumes $f \in Y \to_M$ *real-borel*
$g \in$ *real-borel* $\to_M Y$
and $\bigwedge y.\ y \in$ *space* $Y \Longrightarrow (g \circ f)\ y = y$
shows *standard-borel Y*
⟨*proof*⟩

locale *standard-borel-space-UNIV* = *standard-borel* +
assumes *space-UNIV*:*space M* = *UNIV*
begin

lemma *gf-comp-id'*[*simp*]:
$g \circ f = id\ g\ (f\ x) = x$
⟨*proof*⟩

lemma *f-inj'*:
inj f
⟨*proof*⟩

lemma *g-surj'*:
surj g
⟨*proof*⟩

end

lemma *standard-borel-space-UNIVI*:
assumes $f \in Y \to_M$ *real-borel*
$g \in$ *real-borel* $\to_M Y$
$(g \circ f) = id$
and *space Y* = *UNIV*
shows *standard-borel-space-UNIV Y*
⟨*proof*⟩

lemma *standard-borel-space-UNIVI'*:
assumes *standard-borel Y*
and *space Y* = *UNIV*
shows *standard-borel-space-UNIV Y*
⟨*proof*⟩

1.2 $\mathbb{R}$, $\mathbb{N}$, Boolean, $[0, \infty]$

$\mathbb{R}$ is a standard Borel space.

interpretation *real* : *standard-borel-space-UNIV real-borel*
⟨*proof*⟩

A non-empty Borel subspace of $\mathbb{R}$ is also a standard Borel space.

lemma *real-standard-borel-subset*:
  assumes $U \in$ *sets real-borel*
    and $U \neq \{\}$
  shows *standard-borel* (*restrict-space real-borel* U)
⟨*proof*⟩

A non-empty measurable subset of a standard Borel space is also a standard Borel space.

lemma(**in** *standard-borel*) *standard-borel-subset*:
  assumes $U \in$ *sets* M
    $U \neq \{\}$
  shows *standard-borel* (*restrict-space* M U)
⟨*proof*⟩

$\mathbb{N}$ is a standard Borel space.

interpretation *nat* : *standard-borel-space-UNIV nat-borel*
⟨*proof*⟩

For a countable space X, X is a standard Borel space iff X is a discrete space.

lemma *countable-standard-iff*:
  assumes *space* $X \neq \{\}$
    and *countable* (*space* X)
  shows *standard-borel* $X \longleftrightarrow$ *sets* X = *sets* (*count-space* (*space* X))
⟨*proof*⟩

$\mathbb{B}$ is a standard Borel space.

lemma *to-bool-measurable*:
  assumes f −‘ {*True*} ∩ *space* M ∈ *sets* M
  shows $f \in M \rightarrow_M$ *bool-borel*
⟨*proof*⟩

interpretation *bool* : *standard-borel-space-UNIV bool-borel*
  ⟨*proof*⟩

$[0, \infty]$ (the set of extended non-negative real numbers) is a standard Borel space.

interpretation *ennreal* : *standard-borel-space-UNIV ennreal-borel*
⟨*proof*⟩

1.3 $\mathbb{R} \times \mathbb{R}$

definition *real-to-01open* :: *real* ⇒ *real* **where**
real-to-01open $r \equiv$ *arctan* r / *pi* + 1 / 2

definition *real-to-01open-inverse* :: *real* ⇒ *real* **where**
real-to-01open-inverse $r \equiv$ *tan* (*pi* ∗ r − (*pi* / 2))

lemma *real-to-01open-inverse-correct*:
real-to-01open-inverse ∘ *real-to-01open* = *id*
⟨*proof*⟩

lemma *real-to-01open-inverse-correct′*:
assumes *0* < *r r* < *1*
shows *real-to-01open* (*real-to-01open-inverse r*) = *r*
⟨*proof*⟩

lemma *real-to-01open-01* :
0 < *real-to-01open r* ∧ *real-to-01open r* < *1*
⟨*proof*⟩

lemma *real-to-01open-continuous*:
continuous-on UNIV real-to-01open
⟨*proof*⟩

lemma *real-to-01open-inverse-continuous*:
continuous-on {*0*<..<*1*} *real-to-01open-inverse*
⟨*proof*⟩

lemma *real-to-01open-inverse-measurable*:
real-to-01open-inverse ∈ *restrict-space real-borel* {*0*<..<*1*} $\to_M$ *real-borel*
⟨*proof*⟩

fun *r01-binary-expansion′′* :: *real* ⇒ *nat* ⇒ *nat* × *real* × *real* **where**
r01-binary-expansion′′ r 0 = (*if 1/2* ≤ *r then* (*1*,*1* ,*1/2*)
else (*0*,*1/2*, *0*)) |
r01-binary-expansion′′ r (*Suc n*) = (*let* (-,*ur*,*lr*) = *r01-binary-expansion′′ r n*;
k = (*ur* + *lr*)/*2 in*
(*if k* ≤ *r then* (*1*,*ur*,*k*)
else (*0*,*k*,*lr*)))

a_n where $r = 0.a_0a_1a_2....$ for $0 < r < 1$.

definition *r01-binary-expansion′* :: *real* ⇒ *nat* ⇒ *nat* **where**
r01-binary-expansion′ r n ≡ *fst* (*r01-binary-expansion′′ r n*)

$a_n = 0$ or 1.

lemma *real01-binary-expansion′-0or1*:
r01-binary-expansion′ r n ∈ {*0*,*1*}
⟨*proof*⟩

definition *r01-binary-sum* :: (*nat* ⇒ *nat*) ⇒ *nat* ⇒ *real* **where**
r01-binary-sum a n ≡ (∑ *i*=*0*..*n*. *real* (*a i*) ∗ ((*1/2*) ^ (*Suc i*)))

definition *r01-binary-sum-lim* :: (*nat* ⇒ *nat*) ⇒ *real* **where**
r01-binary-sum-lim ≡ *lim* ∘ *r01-binary-sum*

definition *r01-binary-expression* :: *real* ⇒ *nat* ⇒ *real* **where**
r01-binary-expression ≡ *r01-binary-sum* ∘ *r01-binary-expansion′*

lemma *r01-binary-expansion-lr-r-ur*:
assumes *0* < *r* *r* < *1*
shows (*snd* (*snd* (*r01-binary-expansion″ r n*))) ≤ *r* ∧
r < (*fst* (*snd* (*r01-binary-expansion″ r n*)))
⟨*proof*⟩

$0 \leq lr \wedge lr < ur \wedge ur \leq 1.$

lemma *r01-binary-expansion-lr-ur-nn*:
shows *0* ≤ *snd* (*snd* (*r01-binary-expansion″ r n*)) ∧
snd (*snd* (*r01-binary-expansion″ r n*)) < *fst* (*snd* (*r01-binary-expansion″ r n*)) ∧
fst (*snd* (*r01-binary-expansion″ r n*)) ≤ *1*
⟨*proof*⟩

lemma *r01-binary-expansion-diff*:
shows (*fst* (*snd* (*r01-binary-expansion″ r n*))) − (*snd* (*snd* (*r01-binary-expansion″ r n*))) = (*1/2*)^(*Suc n*)
⟨*proof*⟩

$lrn = Sn.$

lemma *r01-binary-expression-eq-lr*:
snd (*snd* (*r01-binary-expansion″ r n*)) = *r01-binary-expression r n*
⟨*proof*⟩

lemma *r01-binary-expression′-sum-range*:
∃ *k*::*nat*. (*snd* (*snd* (*r01-binary-expansion″ r n*))) = *real k*/*2*^(*Suc n*) ∧
k < *2*^(*Suc n*) ∧
((*r01-binary-expansion′ r n*) = *0* ⟶ *even k*) ∧
((*r01-binary-expansion′ r n*) = *1* ⟶ *odd k*)
⟨*proof*⟩

$an = bn \leftrightarrow Sn = S'n.$

lemma *r01-binary-expansion′-expression-eq*:
r01-binary-expansion′ r1 = *r01-binary-expansion′ r2* ⟷
r01-binary-expression r1 = *r01-binary-expression r2*
⟨*proof*⟩

lemma *power2-e*:
⋀*e*::*real*. *0* < *e* ⟹ ∃ *n*::*nat*. *real-of-rat* (*1/2*)^*n* < *e*
⟨*proof*⟩

lemma *r01-binary-expression-converges-to-r*:
assumes *0* < *r*
and *r* < *1*
shows *LIMSEQ* (*r01-binary-expression r*) *r*

⟨*proof*⟩

lemma *r01-binary-expression-correct*:
assumes $0 < r$
and $r < 1$
shows $r = (\sum n.\ real\ (r01\text{-}binary\text{-}expansion'\ r\ n) * (1/2)\hat{}(Suc\ n))$
⟨*proof*⟩

$S0 \leq S1 \leq S2 \leq ....$

lemma *binary-sum-incseq*:
incseq (*r01-binary-sum a*)
⟨*proof*⟩

lemma *r01-eq-iff*:
assumes $0 < r1\ r1 < 1$
$0 < r2\ r2 < 1$
shows $r1 = r2 \longleftrightarrow r01\text{-}binary\text{-}expansion'\ r1 = r01\text{-}binary\text{-}expansion'\ r2$
⟨*proof*⟩

lemma *power-half-summable*:
$summable\ (\lambda n.\ ((1::real)\ /\ 2)\ \hat{}\ Suc\ n)$
⟨*proof*⟩

lemma *binary-expression-summable*:
assumes $\bigwedge n.\ a\ n \in \{0,1 :: nat\}$
shows $summable\ (\lambda n.\ real\ (a\ n) * (1/2)\hat{}(Suc\ n))$
⟨*proof*⟩

lemma *binary-expression-gteq0*:
assumes $\bigwedge n.\ a\ n \in \{0,1 :: nat\}$
shows $0 \leq (\sum n.\ real\ (a\ (n + k)) * (1\ /\ 2)\ \hat{}\ Suc\ (n + k))$
⟨*proof*⟩

lemma *binary-expression-leeq1*:
assumes $\bigwedge n.\ a\ n \in \{0,1 :: nat\}$
shows $(\sum n.\ real\ (a\ (n + k)) * (1\ /\ 2)\ \hat{}\ Suc\ (n + k)) \leq 1$
⟨*proof*⟩

lemma *binary-expression-less-than*:
assumes $\bigwedge n.\ a\ n \in \{0,1 :: nat\}$
shows $(\sum n.\ real\ (a\ (n + k)) * (1\ /\ 2)\ \hat{}\ Suc\ (n + k)) \leq (\sum n.\ (1\ /\ 2)\ \hat{}\ Suc\ (n + k))$
⟨*proof*⟩

lemma *lim-sum-ai*:
assumes $\bigwedge n.\ a\ n \in \{0,1 :: nat\}$
shows $lim\ (\lambda n.\ (\sum i{=}0..n.\ real\ (a\ i) * (1/2)\hat{}(Suc\ i))) = (\sum n{::}nat.\ real\ (a\ n) * (1/2)\hat{}(Suc\ n))$

⟨*proof*⟩

lemma *half-1-minus-sum*:
$1 - (\sum i<k. ((1::real) / 2) \ \hat{} \ Suc \ i) = (1/2)\hat{}k$
⟨*proof*⟩

lemma *half-sum*:
$(\sum n. ((1::real) / 2) \ \hat{} (Suc \ (n + k))) = (1/2)\hat{}k$
⟨*proof*⟩

lemma *ai-exists0-less-than-sum*:
assumes $\bigwedge n. \ a \ n \in \{0,1\}$
$i \geq m$
and $a \ i = 0$
shows $(\sum n::nat. \ real \ (a \ (n + m)) * (1/2)\hat{}(Suc \ (n + m))) < (1 / 2) \ \hat{} \ m$
⟨*proof*⟩

lemma *ai-exists0-less-than1*:
assumes $\bigwedge n. \ a \ n \in \{0,1\}$
and $\exists i. \ a \ i = 0$
shows $(\sum n::nat. \ real \ (a \ n) * (1/2)\hat{}(Suc \ n)) < 1$
⟨*proof*⟩

lemma *ai-1-gt*:
assumes $\bigwedge n. \ a \ n \in \{0,1\}$
and $a \ i = 1$
shows $(1/2)\hat{}(Suc \ i) \leq (\sum n::nat. \ real \ (a \ (n+i)) * (1/2)\hat{}(Suc \ (n+i)))$
⟨*proof*⟩

lemma *ai-exists1-gt0*:
assumes $\bigwedge n. \ a \ n \in \{0,1\}$
and $\exists i. \ a \ i = 1$
shows $0 < (\sum n::nat. \ real \ (a \ n) * (1/2)\hat{}(Suc \ n))$
⟨*proof*⟩

lemma *r01-binary-expression-ex0*:
assumes $0 < r \ r < 1$
shows $\exists i. \ r01\text{-}binary\text{-}expansion' \ r \ i = 0$
⟨*proof*⟩

lemma *r01-binary-expression-ex1*:
assumes $0 < r \ r < 1$
shows $\exists i. \ r01\text{-}binary\text{-}expansion' \ r \ i = 1$
⟨*proof*⟩

lemma *r01-binary-expansion'-gt1*:
$1 \leq r \longleftrightarrow (\forall n. \ r01\text{-}binary\text{-}expansion' \ r \ n = 1)$
⟨*proof*⟩

lemma *r01-binary-expansion′-lt0*:
$r \le 0 \longleftrightarrow (\forall n.\ r01\text{-}binary\text{-}expansion'\ r\ n = 0)$
⟨*proof*⟩

The sequence 111111... does not appear in $r = 0.a_1a_2\ldots$.

lemma *r01-binary-expression-ex0-strong*:
assumes $0 < r\ r < 1$
shows $\exists\, i \ge n.\ r01\text{-}binary\text{-}expansion'\ r\ i = 0$
⟨*proof*⟩

A binary expression is well-formed when 111... does not appear in the tail of the sequence

definition *biexp01-well-formed* :: (*nat* ⇒ *nat*) ⇒ *bool* **where**
$biexp01\text{-}well\text{-}formed\ a \equiv (\forall n.\ a\ n \in \{0,1\}) \wedge (\forall n.\ \exists\, m \ge n.\ a\ m = 0)$

lemma *biexp01-well-formedE*:
assumes *biexp01-well-formed a*
shows $(\forall n.\ a\ n \in \{0,1\}) \wedge (\forall n.\ \exists\, m \ge n.\ a\ m = 0)$
⟨*proof*⟩

lemma *biexp01-well-formedI*:
assumes $\bigwedge n.\ a\ n \in \{0,1\}$
and $\bigwedge n.\ \exists\, m \ge n.\ a\ m = 0$
shows *biexp01-well-formed a*
⟨*proof*⟩

lemma *r01-binary-expansion-well-formed*:
assumes $0 < r\ r < 1$
shows *biexp01-well-formed* (*r01-binary-expansion′ r*)
⟨*proof*⟩

lemma *biexp01-well-formed-comb*:
assumes *biexp01-well-formed a*
and *biexp01-well-formed b*
shows *biexp01-well-formed* (λ*n. if even n then a* (*n div 2*)
else b ((*n*−*1*) *div 2*))
⟨*proof*⟩

lemma *nat-complete-induction*:
assumes $P\ (0 :: nat)$
and $\bigwedge n.\ (\bigwedge m.\ m \le n \Longrightarrow P\ m) \Longrightarrow P\ (Suc\ n)$
shows $P\ n$
⟨*proof*⟩

$(\sum m.\ real\ (a\ m) * (1\ /\ 2)\ \hat{}\ Suc\ m)\ n = a\ n.$

lemma *biexp01-well-formed-an*:

assumes *biexp01-well-formed a*
shows *r01-binary-expansion′* $(\sum m.\ real\ (a\ m) * (1\ /\ 2)\ \hat{}\ Suc\ m)\ n = a\ n$
⟨*proof*⟩

lemma *f01-borel-measurable*:
assumes $f\ -`\ \{0::real\} \in$ *sets real-borel*
$f\ -`\ \{1\} \in$ *sets borel*
and $\bigwedge r::real.\ f\ r \in \{0,1\}$
shows $f \in$ *borel-measurable real-borel*
⟨*proof*⟩

lemma *r01-binary-expansion′-measurable*:
($\lambda r.$ *real* (*r01-binary-expansion′ r n*)) $\in$ *borel-measurable* (*borel* :: *real measure*)
⟨*proof*⟩

definition *r01-to-r01-r01-fst′* :: *real* ⇒ *nat* ⇒ *nat* **where**
r01-to-r01-r01-fst′ r n ≡ *r01-binary-expansion′ r* $(2*n)$

lemma *r01-to-r01-r01-fst′in01*:
$\bigwedge n.$ *r01-to-r01-r01-fst′ r n* $\in \{0,1\}$
⟨*proof*⟩

definition *r01-to-r01-r01-fst-sum* :: *real* ⇒ *nat* ⇒ *real* **where**
r01-to-r01-r01-fst-sum ≡ *r01-binary-sum* ∘ *r01-to-r01-r01-fst′*

definition *r01-to-r01-r01-fst* :: *real* ⇒ *real* **where**
r01-to-r01-r01-fst = *lim* ∘ *r01-to-r01-r01-fst-sum*

lemma *r01-to-r01-r01-fst-def′*:
r01-to-r01-r01-fst r = $(\sum n.$ *real* (*r01-binary-expansion′ r* $(2*n)) * (1/2)\hat{}(n+1))$
⟨*proof*⟩

lemma *r01-to-r01-r01-fst-measurable*:
r01-to-r01-r01-fst $\in$ *borel-measurable borel*
⟨*proof*⟩

definition *r01-to-r01-r01-snd′* :: *real* ⇒ *nat* ⇒ *nat* **where**
r01-to-r01-r01-snd′ r n = *r01-binary-expansion′ r* $(2*n + 1)$

lemma *r01-to-r01-r01-snd′in01*:
$\bigwedge n.$ *r01-to-r01-r01-snd′ r n* $\in \{0,1\}$
⟨*proof*⟩

definition *r01-to-r01-r01-snd-sum* :: *real* ⇒ *nat* ⇒ *real* **where**
r01-to-r01-r01-snd-sum ≡ *r01-binary-sum* ∘ *r01-to-r01-r01-snd′*

definition *r01-to-r01-r01-snd* :: *real* ⇒ *real* **where**
r01-to-r01-r01-snd = *lim* ∘ *r01-to-r01-r01-snd-sum*

lemma *r01-to-r01-r01-snd-def′*:
r01-to-r01-r01-snd r = ($\sum$ *n. real* (*r01-binary-expansion′ r* (*2*∗*n* + *1*)) ∗ (*1*/*2*) ^(*n*+*1*))
⟨*proof*⟩

lemma *r01-to-r01-r01-snd-measurable*:
r01-to-r01-r01-snd ∈ *borel-measurable borel*
⟨*proof*⟩

definition *r01-to-r01-r01* :: *real* ⇒ *real* × *real* **where**
r01-to-r01-r01 r = (*r01-to-r01-r01-fst r*,*r01-to-r01-r01-snd r*)

lemma *r01-to-r01-r01-image*:
r01-to-r01-r01 r ∈ {*0*..*1*}×{*0*..*1*}
⟨*proof*⟩

lemma *r01-to-r01-r01-measurable*:
r01-to-r01-r01 ∈ *real-borel* $\rightarrow_M$ *real-borel* $\bigotimes_M$ *real-borel*
⟨*proof*⟩

lemma *r01-to-r01-r01-3over4*:
r01-to-r01-r01 (*3*/*4*) = (*1*/*2*,*1*/*2*)
⟨*proof*⟩

definition *r01-r01-to-r01′* :: *real* × *real* ⇒ *nat* ⇒ *nat* **where**
r01-r01-to-r01′ rs ≡ (λ*n. if even n then r01-binary-expansion′* (*fst rs*) (*n div 2*)
else r01-binary-expansion′ (*snd rs*) ((*n*−*1*) *div 2*))

lemma *r01-r01-to-r01′in01*:
⋀*n. r01-r01-to-r01′ rs n* ∈ {*0*,*1*}
⟨*proof*⟩

lemma *r01-r01-to-r01′-well-formed*:
assumes *0* < *r1 r1* < *1*
and *0* < *r2 r2* < *1*
shows *biexp01-well-formed* (*r01-r01-to-r01′* (*r1*,*r2*))
⟨*proof*⟩

definition *r01-r01-to-r01-sum* :: *real* × *real* ⇒ *nat* ⇒ *real* **where**
r01-r01-to-r01-sum ≡ *r01-binary-sum* ∘ *r01-r01-to-r01′*

definition *r01-r01-to-r01* :: *real* × *real* ⇒ *real* **where**
r01-r01-to-r01 ≡ *lim* ∘ *r01-r01-to-r01-sum*

lemma *r01-r01-to-r01-def′*:
r01-r01-to-r01 (*r1*,*r2*) = (∑ *n*. *real* (*r01-r01-to-r01′* (*r1*,*r2*) *n*) ∗ (*1*/*2*)^(*n*+*1*))
⟨*proof*⟩

lemma *r01-r01-to-r01-measurable*:
r01-r01-to-r01 ∈ *real-borel* $\bigotimes_M$ *real-borel* $\rightarrow_M$ *real-borel*
⟨*proof*⟩

lemma *r01-r01-to-r01-image*:
assumes *0* < *r1* *r1* < *1*
shows *r01-r01-to-r01* (*r1*,*r2*) ∈ {*0*<..<*1*}
⟨*proof*⟩

lemma *r01-r01-to-r01-image′*:
assumes *0* < *r2* *r2* < *1*
shows *r01-r01-to-r01* (*r1*,*r2*) ∈ {*0*<..<*1*}
⟨*proof*⟩

lemma *r01-r01-to-r01-binary-nth*:
assumes *0* < *r1* *r1* < *1*
and *0* < *r2* *r2* < *1*
shows *r01-binary-expansion′* *r1* *n* = *r01-binary-expansion′* (*r01-r01-to-r01* (*r1*,*r2*)) (*2*∗*n*) ∧
r01-binary-expansion′ *r2* *n* = *r01-binary-expansion′* (*r01-r01-to-r01* (*r1*,*r2*)) (*2*∗*n* + *1*)
⟨*proof*⟩

lemma *r01-r01--r01--r01-r01-id*:
assumes *0* < *r1* *r1* < *1*
0 < *r2* *r2* < *1*
shows (*r01-to-r01-r01* ∘ *r01-r01-to-r01*) (*r1*,*r2*) = (*r1*,*r2*)
⟨*proof*⟩

We first show that $M \bigotimes_M N$ is a standard Borel space for standard Borel spaces M and N.

lemma *pair-measurable*[*measurable*]:
assumes *f* ∈ *X* $\rightarrow_M$ *Y*
and *g* ∈ *X′* $\rightarrow_M$ *Y′*
shows *map-prod f g* ∈ *X* $\bigotimes_M$ *X′* $\rightarrow_M$ *Y* $\bigotimes_M$ *Y′*
⟨*proof*⟩

lemma *pair-standard-borel-standard*:
assumes *standard-borel M*
and *standard-borel N*

shows *standard-borel* $(M \bigotimes_M N)$
⟨*proof*⟩

lemma *pair-standard-borel-spaceUNIV*:
assumes *standard-borel-space-UNIV M*
and *standard-borel-space-UNIV N*
shows *standard-borel-space-UNIV* $(M \bigotimes_M N)$
⟨*proof*⟩

locale *pair-standard-borel* = *s1*: *standard-borel M* + *s2*: *standard-borel N*
for *M* :: *'a measure* **and** *N* :: *'b measure*
begin

sublocale *standard-borel* $M \bigotimes_M N$
⟨*proof*⟩

end

locale *pair-standard-borel-space-UNIV* = *s1*: *standard-borel-space-UNIV M* + *s2*: *standard-borel-space-UNIV N*
for *M* :: *'a measure* **and** *N* :: *'b measure*
begin

sublocale *pair-standard-borel M N*
⟨*proof*⟩

sublocale *standard-borel-space-UNIV* $M \bigotimes_M N$
⟨*proof*⟩

end

$\mathbb{R} \times \mathbb{R}$ is a standard Borel space.

interpretation *real-real* : *pair-standard-borel-space-UNIV real-borel real-borel*
⟨*proof*⟩

1.4 $\mathbb{N} \times \mathbb{R}$

$\mathbb{N} \times \mathbb{R}$ is a standard Borel space.

interpretation *nat-real*: *pair-standard-borel-space-UNIV nat-borel real-borel*
⟨*proof*⟩

end

2 Quasi-Borel Spaces

theory *QuasiBorel*
imports *StandardBorel*

begin

2.1 Definitions

We formalize quasi-Borel spaces introduced by Heunen et al. [1].

2.1.1 Quasi-Borel Spaces

definition *qbs-closed1* :: (*real* ⇒ ′*a*) *set* ⇒ *bool*
where *qbs-closed1 Mx* ≡ (∀ *a* ∈ *Mx*. ∀ *f* ∈ *real-borel* $\to_M$ *real-borel*. *a* ∘ *f* ∈ *Mx*)

definition *qbs-closed2* :: [′*a set*, (*real* ⇒ ′*a*) *set*] ⇒ *bool*
where *qbs-closed2 X Mx* ≡ (∀ *x* ∈ *X*. (λ*r*. *x*) ∈ *Mx*)

definition *qbs-closed3* :: (*real* ⇒ ′*a*) *set* ⇒ *bool*
where *qbs-closed3 Mx* ≡ (∀ *P*::*real* ⇒ *nat*. ∀ *Fi*::*nat* ⇒ *real* ⇒ ′*a*.
(∀ *i*. *P* −‘ {*i*} ∈ *sets real-borel*)
⟶ (∀ *i*. *Fi i* ∈ *Mx*)
⟶ (λ*r*. *Fi* (*P r*) *r*) ∈ *Mx*)

lemma *separate-measurable*:
fixes *P* :: *real* ⇒ *nat*
assumes ⋀*i*. *P* −‘ {*i*} ∈ *sets real-borel*
shows *P* ∈ *real-borel* $\to_M$ *nat-borel*
⟨*proof*⟩

lemma *measurable-separate*:
fixes *P* :: *real* ⇒ *nat*
assumes *P* ∈ *real-borel* $\to_M$ *nat-borel*
shows *P* −‘ {*i*} ∈ *sets real-borel*
⟨*proof*⟩

definition *is-quasi-borel X Mx* ⟷ *Mx* ⊆ *UNIV* → *X* ∧ *qbs-closed1 Mx* ∧ *qbs-closed2 X Mx* ∧ *qbs-closed3 Mx*

lemma *is-quasi-borel-intro*[*simp*]:
assumes *Mx* ⊆ *UNIV* → *X*
and *qbs-closed1 Mx qbs-closed2 X Mx qbs-closed3 Mx*
shows *is-quasi-borel X Mx*
⟨*proof*⟩

typedef ′*a quasi-borel* = {(*X*::′*a set*, *Mx*). *is-quasi-borel X Mx*}
⟨*proof*⟩

definition *qbs-space* :: ′*a quasi-borel* ⇒ ′*a set* **where**
qbs-space X ≡ *fst* (*Rep-quasi-borel X*)

definition *qbs-Mx* :: ′*a quasi-borel* ⇒ (*real* ⇒ ′*a*) *set* **where**
qbs-Mx X ≡ *snd* (*Rep-quasi-borel X*)

lemma *qbs-decomp* :
$(qbs\text{-}space\ X, qbs\text{-}Mx\ X) \in \{(X{::}'a\ set,\ Mx).\ is\text{-}quasi\text{-}borel\ X\ Mx\}$
⟨*proof*⟩

lemma *qbs-Mx-to-X*[*dest*]:
assumes $\alpha \in qbs\text{-}Mx\ X$
shows $\alpha \in UNIV \rightarrow qbs\text{-}space\ X$
$\alpha\ r \in qbs\text{-}space\ X$
⟨*proof*⟩

lemma *qbs-closed1I*:
assumes $\bigwedge\alpha\ f.\ \alpha \in Mx \Longrightarrow f \in real\text{-}borel \rightarrow_M real\text{-}borel \Longrightarrow \alpha \circ f \in Mx$
shows *qbs-closed1 Mx*
⟨*proof*⟩

lemma *qbs-closed1-dest*[*simp*]:
assumes $\alpha \in qbs\text{-}Mx\ X$
and $f \in real\text{-}borel \rightarrow_M real\text{-}borel$
shows $\alpha \circ f \in qbs\text{-}Mx\ X$
⟨*proof*⟩

lemma *qbs-closed2I*:
assumes $\bigwedge x.\ x \in X \Longrightarrow (\lambda r.\ x) \in Mx$
shows *qbs-closed2 X Mx*
⟨*proof*⟩

lemma *qbs-closed2-dest*[*simp*]:
assumes $x \in qbs\text{-}space\ X$
shows $(\lambda r.\ x) \in qbs\text{-}Mx\ X$
⟨*proof*⟩

lemma *qbs-closed3I*:
assumes $\bigwedge(P :: real \Rightarrow nat)\ Fi.\ (\bigwedge i.\ P\ -`\ \{i\} \in sets\ real\text{-}borel) \Longrightarrow (\bigwedge i.\ Fi\ i \in Mx)$
$\Longrightarrow (\lambda r.\ Fi\ (P\ r)\ r) \in Mx$
shows *qbs-closed3 Mx*
⟨*proof*⟩

lemma *qbs-closed3I′*:
assumes $\bigwedge(P :: real \Rightarrow nat)\ Fi.\ P \in real\text{-}borel \rightarrow_M nat\text{-}borel \Longrightarrow (\bigwedge i.\ Fi\ i \in Mx)$
$\Longrightarrow (\lambda r.\ Fi\ (P\ r)\ r) \in Mx$
shows *qbs-closed3 Mx*
⟨*proof*⟩

lemma *qbs-closed3-dest*[*simp*]:
fixes $P{::}real \Rightarrow nat$ **and** $Fi :: nat \Rightarrow real \Rightarrow$ -

assumes $\bigwedge i.\ P -\text{`}\ \{i\} \in$ *sets real-borel*
and $\bigwedge i.\ Fi\ i \in$ *qbs-Mx X*
shows $(\lambda r.\ Fi\ (P\ r)\ r) \in$ *qbs-Mx X*
⟨*proof*⟩

lemma *qbs-closed3-dest′*:
fixes *P*::*real* ⇒ *nat* **and** *Fi* :: *nat* ⇒ *real* ⇒ -
assumes $P \in$ *real-borel* $\rightarrow_M$ *nat-borel*
and $\bigwedge i.\ Fi\ i \in$ *qbs-Mx X*
shows $(\lambda r.\ Fi\ (P\ r)\ r) \in$ *qbs-Mx X*
⟨*proof*⟩

lemma *qbs-closed3-dest2*:
assumes *countable I*
and [*measurable*]: $P \in$ *real-borel* $\rightarrow_M$ *count-space I*
and $\bigwedge i.\ i \in I \Longrightarrow Fi\ i \in$ *qbs-Mx X*
shows $(\lambda r.\ Fi\ (P\ r)\ r) \in$ *qbs-Mx X*
⟨*proof*⟩

lemma *qbs-closed3-dest2′*:
assumes *countable I*
and [*measurable*]: $P \in$ *real-borel* $\rightarrow_M$ *count-space I*
and $\bigwedge i.\ i \in$ *range P* $\Longrightarrow Fi\ i \in$ *qbs-Mx X*
shows $(\lambda r.\ Fi\ (P\ r)\ r) \in$ *qbs-Mx X*
⟨*proof*⟩

lemma *qbs-space-Mx*:
qbs-space X = $\{\alpha\ x\ |x\ \alpha.\ \alpha \in$ *qbs-Mx X*$\}$
⟨*proof*⟩

lemma *qbs-space-eq-Mx*:
assumes *qbs-Mx X* = *qbs-Mx Y*
shows *qbs-space X* = *qbs-space Y*
⟨*proof*⟩

lemma *qbs-eqI*:
assumes *qbs-Mx X* = *qbs-Mx Y*
shows *X* = *Y*
⟨*proof*⟩

2.1.2 Morphism of Quasi-Borel Spaces

definition *qbs-morphism* :: [*′a quasi-borel*, *′b quasi-borel*] ⇒ (*′a* ⇒ *′b*) *set* (**infixr** ‹$\rightarrow_Q$› *60*) **where**
$X \rightarrow_Q Y \equiv \{f \in$ *qbs-space X* → *qbs-space Y*. $\forall\alpha \in$ *qbs-Mx X*. $f \circ \alpha \in$ *qbs-Mx Y*$\}$

lemma *qbs-morphismI*:
assumes $\bigwedge\alpha.\ \alpha \in$ *qbs-Mx X* $\Longrightarrow f \circ \alpha \in$ *qbs-Mx Y*

shows $f \in X \to_Q Y$
⟨*proof*⟩

lemma *qbs-morphismE*[*dest*]:
assumes $f \in X \to_Q Y$
shows $f \in$ *qbs-space* $X \to$ *qbs-space* Y
$\bigwedge x.\ x \in$ *qbs-space* $X \Longrightarrow f\ x \in$ *qbs-space* Y
$\bigwedge \alpha.\ \alpha \in$ *qbs-Mx* $X \Longrightarrow f \circ \alpha \in$ *qbs-Mx* Y
⟨*proof*⟩

lemma *qbs-morphism-ident*[*simp*]:
id $\in X \to_Q X$
⟨*proof*⟩

lemma *qbs-morphism-ident'*[*simp*]:
$(\lambda x.\ x) \in X \to_Q X$
⟨*proof*⟩

lemma *qbs-morphism-comp*:
assumes $f \in X \to_Q Y$ $g \in Y \to_Q Z$
shows $g \circ f \in X \to_Q Z$
⟨*proof*⟩

lemma *qbs-morphism-cong*:
assumes $\bigwedge x.\ x \in$ *qbs-space* $X \Longrightarrow f\ x = g\ x$
and $f \in X \to_Q Y$
shows $g \in X \to_Q Y$
⟨*proof*⟩

lemma *qbs-morphism-const*:
assumes $y \in$ *qbs-space* Y
shows $(\lambda\text{-}.\ y) \in X \to_Q Y$
⟨*proof*⟩

2.1.3 Empty Space

definition *empty-quasi-borel* :: *'a quasi-borel* **where**
empty-quasi-borel ≡ *Abs-quasi-borel* ({},{})

lemma *eqb-correct*: *Rep-quasi-borel empty-quasi-borel* = ({}, {})
⟨*proof*⟩

lemma *eqb-space*[*simp*]: *qbs-space empty-quasi-borel* = {}
⟨*proof*⟩

lemma *eqb-Mx*[*simp*]: *qbs-Mx empty-quasi-borel* = {}
⟨*proof*⟩

lemma *qbs-empty-equiv* :*qbs-space* $X = \{\} \longleftrightarrow$ *qbs-Mx* $X = \{\}$
⟨*proof*⟩

lemma *empty-quasi-borel-iff*:
qbs-space $X = \{\} \longleftrightarrow X =$ *empty-quasi-borel*
⟨*proof*⟩

2.1.4 Unit Space

definition *unit-quasi-borel* :: *unit quasi-borel* (‹1_Q›) **where**
unit-quasi-borel ≡ *Abs-quasi-borel* (*UNIV*,*UNIV*)

lemma *uqb-correct*: *Rep-quasi-borel unit-quasi-borel* = (*UNIV*,*UNIV*)
⟨*proof*⟩

lemma *uqb-space*[*simp*]: *qbs-space unit-quasi-borel* = {()}
⟨*proof*⟩

lemma *uqb-Mx*[*simp*]: *qbs-Mx unit-quasi-borel* = $\{\lambda r.\ ()\}$
⟨*proof*⟩

lemma *unit-quasi-borel-terminal*:
$\exists!\ f.\ f \in X \to_Q$ *unit-quasi-borel*
⟨*proof*⟩

definition *to-unit-quasi-borel* :: *'a* ⇒ *unit* (‹$!_Q$›) **where**
to-unit-quasi-borel ≡ (λ-.())

lemma *to-unit-quasi-borel-morphism* :
$!_Q \in X \to_Q$ *unit-quasi-borel*
⟨*proof*⟩

2.1.5 Subspaces

definition *sub-qbs* :: [*'a quasi-borel*, *'a set*] ⇒ *'a quasi-borel* **where**
sub-qbs X U ≡ *Abs-quasi-borel* (*qbs-space X* ∩ *U*,{*f* ∈ *UNIV* → *qbs-space X* ∩ *U*. *f* ∈ *qbs-Mx X*})

lemma *sub-qbs-closed*:
qbs-closed1 {*f* ∈ *UNIV* → *qbs-space X* ∩ *U*. *f* ∈ *qbs-Mx X*}
qbs-closed2 (*qbs-space X* ∩ *U*) {*f* ∈ *UNIV* → *qbs-space X* ∩ *U*. *f* ∈ *qbs-Mx X*}
qbs-closed3 {*f* ∈ *UNIV* → *qbs-space X* ∩ *U*. *f* ∈ *qbs-Mx X*}
⟨*proof*⟩

lemma *sub-qbs-correct*[*simp*]: *Rep-quasi-borel* (*sub-qbs X U*) = (*qbs-space X* ∩ *U*,{*f* ∈ *UNIV* → *qbs-space X* ∩ *U*. *f* ∈ *qbs-Mx X*})
⟨*proof*⟩

lemma *sub-qbs-space*[*simp*]: *qbs-space* (*sub-qbs X U*) = *qbs-space X* ∩ *U*
⟨*proof*⟩

lemma *sub-qbs-Mx*[*simp*]: *qbs-Mx* (*sub-qbs X U*) = {*f* ∈ *UNIV* → *qbs-space X* ∩ *U*. *f* ∈ *qbs-Mx X*}
⟨*proof*⟩

lemma *sub-qbs*:
assumes *U* ⊆ *qbs-space X*
shows (*qbs-space* (*sub-qbs X U*), *qbs-Mx* (*sub-qbs X U*)) = (*U*, {*f* ∈ *UNIV* → *U*. *f* ∈ *qbs-Mx X*})
⟨*proof*⟩

2.1.6 Image Spaces

definition *map-qbs* :: [′*a* ⇒ ′*b*] ⇒ ′*a quasi-borel* ⇒ ′*b quasi-borel* **where**
map-qbs f X = *Abs-quasi-borel* (*f* ‘ (*qbs-space X*),{β. ∃α∈ *qbs-Mx X*. β = *f* ∘ α})

lemma *map-qbs-f*:
{β. ∃α∈ *qbs-Mx X*. β = *f* ∘ α} ⊆ *UNIV* → *f* ‘ (*qbs-space X*)
⟨*proof*⟩

lemma *map-qbs-closed1*:
qbs-closed1 {β. ∃α∈ *qbs-Mx X*. β = *f* ∘ α}
⟨*proof*⟩

lemma *map-qbs-closed2*:
qbs-closed2 (*f* ‘ (*qbs-space X*)) {β. ∃α∈ *qbs-Mx X*. β = *f* ∘ α}
⟨*proof*⟩

lemma *map-qbs-closed3*:
qbs-closed3 {β. ∃α∈ *qbs-Mx X*. β = *f* ∘ α}
⟨*proof*⟩

lemma *map-qbs-correct*[*simp*]:
Rep-quasi-borel (*map-qbs f X*) = (*f* ‘ (*qbs-space X*),{β. ∃α∈ *qbs-Mx X*. β = *f* ∘ α})
⟨*proof*⟩

lemma *map-qbs-space*[*simp*]:
qbs-space (*map-qbs f X*) = *f* ‘ (*qbs-space X*)
⟨*proof*⟩

lemma *map-qbs-Mx*[*simp*]:
qbs-Mx (*map-qbs f X*) = {β. ∃α∈ *qbs-Mx X*. β = *f* ∘ α}
⟨*proof*⟩

inductive-set *generating-Mx* :: ′*a set* ⇒ (*real* ⇒ ′*a*) *set* ⇒ (*real* ⇒ ′*a*) *set*
for *X* :: ′*a set* **and** *Mx* :: (*real* ⇒ ′*a*) *set*
where

Basic: $\alpha \in Mx \Longrightarrow \alpha \in$ *generating-Mx X Mx*
| *Const*: $x \in X \Longrightarrow (\lambda r.\ x) \in$ *generating-Mx X Mx*
| *Comp* : $f \in$ *real-borel* $\rightarrow_M$ *real-borel* $\Longrightarrow \alpha \in$ *generating-Mx X Mx* $\Longrightarrow \alpha \circ f \in$ *generating-Mx X Mx*
| *Part* : $(\bigwedge i.\ Fi\ i \in$ *generating-Mx X Mx*$) \Longrightarrow P \in$ *real-borel* $\rightarrow_M$ *nat-borel* $\Longrightarrow (\lambda r.\ Fi\ (P\ r)\ r) \in$ *generating-Mx X Mx*

lemma *generating-Mx-to-space*:
assumes $Mx \subseteq UNIV \rightarrow X$
shows *generating-Mx X Mx* $\subseteq UNIV \rightarrow X$
⟨*proof*⟩

lemma *generating-Mx-closed1*:
qbs-closed1 (*generating-Mx X Mx*)
⟨*proof*⟩

lemma *generating-Mx-closed2*:
qbs-closed2 X (*generating-Mx X Mx*)
⟨*proof*⟩

lemma *generating-Mx-closed3*:
qbs-closed3 (*generating-Mx X Mx*)
⟨*proof*⟩

lemma *generating-Mx-Mx*:
generating-Mx (*qbs-space X*) (*qbs-Mx X*) = *qbs-Mx X*
⟨*proof*⟩

2.1.7 Ordering of Quasi-Borel Spaces

instantiation *quasi-borel* :: (*type*) *order-bot*
begin

inductive *less-eq-quasi-borel* :: *'a quasi-borel* ⇒ *'a quasi-borel* ⇒ *bool* **where**
qbs-space X ⊂ *qbs-space Y* ⟹ *less-eq-quasi-borel X Y*
| *qbs-space X* = *qbs-space Y* ⟹ *qbs-Mx Y* ⊆ *qbs-Mx X* ⟹ *less-eq-quasi-borel X Y*

lemma *le-quasi-borel-iff*:
$X \le Y \longleftrightarrow$ (*if qbs-space X* = *qbs-space Y then qbs-Mx Y* ⊆ *qbs-Mx X else qbs-space X* ⊂ *qbs-space Y*)
⟨*proof*⟩

definition *less-quasi-borel* :: *'a quasi-borel* ⇒ *'a quasi-borel* ⇒ *bool* **where**
less-quasi-borel X Y $\longleftrightarrow (X \le Y \wedge \neg\ Y \le X)$

definition *bot-quasi-borel* :: *'a quasi-borel* **where**
bot-quasi-borel = *empty-quasi-borel*

instance
⟨*proof*⟩
end

definition *inf-quasi-borel* :: [*'a quasi-borel*, *'a quasi-borel*] ⇒ *'a quasi-borel* **where**
inf-quasi-borel X X' = Abs-quasi-borel (*qbs-space X* ∩ *qbs-space X'*, *qbs-Mx X* ∩ *qbs-Mx X'*)

lemma *inf-quasi-borel-correct*: *Rep-quasi-borel* (*inf-quasi-borel X X'*) = (*qbs-space X* ∩ *qbs-space X'*, *qbs-Mx X* ∩ *qbs-Mx X'*)
⟨*proof*⟩

lemma *inf-qbs-space*[*simp*]: *qbs-space* (*inf-quasi-borel X X'*) = *qbs-space X* ∩ *qbs-space X'*
⟨*proof*⟩

lemma *inf-qbs-Mx*[*simp*]: *qbs-Mx* (*inf-quasi-borel X X'*) = *qbs-Mx X* ∩ *qbs-Mx X'*
⟨*proof*⟩

definition *max-quasi-borel* :: *'a set* ⇒ *'a quasi-borel* **where**
max-quasi-borel X = Abs-quasi-borel (*X*, *UNIV* → *X*)

lemma *max-quasi-borel-correct*: *Rep-quasi-borel* (*max-quasi-borel X*) = (*X*, *UNIV* → *X*)
⟨*proof*⟩

lemma *max-qbs-space*[*simp*]: *qbs-space* (*max-quasi-borel X*) = *X*
⟨*proof*⟩

lemma *max-qbs-Mx*[*simp*]: *qbs-Mx* (*max-quasi-borel X*) = *UNIV* → *X*
⟨*proof*⟩

instantiation *quasi-borel* :: (*type*) *semilattice-sup*
begin

definition *sup-quasi-borel* :: *'a quasi-borel* ⇒ *'a quasi-borel* ⇒ *'a quasi-borel* **where**
sup-quasi-borel X Y ≡ (*if qbs-space X = qbs-space Y then inf-quasi-borel X Y*
else if qbs-space X ⊂ *qbs-space Y then Y*
else if qbs-space Y ⊂ *qbs-space X then X*
else max-quasi-borel (*qbs-space X* ∪ *qbs-space Y*))

instance
⟨*proof*⟩
end

end

2.2 Relation to Measurable Spaces

theory *Measure-QuasiBorel-Adjunction*
imports *QuasiBorel*
begin

We construct the adjunction between **Meas** and **QBS**, where **Meas** is the category of measurable spaces and measurable functions and **QBS** is the category of quasi-Borel spaces and morphisms.

2.2.1 The Functor R

definition *measure-to-qbs* :: *'a measure* ⇒ *'a quasi-borel* **where**
measure-to-qbs X ≡ *Abs-quasi-borel* (*space X*, *real-borel* $\to_M$ *X*)

lemma *R-Mx-correct*: *real-borel* $\to_M$ *X* ⊆ *UNIV* → *space X*
⟨*proof*⟩

lemma *R-qbs-closed1*: *qbs-closed1* (*real-borel* $\to_M$ *X*)
⟨*proof*⟩

lemma *R-qbs-closed2*: *qbs-closed2* (*space X*) (*real-borel* $\to_M$ *X*)
⟨*proof*⟩

lemma *R-qbs-closed3*: *qbs-closed3* (*real-borel* $\to_M$ (*X* :: *'a measure*))
⟨*proof*⟩

lemma *R-correct*[*simp*]: *Rep-quasi-borel* (*measure-to-qbs X*) = (*space X*, *real-borel* $\to_M$ *X*)
⟨*proof*⟩

lemma *qbs-space-R*[*simp*]: *qbs-space* (*measure-to-qbs X*) = *space X*
⟨*proof*⟩

lemma *qbs-Mx-R*[*simp*]: *qbs-Mx* (*measure-to-qbs X*) = *real-borel* $\to_M$ *X*
⟨*proof*⟩

The following lemma says that *measure-to-qbs* is a functor from **Meas** to **QBS**.

lemma *r-preserves-morphisms*:
X $\to_M$ *Y* ⊆ (*measure-to-qbs X*) $\to_Q$ (*measure-to-qbs Y*)
⟨*proof*⟩

2.2.2 The Functor L

definition *sigma-Mx* :: *'a quasi-borel* ⇒ *'a set set* **where**
sigma-Mx X ≡ {*U* ∩ *qbs-space X* |*U*. ∀ α∈*qbs-Mx X*. α −‘ *U* ∈ *sets real-borel*}

definition *qbs-to-measure* :: *'a quasi-borel* ⇒ *'a measure* **where**

qbs-to-measure X ≡ *Abs-measure* (*qbs-space X*, *sigma-Mx X*, λA. (*if* $A = \{\}$ *then 0 else if* $A \in -$ *sigma-Mx X then 0 else* ∞))

lemma *measure-space-L*: *measure-space* (*qbs-space X*) (*sigma-Mx X*) (λA. (*if* $A = \{\}$ *then 0 else if* $A \in -$ *sigma-Mx X then 0 else* ∞))
⟨*proof*⟩

lemma *L-correct*[*simp*]:*Rep-measure* (*qbs-to-measure X*) = (*qbs-space X*, *sigma-Mx X*, λA. (*if* $A = \{\}$ *then 0 else if* $A \in -$ *sigma-Mx X then 0 else* ∞))
⟨*proof*⟩

lemma *space-L*[*simp*]: *space* (*qbs-to-measure X*) = *qbs-space X*
⟨*proof*⟩

lemma *sets-L*[*simp*]: *sets* (*qbs-to-measure X*) = *sigma-Mx X*
⟨*proof*⟩

lemma *emeasure-L*[*simp*]: *emeasure* (*qbs-to-measure X*) = (λA. *if* $A = \{\} \vee A \notin$ *sigma-Mx X then 0 else* ∞)
⟨*proof*⟩

lemma *qbs-Mx-sigma-Mx-contra*:
assumes *qbs-space X* = *qbs-space Y*
and *qbs-Mx X* ⊆ *qbs-Mx Y*
shows *sigma-Mx Y* ⊆ *sigma-Mx X*
⟨*proof*⟩

The following lemma says that *qbs-to-measure* is a functor from **QBS** to **Meas**.

lemma *l-preserves-morphisms*:
$X \to_Q Y \subseteq$ (*qbs-to-measure X*) $\to_M$ (*qbs-to-measure Y*)
⟨*proof*⟩

abbreviation *qbs-borel* ≡ *measure-to-qbs borel*

declare [[*coercion measure-to-qbs*]]

abbreviation *real-quasi-borel* :: *real quasi-borel* (‹$\mathbb{R}_Q$›) **where**
real-quasi-borel ≡ *qbs-borel*
abbreviation *nat-quasi-borel* :: *nat quasi-borel* (‹$\mathbb{N}_Q$›) **where**
nat-quasi-borel ≡ *qbs-borel*
abbreviation *ennreal-quasi-borel* :: *ennreal quasi-borel* (‹$\mathbb{R}_{Q\geq 0}$›) **where**
ennreal-quasi-borel ≡ *qbs-borel*
abbreviation *bool-quasi-borel* :: *bool quasi-borel* (‹$\mathbb{B}_Q$›) **where**
bool-quasi-borel ≡ *qbs-borel*

lemma *qbs-Mx-is-morphisms*:
$qbs\text{-}Mx\ X = real\text{-}quasi\text{-}borel \rightarrow_Q X$
⟨*proof*⟩

lemma *qbs-Mx-subset-of-measurable*:
$qbs\text{-}Mx\ X \subseteq real\text{-}borel \rightarrow_M qbs\text{-}to\text{-}measure\ X$
⟨*proof*⟩

lemma *L-max-of-measurables*:
assumes $space\ M = qbs\text{-}space\ X$
and $qbs\text{-}Mx\ X \subseteq real\text{-}borel \rightarrow_M M$
shows $sets\ M \subseteq sets\ (qbs\text{-}to\text{-}measure\ X)$
⟨*proof*⟩

lemma *qbs-Mx-are-measurable*[*simp*,*measurable*]:
assumes $\alpha \in qbs\text{-}Mx\ X$
shows $\alpha \in real\text{-}borel \rightarrow_M qbs\text{-}to\text{-}measure\ X$
⟨*proof*⟩

lemma *measure-to-qbs-cong-sets*:
assumes $sets\ M = sets\ N$
shows $measure\text{-}to\text{-}qbs\ M = measure\text{-}to\text{-}qbs\ N$
⟨*proof*⟩

lemma *lr-sets*[*simp*,*measurable-cong*]:
$sets\ X \subseteq sets\ (qbs\text{-}to\text{-}measure\ (measure\text{-}to\text{-}qbs\ X))$
⟨*proof*⟩

lemma(**in** *standard-borel*) *standard-borel-lr-sets-ident*[*simp*, *measurable-cong*]:
$sets\ (qbs\text{-}to\text{-}measure\ (measure\text{-}to\text{-}qbs\ M)) = sets\ M$
⟨*proof*⟩

2.2.3 The Adjunction

lemma *lr-adjunction-correspondence* :
$X \rightarrow_Q (measure\text{-}to\text{-}qbs\ Y) = (qbs\text{-}to\text{-}measure\ X) \rightarrow_M Y$
⟨*proof*⟩

lemma(**in** *standard-borel*) *standard-borel-r-full-faithful*:
$M \rightarrow_M Y = measure\text{-}to\text{-}qbs\ M \rightarrow_Q measure\text{-}to\text{-}qbs\ Y$
⟨*proof*⟩

lemma *qbs-morphism-dest*[*dest*]:
assumes $f \in X \rightarrow_Q measure\text{-}to\text{-}qbs\ Y$
shows $f \in qbs\text{-}to\text{-}measure\ X \rightarrow_M Y$
⟨*proof*⟩

lemma(**in** *standard-borel*) *qbs-morphism-dest*[*dest*]:

assumes $k \in$ *measure-to-qbs* $M \to_Q$ *measure-to-qbs* Y
shows $k \in M \to_M Y$
⟨*proof*⟩

lemma *qbs-morphism-measurable-intro*[*intro*!]:
assumes $f \in$ *qbs-to-measure* $X \to_M Y$
shows $f \in X \to_Q$ *measure-to-qbs* Y
⟨*proof*⟩

lemma(**in** *standard-borel*) *qbs-morphism-measurable-intro*[*intro*!]:
assumes $k \in M \to_M Y$
shows $k \in$ *measure-to-qbs* $M \to_Q$ *measure-to-qbs* Y
⟨*proof*⟩

We can use the measurability prover when we reason about morphisms.

lemma
assumes $f \in \mathbb{R}_Q \to_Q \mathbb{R}_Q$
shows $(\lambda x.\ 2 * f\, x + (f\, x)\hat{}2) \in \mathbb{R}_Q \to_Q \mathbb{R}_Q$
⟨*proof*⟩

lemma
assumes $f \in X \to_Q \mathbb{R}_Q$
and $\alpha \in$ *qbs-Mx* X
shows $(\lambda x.\ 2 * f\ (\alpha\ x) + (f\ (\alpha\ x))\hat{}2) \in \mathbb{R}_Q \to_Q \mathbb{R}_Q$
⟨*proof*⟩

lemma *qbs-morphisn-from-countable*:
fixes X :: *'a quasi-borel*
assumes *countable* (*qbs-space* X)
qbs-Mx $X \subseteq$ *real-borel* $\to_M$ *count-space* (*qbs-space* X)
and $\bigwedge i.\ i \in$ *qbs-space* $X \Longrightarrow f\ i \in$ *qbs-space* Y
shows $f \in X \to_Q Y$
⟨*proof*⟩

corollary *nat-qbs-morphism*:
assumes $\bigwedge n.\ f\ n \in$ *qbs-space* Y
shows $f \in \mathbb{N}_Q \to_Q Y$
⟨*proof*⟩

corollary *bool-qbs-morphism*:
assumes $\bigwedge b.\ f\ b \in$ *qbs-space* Y
shows $f \in \mathbb{B}_Q \to_Q Y$
⟨*proof*⟩

2.2.4 The Adjunction w.r.t. Ordering

lemma *l-mono*:
mono qbs-to-measure

⟨*proof*⟩

lemma *r-mono*:
mono measure-to-qbs
⟨*proof*⟩

lemma *rl-order-adjunction*:
$X \leq$ *qbs-to-measure* $Y \longleftrightarrow$ *measure-to-qbs* $X \leq Y$
⟨*proof*⟩

end

2.3 Product Spaces

theory *Binary-Product-QuasiBorel*
imports *Measure-QuasiBorel-Adjunction*
begin

2.3.1 Binary Product Spaces

definition *pair-qbs-Mx* :: [*'a quasi-borel*, *'b quasi-borel*] ⇒ (*real* => *'a* × *'b*) *set*
where
pair-qbs-Mx X Y ≡ {f. *fst* ∘ f ∈ *qbs-Mx* X ∧ *snd* ∘ f ∈ *qbs-Mx* Y}

definition *pair-qbs* :: [*'a quasi-borel*, *'b quasi-borel*] ⇒ (*'a* × *'b*) *quasi-borel* (**infixr** ‹$\bigotimes_Q$› *80*) **where**
pair-qbs X Y = *Abs-quasi-borel* (*qbs-space* X × *qbs-space* Y, *pair-qbs-Mx* X Y)

lemma *pair-qbs-f*[*simp*]: *pair-qbs-Mx* X Y ⊆ *UNIV* → *qbs-space* X × *qbs-space* Y
⟨*proof*⟩

lemma *pair-qbs-closed1*: *qbs-closed1* (*pair-qbs-Mx* (X::*'a quasi-borel*) (Y::*'b quasi-borel*))
⟨*proof*⟩

lemma *pair-qbs-closed2*: *qbs-closed2* (*qbs-space* X × *qbs-space* Y) (*pair-qbs-Mx* X Y)
⟨*proof*⟩

lemma *pair-qbs-closed3*: *qbs-closed3* (*pair-qbs-Mx* (X::*'a quasi-borel*) (Y::*'b quasi-borel*))
⟨*proof*⟩

lemma *pair-qbs-correct*: *Rep-quasi-borel* ($X \bigotimes_Q Y$) = (*qbs-space* X × *qbs-space* Y, *pair-qbs-Mx* X Y)
⟨*proof*⟩

lemma *pair-qbs-space*[*simp*]: *qbs-space* ($X \bigotimes_Q Y$) = *qbs-space* X × *qbs-space* Y
⟨*proof*⟩

lemma *pair-qbs-Mx*[*simp*]: *qbs-Mx* ($X \bigotimes_Q Y$) = *pair-qbs-Mx* X Y

⟨*proof*⟩

lemma *pair-qbs-morphismI*:
assumes $\bigwedge\alpha\ \beta.\ \alpha \in$ *qbs-Mx* $X \Longrightarrow \beta \in$ *qbs-Mx* Y
$\Longrightarrow f \circ (\lambda r.\ (\alpha\ r,\ \beta\ r)) \in$ *qbs-Mx* Z
shows $f \in (X \bigotimes_Q Y) \rightarrow_Q Z$
⟨*proof*⟩

lemma *fst-qbs-morphism*:
fst $\in X \bigotimes_Q Y \rightarrow_Q X$
⟨*proof*⟩

lemma *snd-qbs-morphism*:
snd $\in X \bigotimes_Q Y \rightarrow_Q Y$
⟨*proof*⟩

lemma *qbs-morphism-pair-iff*:
$f \in X \rightarrow_Q Y \bigotimes_Q Z \longleftrightarrow$ *fst* $\circ f \in X \rightarrow_Q Y \wedge$ *snd* $\circ f \in X \rightarrow_Q Z$
⟨*proof*⟩

lemma *qbs-morphism-Pair1*:
assumes $x \in$ *qbs-space* X
shows *Pair* $x \in Y \rightarrow_Q X \bigotimes_Q Y$
⟨*proof*⟩

lemma *qbs-morphism-Pair1'*:
assumes $x \in$ *qbs-space* X
and $f \in X \bigotimes_Q Y \rightarrow_Q Z$
shows $(\lambda y.\ f\ (x,y)) \in Y \rightarrow_Q Z$
⟨*proof*⟩

lemma *qbs-morphism-Pair2*:
assumes $y \in$ *qbs-space* Y
shows $(\lambda x.\ (x,y)) \in X \rightarrow_Q X \bigotimes_Q Y$
⟨*proof*⟩

lemma *qbs-morphism-Pair2'*:
assumes $y \in$ *qbs-space* Y
and $f \in X \bigotimes_Q Y \rightarrow_Q Z$
shows $(\lambda x.\ f\ (x,y)) \in X \rightarrow_Q Z$
⟨*proof*⟩

lemma *qbs-morphism-fst''*:
assumes $f \in X \rightarrow_Q Y$
shows $(\lambda k.\ f\ (fst\ k)) \in X \bigotimes_Q Z \rightarrow_Q Y$
⟨*proof*⟩

lemma *qbs-morphism-snd''*:
  assumes $f \in X \to_Q Y$
  shows $(\lambda k.\ f\ (snd\ k)) \in Z \bigotimes_Q X \to_Q Y$
  ⟨*proof*⟩

lemma *qbs-morphism-tuple*:
  assumes $f \in Z \to_Q X$
    and $g \in Z \to_Q Y$
   shows $(\lambda z.\ (f\ z,\ g\ z)) \in Z \to_Q X \bigotimes_Q Y$
⟨*proof*⟩

lemma *qbs-morphism-map-prod*:
  assumes $f \in X \to_Q Y$
    and $g \in X' \to_Q Y'$
   shows $map\text{-}prod\ f\ g \in X \bigotimes_Q X' \to_Q Y \bigotimes_Q Y'$
⟨*proof*⟩

lemma *qbs-morphism-pair-swap'*:
  $(\lambda(x,y).\ (y,x)) \in (X::'a\ quasi\text{-}borel) \bigotimes_Q (Y::'b\ quasi\text{-}borel) \to_Q Y \bigotimes_Q X$
  ⟨*proof*⟩

lemma *qbs-morphism-pair-swap*:
  assumes $f \in X \bigotimes_Q Y \to_Q Z$
  shows $(\lambda(x,y).\ f\ (y,x)) \in Y \bigotimes_Q X \to_Q Z$
⟨*proof*⟩

lemma *qbs-morphism-pair-assoc1*:
 $(\lambda((x,y),z).\ (x,(y,z))) \in (X \bigotimes_Q Y) \bigotimes_Q Z \to_Q X \bigotimes_Q (Y \bigotimes_Q Z)$
  ⟨*proof*⟩

lemma *qbs-morphism-pair-assoc2*:
 $(\lambda(x,(y,z)).\ ((x,y),z)) \in X \bigotimes_Q (Y \bigotimes_Q Z) \to_Q (X \bigotimes_Q Y) \bigotimes_Q Z$
  ⟨*proof*⟩

lemma *pair-qbs-fst*:
  assumes $qbs\text{-}space\ Y \neq \{\}$
  shows $map\text{-}qbs\ fst\ (X \bigotimes_Q Y) = X$
⟨*proof*⟩

lemma *pair-qbs-snd*:
  assumes $qbs\text{-}space\ X \neq \{\}$
  shows $map\text{-}qbs\ snd\ (X \bigotimes_Q Y) = Y$
⟨*proof*⟩

The following lemma corresponds to [1] Proposition 19(1).

lemma *r-preserves-product* :
  $measure\text{-}to\text{-}qbs\ (X \bigotimes_M Y) = measure\text{-}to\text{-}qbs\ X \bigotimes_Q measure\text{-}to\text{-}qbs\ Y$
  ⟨*proof*⟩

lemma *l-product-sets*[*simp*,*measurable-cong*]:
sets (*qbs-to-measure* $X \bigotimes_M$ *qbs-to-measure* Y) $\subseteq$ *sets* (*qbs-to-measure* ($X \bigotimes_Q$ Y))
⟨*proof*⟩

lemma(**in** *pair-standard-borel*) *l-r-r-sets*[*simp*,*measurable-cong*]:
sets (*qbs-to-measure* (*measure-to-qbs* $M \bigotimes_Q$ *measure-to-qbs* N)) = *sets* ($M \bigotimes_M$ N)
⟨*proof*⟩

end

2.3.2 Product Spaces

theory *Product-QuasiBorel*

imports *Binary-Product-QuasiBorel*

begin

definition *prod-qbs-Mx* :: [′*a set*, ′*a* ⇒ ′*b quasi-borel*] ⇒ (*real* ⇒ ′*a* ⇒ ′*b*) *set*
where
prod-qbs-Mx I M ≡ { α | α. ∀ *i*. (*i* ∈ *I* ⟶ (λ*r*. α *r i*) ∈ *qbs-Mx* (*M i*)) ∧ (*i* ∉ *I* ⟶ (λ*r*. α *r i*) = (λ*r*. *undefined*))}

lemma *prod-qbs-MxI*:
assumes ⋀*i*. *i* ∈ *I* ⟹ (λ*r*. α *r i*) ∈ *qbs-Mx* (*M i*)
and ⋀*i*. *i* ∉ *I* ⟹ (λ*r*. α *r i*) = (λ*r*. *undefined*)
shows α ∈ *prod-qbs-Mx I M*
⟨*proof*⟩

lemma *prod-qbs-MxE*:
assumes α ∈ *prod-qbs-Mx I M*
shows ⋀*i*. *i* ∈ *I* ⟹ (λ*r*. α *r i*) ∈ *qbs-Mx* (*M i*)
and ⋀*i*. *i* ∉ *I* ⟹ (λ*r*. α *r i*) = (λ*r*. *undefined*)
and ⋀*i r*. *i* ∉ *I* ⟹ α *r i* = *undefined*
⟨*proof*⟩

definition *PiQ* :: ′*a set* ⇒ (′*a* ⇒ ′*b quasi-borel*) ⇒ (′*a* ⇒ ′*b*) *quasi-borel* **where**
PiQ I M ≡ *Abs-quasi-borel* (Π_E *i*∈*I*. *qbs-space* (*M i*), *prod-qbs-Mx I M*)

syntax
-PiQ :: *pttrn* ⇒ ′*i set* ⇒ ′*a quasi-borel* ⇒ (′*i* => ′*a*) *quasi-borel* (‹(*3*Π_Q -∈-./ -)› *10*)
syntax-consts
-PiQ == *PiQ*
translations
Π_Q *x*∈*I*. *M* == *CONST PiQ I* (λ*x*. *M*)

lemma *PiQ-f*: $\textit{prod-qbs-Mx}\ I\ M \subseteq \textit{UNIV} \rightarrow (\Pi_E\ i{\in}I.\ \textit{qbs-space}\ (M\ i))$
⟨*proof*⟩

lemma *PiQ-closed1*: $\textit{qbs-closed1}\ (\textit{prod-qbs-Mx}\ I\ M)$
⟨*proof*⟩

lemma *PiQ-closed2*: $\textit{qbs-closed2}\ (\Pi_E\ i{\in}I.\ \textit{qbs-space}\ (M\ i))\ (\textit{prod-qbs-Mx}\ I\ M)$
⟨*proof*⟩

lemma *PiQ-closed3*: $\textit{qbs-closed3}\ (\textit{prod-qbs-Mx}\ I\ M)$
⟨*proof*⟩

lemma *PiQ-correct*: $\textit{Rep-quasi-borel}\ (\textit{PiQ}\ I\ M) = (\Pi_E\ i{\in}I.\ \textit{qbs-space}\ (M\ i), \textit{prod-qbs-Mx}\ I\ M)$
⟨*proof*⟩

lemma *PiQ-space*[*simp*]: $\textit{qbs-space}\ (\textit{PiQ}\ I\ M) = (\Pi_E\ i{\in}I.\ \textit{qbs-space}\ (M\ i))$
⟨*proof*⟩

lemma *PiQ-Mx*[*simp*]: $\textit{qbs-Mx}\ (\textit{PiQ}\ I\ M) = \textit{prod-qbs-Mx}\ I\ M$
⟨*proof*⟩

lemma *qbs-morphism-component-singleton*:
assumes $i \in I$
shows $(\lambda x.\ x\ i) \in (\Pi_Q\ i{\in}I.\ (M\ i)) \rightarrow_Q M\ i$
⟨*proof*⟩

lemma *product-qbs-canonical1*:
assumes $\bigwedge i.\ i \in I \Longrightarrow f\ i \in Y \rightarrow_Q X\ i$
and $\bigwedge i.\ i \notin I \Longrightarrow f\ i = (\lambda y.\ \textit{undefined})$
shows $(\lambda y\ i.\ f\ i\ y) \in Y \rightarrow_Q (\Pi_Q\ i{\in}I.\ X\ i)$
⟨*proof*⟩

lemma *product-qbs-canonical2*:
assumes $\bigwedge i.\ i \in I \Longrightarrow f\ i \in Y \rightarrow_Q X\ i$
$\bigwedge i.\ i \notin I \Longrightarrow f\ i = (\lambda y.\ \textit{undefined})$
$g \in Y \rightarrow_Q (\Pi_Q\ i{\in}I.\ X\ i)$
$\bigwedge i.\ i \in I \Longrightarrow f\ i = (\lambda x.\ x\ i) \circ g$
and $y \in \textit{qbs-space}\ Y$
shows $g\ y = (\lambda i.\ f\ i\ y)$
⟨*proof*⟩

lemma *merge-qbs-morphism*:
$\textit{merge}\ I\ J \in (\Pi_Q\ i{\in}I.\ (M\ i)) \bigotimes_Q (\Pi_Q\ j{\in}J.\ (M\ j)) \rightarrow_Q (\Pi_Q\ i{\in}I{\cup}J.\ (M\ i))$
⟨*proof*⟩

The following lemma corresponds to [1] Proposition 19(1).

lemma *r-preserves-product'*:
measure-to-qbs (Π_M *i*∈*I*. *M i*) = (Π_Q *i*∈*I*. *measure-to-qbs* (*M i*))
⟨*proof*⟩

$\prod_{i=0,1} X_i \cong X_1 \times X_2$.

lemma *product-binary-product*:
∃*f g*. *f* ∈ (Π_Q *i*∈*UNIV*. *if i then X else Y*) $\to_Q$ *X* $\bigotimes_Q$ *Y* ∧ *g* ∈ *X* $\bigotimes_Q$ *Y* $\to_Q$ (Π_Q *i*∈*UNIV*. *if i then X else Y*) ∧
g ∘ *f* = *id* ∧ *f* ∘ *g* = *id*
⟨*proof*⟩

end

2.4 Coproduct Spaces

theory *Binary-CoProduct-QuasiBorel*
imports *Measure-QuasiBorel-Adjunction*
begin

2.4.1 Binary Coproduct Spaces

definition *copair-qbs-Mx* :: ['*a quasi-borel*, '*b quasi-borel*] ⇒ (*real* => '*a* + '*b*) *set*
where
copair-qbs-Mx X Y ≡
{*g*. ∃ *S* ∈ *sets real-borel*.
(*S* = {} ⟶ (∃ *α1*∈ *qbs-Mx X*. *g* = (λ*r*. *Inl* (*α1 r*)))) ∧
(*S* = *UNIV* ⟶ (∃ *α2*∈ *qbs-Mx Y*. *g* = (λ*r*. *Inr* (*α2 r*)))) ∧
((*S* ≠ {} ∧ *S* ≠ *UNIV*) ⟶
(∃ *α1*∈ *qbs-Mx X*.
∃ *α2*∈ *qbs-Mx Y*.
g = (λ*r*::*real*. (*if* (*r* ∈ *S*) *then Inl* (*α1 r*) *else Inr* (*α2 r*)))))}

definition *copair-qbs* :: ['*a quasi-borel*, '*b quasi-borel*] ⇒ ('*a* + '*b*) *quasi-borel* (**infixr** ‹<+>$_Q$› *65*) **where**
copair-qbs X Y ≡ *Abs-quasi-borel* (*qbs-space X* <+> *qbs-space Y*, *copair-qbs-Mx X Y*)

The followin is an equivalent definition of *copair-qbs-Mx*.

definition *copair-qbs-Mx2* :: ['*a quasi-borel*, '*b quasi-borel*] ⇒ (*real* => '*a* + '*b*) *set* **where**
copair-qbs-Mx2 X Y ≡
{*g*. (*if qbs-space X* = {} ∧ *qbs-space Y* = {} *then False*
else if qbs-space X ≠ {} ∧ *qbs-space Y* = {} *then*
(∃*α1*∈ *qbs-Mx X*. *g* = (λ*r*. *Inl* (*α1 r*)))
else if qbs-space X = {} ∧ *qbs-space Y* ≠ {} *then*
(∃*α2*∈ *qbs-Mx Y*. *g* = (λ*r*. *Inr* (*α2 r*)))
else
(∃*S* ∈ *sets real-borel*. ∃*α1*∈ *qbs-Mx X*. ∃*α2*∈ *qbs-Mx Y*.

$g = (\lambda r{::}real.\ (if\ (r \in S)\ then\ Inl\ (\alpha 1\ r)\ else\ Inr\ (\alpha 2\ r))))) \}$

lemma *copair-qbs-Mx-equiv* :*copair-qbs-Mx* (*X* :: *'a quasi-borel*) (*Y* :: *'b quasi-borel*) = *copair-qbs-Mx2 X Y*
⟨*proof*⟩

lemma *copair-qbs-f*[*simp*]: *copair-qbs-Mx X Y* ⊆ *UNIV* → *qbs-space X* <+> *qbs-space Y*
⟨*proof*⟩

lemma *copair-qbs-closed1*: *qbs-closed1* (*copair-qbs-Mx X Y*)
⟨*proof*⟩

lemma *copair-qbs-closed2*: *qbs-closed2* (*qbs-space X* <+> *qbs-space Y*) (*copair-qbs-Mx X Y*)
⟨*proof*⟩

lemma *copair-qbs-closed3*: *qbs-closed3* (*copair-qbs-Mx X Y*)
⟨*proof*⟩

lemma *copair-qbs-correct*: *Rep-quasi-borel* (*copair-qbs X Y*) = (*qbs-space X* <+> *qbs-space Y*, *copair-qbs-Mx X Y*)
⟨*proof*⟩

lemma *copair-qbs-space*[*simp*]: *qbs-space* (*copair-qbs X Y*) = *qbs-space X* <+> *qbs-space Y*
⟨*proof*⟩

lemma *copair-qbs-Mx*[*simp*]: *qbs-Mx* (*copair-qbs X Y*) = *copair-qbs-Mx X Y*
⟨*proof*⟩

lemma *Inl-qbs-morphism*:
$Inl \in X \rightarrow_Q X <\!+\!>_Q Y$
⟨*proof*⟩

lemma *Inr-qbs-morphism*:
$Inr \in Y \rightarrow_Q X <\!+\!>_Q Y$
⟨*proof*⟩

lemma *case-sum-preserves-morphisms*:
assumes $f \in X \rightarrow_Q Z$
and $g \in Y \rightarrow_Q Z$
shows $case\text{-}sum\ f\ g \in X <\!+\!>_Q Y \rightarrow_Q Z$
⟨*proof*⟩

lemma *map-sum-preserves-morphisms*:

assumes $f \in X \rightarrow_Q Y$
and $g \in X' \rightarrow_Q Y'$
shows *map-sum f g* $\in X$ <+>$_Q$ $X' \rightarrow_Q Y$ <+>$_Q$ Y'
⟨*proof*⟩

end

2.4.2 Countable Coproduct Spaces

theory *CoProduct-QuasiBorel*

imports
Product-QuasiBorel
Binary-CoProduct-QuasiBorel
begin

definition *coprod-qbs-Mx* :: ['*a set*, '*a* ⇒ '*b quasi-borel*] ⇒ (*real* ⇒ '*a* × '*b*) *set*
where
coprod-qbs-Mx I X ≡ { λ*r*. (*f r*, α (*f r*) *r*) |*f* α. *f* ∈ *real-borel* $\rightarrow_M$ *count-space I* ∧ (∀ *i*∈*range f*. α *i* ∈ *qbs-Mx* (*X i*))}

lemma *coprod-qbs-MxI*:
assumes *f* ∈ *real-borel* $\rightarrow_M$ *count-space I*
and ⋀*i*. *i* ∈ *range f* ⟹ α *i* ∈ *qbs-Mx* (*X i*)
shows (λ*r*. (*f r*, α (*f r*) *r*)) ∈ *coprod-qbs-Mx I X*
⟨*proof*⟩

definition *coprod-qbs-Mx'* :: ['*a set*, '*a* ⇒ '*b quasi-borel*] ⇒ (*real* ⇒ '*a* × '*b*) *set*
where
coprod-qbs-Mx' I X ≡ { λ*r*. (*f r*, α (*f r*) *r*) |*f* α. *f* ∈ *real-borel* $\rightarrow_M$ *count-space I* ∧ (∀ *i*. (*i* ∈ *range f* ∨ *qbs-space* (*X i*) ≠ {}) ⟶ α *i* ∈ *qbs-Mx* (*X i*))}

lemma *coproduct-qbs-Mx-eq*:
coprod-qbs-Mx I X = *coprod-qbs-Mx' I X*
⟨*proof*⟩

definition *coprod-qbs* :: ['*a set*, '*a* ⇒ '*b quasi-borel*] ⇒ ('*a* × '*b*) *quasi-borel* **where**
coprod-qbs I X ≡ *Abs-quasi-borel* (*SIGMA i*:*I*. *qbs-space* (*X i*), *coprod-qbs-Mx I X*)

syntax
-coprod-qbs :: *pttrn* ⇒ '*i set* ⇒ '*a quasi-borel* ⇒ ('*i* × '*a*) *quasi-borel* (‹(*3*∐$_Q$ -∈-./ -)› *10*)
syntax-consts
-coprod-qbs ⇌ *coprod-qbs*
translations
∐$_Q$ *x*∈*I*. *M* ⇌ *CONST coprod-qbs I* (λ*x*. *M*)

lemma *coprod-qbs-f*[*simp*]: *coprod-qbs-Mx I X* ⊆ *UNIV* → (*SIGMA i*:*I*. *qbs-space*

(*X i*))
⟨*proof*⟩

lemma *coprod-qbs-closed1*: *qbs-closed1* (*coprod-qbs-Mx I X*)
⟨*proof*⟩

lemma *coprod-qbs-closed2*: *qbs-closed2* (*SIGMA i*:*I*. *qbs-space* (*X i*)) (*coprod-qbs-Mx I X*)
⟨*proof*⟩

lemma *coprod-qbs-closed3*:
qbs-closed3 (*coprod-qbs-Mx I X*)
⟨*proof*⟩

lemma *coprod-qbs-correct*: *Rep-quasi-borel* (*coprod-qbs I X*) = (*SIGMA i*:*I*. *qbs-space* (*X i*), *coprod-qbs-Mx I X*)
⟨*proof*⟩

lemma *coproduct-qbs-space*[*simp*]: *qbs-space* (*coprod-qbs I X*) = (*SIGMA i*:*I*. *qbs-space* (*X i*))
⟨*proof*⟩

lemma *coproduct-qbs-Mx*[*simp*]: *qbs-Mx* (*coprod-qbs I X*) = *coprod-qbs-Mx I X*
⟨*proof*⟩

lemma *ini-morphism*:
assumes $j \in I$
shows $(\lambda x.\ (j,x)) \in X\ j \rightarrow_Q (\amalg_Q\ i{\in}I.\ X\ i)$
⟨*proof*⟩

lemma *coprod-qbs-canonical1*:
assumes *countable I*
and $\bigwedge i.\ i \in I \Longrightarrow f\ i \in X\ i \rightarrow_Q Y$
shows $(\lambda(i,x).\ f\ i\ x) \in (\amalg_Q\ i \in I.\ X\ i) \rightarrow_Q Y$
⟨*proof*⟩

lemma *coprod-qbs-canonical1′*:
assumes *countable I*
and $\bigwedge i.\ i \in I \Longrightarrow (\lambda x.\ f\ (i,x)) \in X\ i \rightarrow_Q Y$
shows $f \in (\amalg_Q\ i \in I.\ X\ i) \rightarrow_Q Y$
⟨*proof*⟩

$\coprod_{i=0,1} X_i \cong X_1 + X_2$.

lemma *coproduct-binary-coproduct*:
$\exists f\ g.\ f \in (\amalg_Q\ i{\in}UNIV.\ \textit{if } i \textit{ then } X \textit{ else } Y) \rightarrow_Q X <\!+\!>_Q Y \wedge g \in X <\!+\!>_Q Y \rightarrow_Q (\amalg_Q\ i{\in}UNIV.\ \textit{if } i \textit{ then } X \textit{ else } Y) \wedge$
$g \circ f = id \wedge f \circ g = id$
⟨*proof*⟩

2.4.3 Lists

abbreviation *list-of* $X \equiv \Pi_Q\ n{\in}(UNIV :: nat\ set).\ (\Pi_Q\ i{\in}\{..{<}n\}.\ X)$
abbreviation *list-nil* :: $nat \times (nat \Rightarrow 'a)$ **where**
list-nil $\equiv (0,\ \lambda n.\ undefined)$
abbreviation *list-cons* :: $['a,\ nat \times (nat \Rightarrow 'a)] \Rightarrow nat \times (nat \Rightarrow 'a)$ **where**
list-cons $x\ l \equiv (Suc\ (fst\ l),\ (\lambda n.\ if\ n = 0\ then\ x\ else\ (snd\ l)\ (n - 1)))$

definition *list-head* :: $nat \times (nat \Rightarrow 'a) \Rightarrow 'a$ **where**
list-head $l = snd\ l\ 0$
definition *list-tail* :: $nat \times (nat \Rightarrow 'a) \Rightarrow nat \times (nat \Rightarrow 'a)$ **where**
list-tail $l = (fst\ l - 1,\ \lambda m.\ (snd\ l)\ (Suc\ m))$

lemma *list-simp1*:
list-nil $\neq$ *list-cons* $x\ l$
⟨*proof*⟩

lemma *list-simp2*:
assumes *list-cons* $a\ al$ = *list-cons* $b\ bl$
shows $a = b\ al = bl$
⟨*proof*⟩

lemma *list-simp3*:
shows *list-head* (*list-cons* $a\ l$) $= a$
⟨*proof*⟩

lemma *list-simp4*:
assumes $l \in$ *qbs-space* (*list-of* X)
shows *list-tail* (*list-cons* $a\ l$) $= l$
⟨*proof*⟩

lemma *list-decomp1*:
assumes $l \in$ *qbs-space* (*list-of* X)
shows l = *list-nil* $\vee$
$(\exists\ a\ l'.\ a \in$ *qbs-space* $X \wedge l' \in$ *qbs-space* (*list-of* X) $\wedge\ l$ = *list-cons* $a\ l')$
⟨*proof*⟩

lemma *list-simp5*:
assumes $l \in$ *qbs-space* (*list-of* X)
and $l \neq$ *list-nil*
shows l = *list-cons* (*list-head* l) (*list-tail* l)
⟨*proof*⟩

lemma *list-simp6*:
list-nil $\in$ *qbs-space* (*list-of* X)
⟨*proof*⟩

lemma *list-simp7*:
assumes $a \in$ *qbs-space* X

and $l \in$ *qbs-space* (*list-of* X)
shows *list-cons a* $l \in$ *qbs-space* (*list-of* X)
⟨*proof*⟩

lemma *list-destruct-rule*:
assumes $l \in$ *qbs-space* (*list-of* X)
P list-nil
and $\bigwedge a\ l'.\ a \in$ *qbs-space* $X \Longrightarrow l' \in$ *qbs-space* (*list-of* X) $\Longrightarrow P$ (*list-cons a* l')
shows $P\ l$
⟨*proof*⟩

lemma *list-induct-rule*:
assumes $l \in$ *qbs-space* (*list-of* X)
P list-nil
and $\bigwedge a\ l'.\ a \in$ *qbs-space* $X \Longrightarrow l' \in$ *qbs-space* (*list-of* X) $\Longrightarrow P\ l' \Longrightarrow P$ (*list-cons a* l')
shows $P\ l$
⟨*proof*⟩

fun *from-list* :: *'a list* ⇒ *nat* × (*nat* ⇒ *'a*) **where**
from-list [] = *list-nil* |
from-list (*a*#*l*) = *list-cons a* (*from-list l*)

fun *to-list'* :: *nat* ⇒ (*nat* ⇒ *'a*) ⇒ *'a list* **where**
to-list' 0 - = [] |
to-list' (*Suc n*) *f* = *f 0* # *to-list' n* (λ*n*. *f* (*Suc n*))

definition *to-list* :: *nat* × (*nat* ⇒ *'a*) ⇒ *'a list* **where**
to-list ≡ *case-prod to-list'*

lemma *to-list-simp1*:
shows *to-list list-nil* = []
⟨*proof*⟩

lemma *to-list-simp2*:
assumes $l \in$ *qbs-space* (*list-of* X)
shows *to-list* (*list-cons a l*) = *a* # *to-list l*
⟨*proof*⟩

lemma *from-list-length*:
fst (*from-list l*) = *length l*
⟨*proof*⟩

lemma *from-list-in-list-of*:
assumes *set* $l \subseteq$ *qbs-space* X
shows *from-list* $l \in$ *qbs-space* (*list-of* X)
⟨*proof*⟩

lemma *from-list-in-list-of'*:
shows *from-list* $l \in$ *qbs-space* (*list-of* (*Abs-quasi-borel* (*UNIV*,*UNIV*)))
⟨*proof*⟩

lemma *list-cons-in-list-of*:
assumes *set* ($a\#l$) $\subseteq$ *qbs-space X*
shows *list-cons a* (*from-list l*) $\in$ *qbs-space* (*list-of X*)
⟨*proof*⟩

lemma *from-list-to-list-ident*:
(*to-list* $\circ$ *from-list*) $l = l$
⟨*proof*⟩

lemma *to-list-from-list-ident*:
assumes $l \in$ *qbs-space* (*list-of X*)
shows (*from-list* $\circ$ *to-list*) $l = l$
⟨*proof*⟩

definition *rec-list'* :: $'b \Rightarrow ('a \Rightarrow (nat \times (nat \Rightarrow 'a)) \Rightarrow 'b \Rightarrow 'b) \Rightarrow (nat \times (nat \Rightarrow 'a)) \Rightarrow 'b$ **where**
rec-list' t0 f l $\equiv$ (*rec-list t0* ($\lambda x\ l'$. *f x* (*from-list l'*)) (*to-list l*))

lemma *rec-list'-simp1*:
rec-list' t f list-nil $= t$
⟨*proof*⟩

lemma *rec-list'-simp2*:
assumes $l \in$ *qbs-space* (*list-of X*)
shows *rec-list' t f* (*list-cons x l*) $=$ *f x l* (*rec-list' t f l*)
⟨*proof*⟩

end

2.5 Function Spaces

theory *Exponent-QuasiBorel*
imports *CoProduct-QuasiBorel*
begin

2.5.1 Function Spaces

definition *exp-qbs-Mx* :: ['a *quasi-borel*, 'b *quasi-borel*] $\Rightarrow$ (*real* $\Rightarrow$ 'a => 'b) *set* **where**
exp-qbs-Mx X Y $\equiv \{g :: real \Rightarrow 'a \Rightarrow 'b.$ *case-prod g* $\in \mathbb{R}_Q \bigotimes_Q X \to_Q Y\}$

definition *exp-qbs* :: ['a *quasi-borel*, 'b *quasi-borel*] $\Rightarrow$ ('a $\Rightarrow$ 'b) *quasi-borel* (**infixr** ‹$\Rightarrow_Q$› *61*) **where**
$X \Rightarrow_Q Y \equiv$ *Abs-quasi-borel* ($X \to_Q Y$, *exp-qbs-Mx X Y*)

lemma *exp-qbs-f*[*simp*]: *exp-qbs-Mx X Y* $\subseteq$ *UNIV* $\rightarrow$ (*X* :: *'a quasi-borel*) $\rightarrow_Q$ (*Y* :: *'b quasi-borel*)
⟨*proof*⟩

lemma *exp-qbs-closed1*: *qbs-closed1* (*exp-qbs-Mx X Y*)
⟨*proof*⟩

lemma *exp-qbs-closed2*: *qbs-closed2* (*X* $\rightarrow_Q$ *Y*) (*exp-qbs-Mx X Y*)
⟨*proof*⟩

lemma *exp-qbs-closed3*:*qbs-closed3* (*exp-qbs-Mx X Y*)
⟨*proof*⟩

lemma *exp-qbs-correct*: *Rep-quasi-borel* (*exp-qbs X Y*) = (*X* $\rightarrow_Q$ *Y*, *exp-qbs-Mx X Y*)
⟨*proof*⟩

lemma *exp-qbs-space*[*simp*]: *qbs-space* (*exp-qbs X Y*) = *X* $\rightarrow_Q$ *Y*
⟨*proof*⟩

lemma *exp-qbs-Mx*[*simp*]: *qbs-Mx* (*exp-qbs X Y*) = *exp-qbs-Mx X Y*
⟨*proof*⟩

lemma *qbs-exp-morphismI*:
assumes $\bigwedge\alpha\ \beta.\ \alpha \in$ *qbs-Mx X* $\Longrightarrow$
$\beta \in$ *pair-qbs-Mx real-quasi-borel Y* $\Longrightarrow$
$(\lambda(r,x).\ (f \circ \alpha)\ r\ x) \circ \beta \in$ *qbs-Mx Z*
shows *f* $\in$ *X* $\rightarrow_Q$ *exp-qbs Y Z*
⟨*proof*⟩

definition *qbs-eval* :: (('a $\Rightarrow$ 'b) $\times$ 'a) $\Rightarrow$ 'b **where**
qbs-eval a $\equiv$ (*fst a*) (*snd a*)

lemma *qbs-eval-morphism*:
qbs-eval $\in$ (*exp-qbs X Y*) $\bigotimes_Q$ *X* $\rightarrow_Q$ *Y*
⟨*proof*⟩

lemma *curry-morphism*:
curry $\in$ *exp-qbs* (*X* $\bigotimes_Q$ *Y*) *Z* $\rightarrow_Q$ *exp-qbs X* (*exp-qbs Y Z*)
⟨*proof*⟩

lemma *curry-preserves-morphisms*:
assumes *f* $\in$ *X* $\bigotimes_Q$ *Y* $\rightarrow_Q$ *Z*
shows *curry f* $\in$ *X* $\rightarrow_Q$ *exp-qbs Y Z*
⟨*proof*⟩

lemma *uncurry-morphism*:
$\textit{case-prod} \in \textit{exp-qbs}\ X\ (\textit{exp-qbs}\ Y\ Z) \to_Q \textit{exp-qbs}\ (X \bigotimes_Q Y)\ Z$
⟨*proof*⟩

lemma *uncurry-preserves-morphisms*:
assumes $f \in X \to_Q \textit{exp-qbs}\ Y\ Z$
shows $\textit{case-prod}\ f \in X \bigotimes_Q Y \to_Q Z$
⟨*proof*⟩

lemma *arg-swap-morphism*:
assumes $f \in X \to_Q \textit{exp-qbs}\ Y\ Z$
shows $(\lambda y\ x.\ f\ x\ y) \in Y \to_Q \textit{exp-qbs}\ X\ Z$
⟨*proof*⟩

lemma *exp-qbs-comp-morphism*:
assumes $f \in W \to_Q \textit{exp-qbs}\ X\ Y$
and $g \in W \to_Q \textit{exp-qbs}\ Y\ Z$
shows $(\lambda w.\ g\ w \circ f\ w) \in W \to_Q \textit{exp-qbs}\ X\ Z$
⟨*proof*⟩

lemma *case-sum-morphism*:
$\textit{case-prod}\ \textit{case-sum} \in \textit{exp-qbs}\ X\ Z \bigotimes_Q \textit{exp-qbs}\ Y\ Z \to_Q \textit{exp-qbs}\ (X <\!+\!>_Q Y)$
Z
⟨*proof*⟩

lemma *not-qbs-morphism*:
$\textit{Not} \in \mathbb{B}_Q \to_Q \mathbb{B}_Q$
⟨*proof*⟩

lemma *or-qbs-morphism*:
$(\vee) \in \mathbb{B}_Q \to_Q \textit{exp-qbs}\ \mathbb{B}_Q\ \mathbb{B}_Q$
⟨*proof*⟩

lemma *and-qbs-morphism*:
$(\wedge) \in \mathbb{B}_Q \to_Q \textit{exp-qbs}\ \mathbb{B}_Q\ \mathbb{B}_Q$
⟨*proof*⟩

lemma *implies-qbs-morphism*:
$(\longrightarrow) \in \mathbb{B}_Q \to_Q \mathbb{B}_Q \Rightarrow_Q \mathbb{B}_Q$
⟨*proof*⟩

lemma *less-nat-qbs-morphism*:
$(<) \in \mathbb{N}_Q \to_Q \textit{exp-qbs}\ \mathbb{N}_Q\ \mathbb{B}_Q$
⟨*proof*⟩

lemma *less-real-qbs-morphism*:

$(<) \in \mathbb{R}_Q \to_Q$ *exp-qbs* $\mathbb{R}_Q$ $\mathbb{B}_Q$
⟨*proof*⟩

lemma *rec-list-morphism'*:
rec-list' ∈ *qbs-space* (*exp-qbs* Y (*exp-qbs* (*exp-qbs* X (*exp-qbs* (*list-of* X) (*exp-qbs* Y Y))) (*exp-qbs* (*list-of* X) Y)))
⟨*proof*⟩

end

3 Probability Spaces

3.1 Probability Measures

theory *Probability-Space-QuasiBorel*
imports *Exponent-QuasiBorel*
begin

3.1.1 Probability Measures

type-synonym *'a qbs-prob-t* = *'a quasi-borel* ∗ (*real* ⇒ *'a*) ∗ *real measure*

locale *in-Mx* =
fixes X :: *'a quasi-borel*
and α :: *real* ⇒ *'a*
assumes *in-Mx*[*simp*]:α ∈ *qbs-Mx* X

locale *qbs-prob* = *in-Mx* X α + *real-distribution* μ
for X :: *'a quasi-borel* **and** α **and** μ
begin
declare *prob-space-axioms*[*simp*]

lemma *m-in-space-prob-algebra*[*simp*]:
μ ∈ *space* (*prob-algebra real-borel*)
⟨*proof*⟩
end

locale *pair-qbs-probs* = *qp1*:*qbs-prob* X α μ + *qp2*:*qbs-prob* Y β ν
for X :: *'a quasi-borel***and** α μ **and** Y :: *'b quasi-borel* **and** β ν
begin

sublocale *pair-prob-space* μ ν
⟨*proof*⟩

lemma *ab-measurable*[*measurable*]:
map-prod α β ∈ *real-borel* $\bigotimes_M$ *real-borel* $\to_M$ *qbs-to-measure* (X $\bigotimes_Q$ Y)
⟨*proof*⟩

lemma *ab-g-in-Mx*[*simp*]:
map-prod α β $\circ$ *real-real.g* $\in$ *pair-qbs-Mx X Y*
⟨*proof*⟩

sublocale *qbs-prob X* $\bigotimes_Q$ *Y map-prod* α β $\circ$ *real-real.g distr* (μ $\bigotimes_M$ ν) *real-borel real-real.f*
⟨*proof*⟩

end

locale *pair-qbs-prob* = *qp1*:*qbs-prob X* α μ + *qp2*:*qbs-prob Y* β ν
for *X* :: *'a quasi-borel***and** α μ **and** *Y* :: *'a quasi-borel* **and** β ν
begin

sublocale *pair-qbs-probs*
⟨*proof*⟩

lemma *same-spaces*[*simp*]:
assumes *Y* = *X*
shows β $\in$ *qbs-Mx X*
⟨*proof*⟩

end

lemma *prob-algebra-real-prob-measure*:
p $\in$ *space* (*prob-algebra* (*real-borel*)) = *real-distribution p*
⟨*proof*⟩

lemma *qbs-probI*:
assumes α $\in$ *qbs-Mx X*
and *sets* μ = *sets borel*
and *prob-space* μ
shows *qbs-prob X* α μ
⟨*proof*⟩

lemma *qbs-empty-not-qbs-prob* :¬ *qbs-prob* (*empty-quasi-borel*) *f M*
⟨*proof*⟩

definition *qbs-prob-eq* :: [*'a qbs-prob-t*, *'a qbs-prob-t*] ⇒ *bool* **where**
qbs-prob-eq p1 p2 ≡
(*let* (*qbs1*, *a1*, *m1*) = *p1*;
(*qbs2*, *a2*, *m2*) = *p2 in*
qbs-prob qbs1 a1 m1 ∧ *qbs-prob qbs2 a2 m2* ∧ *qbs1* = *qbs2* ∧
distr m1 (*qbs-to-measure qbs1*) *a1* = *distr m2* (*qbs-to-measure qbs2*) *a2*)

definition *qbs-prob-eq2* :: [*'a qbs-prob-t*, *'a qbs-prob-t*] ⇒ *bool* **where**
qbs-prob-eq2 p1 p2 ≡
(*let* (*qbs1*, *a1*, *m1*) = *p1*;

$(qbs2,\ a2,\ m2) = p2\ in$
$qbs\text{-}prob\ qbs1\ a1\ m1 \wedge qbs\text{-}prob\ qbs2\ a2\ m2 \wedge qbs1 = qbs2 \wedge$
$(\forall f \in qbs1 \rightarrow_Q real\text{-}quasi\text{-}borel.$
$(\int x.\ f\ (a1\ x)\ \partial\ m1) = (\int x.\ f\ (a2\ x)\ \partial\ m2)))$

definition *qbs-prob-eq3* :: $['a\ qbs\text{-}prob\text{-}t,\ 'a\ qbs\text{-}prob\text{-}t] \Rightarrow bool$ **where**
$qbs\text{-}prob\text{-}eq3\ p1\ p2 \equiv$
$(let\ (qbs1,\ a1,\ m1) = p1;$
$(qbs2,\ a2,\ m2) = p2\ in$
$(qbs\text{-}prob\ qbs1\ a1\ m1 \wedge qbs\text{-}prob\ qbs2\ a2\ m2 \wedge qbs1 = qbs2 \wedge$
$(\forall f \in qbs1 \rightarrow_Q real\text{-}quasi\text{-}borel.$
$(\forall\ k \in qbs\text{-}space\ qbs1.\ 0 \le f\ k) \longrightarrow$
$(\int x.\ f\ (a1\ x)\ \partial\ m1) = (\int x.\ f\ (a2\ x)\ \partial\ m2))))$

definition *qbs-prob-eq4* :: $['a\ qbs\text{-}prob\text{-}t,\ 'a\ qbs\text{-}prob\text{-}t] \Rightarrow bool$ **where**
$qbs\text{-}prob\text{-}eq4\ p1\ p2 \equiv$
$(let\ (qbs1,\ a1,\ m1) = p1;$
$(qbs2,\ a2,\ m2) = p2\ in$
$(qbs\text{-}prob\ qbs1\ a1\ m1 \wedge qbs\text{-}prob\ qbs2\ a2\ m2 \wedge qbs1 = qbs2 \wedge$
$(\forall f \in qbs1 \rightarrow_Q \mathbb{R}_{Q\ge 0}.$
$(\int^+ x.\ f\ (a1\ x)\ \partial\ m1) = (\int^+ x.\ f\ (a2\ x)\ \partial\ m2))))$

lemma(**in** *qbs-prob*) *qbs-prob-eq-refl*[*simp*]:
$qbs\text{-}prob\text{-}eq\ (X,\alpha,\mu)\ (X,\alpha,\mu)$
⟨*proof*⟩

lemma(**in** *qbs-prob*) *qbs-prob-eq2-refl*[*simp*]:
$qbs\text{-}prob\text{-}eq2\ (X,\alpha,\mu)\ (X,\alpha,\mu)$
⟨*proof*⟩

lemma(**in** *qbs-prob*) *qbs-prob-eq3-refl*[*simp*]:
$qbs\text{-}prob\text{-}eq3\ (X,\alpha,\mu)\ (X,\alpha,\mu)$
⟨*proof*⟩

lemma(**in** *qbs-prob*) *qbs-prob-eq4-refl*[*simp*]:
$qbs\text{-}prob\text{-}eq4\ (X,\alpha,\mu)\ (X,\alpha,\mu)$
⟨*proof*⟩

lemma(**in** *pair-qbs-prob*) *qbs-prob-eq-intro*:
assumes $X = Y$
and $distr\ \mu\ (qbs\text{-}to\text{-}measure\ X)\ \alpha = distr\ \nu\ (qbs\text{-}to\text{-}measure\ X)\ \beta$
shows $qbs\text{-}prob\text{-}eq\ (X,\alpha,\mu)\ (Y,\beta,\nu)$
⟨*proof*⟩

lemma(**in** *pair-qbs-prob*) *qbs-prob-eq2-intro*:
assumes $X = Y$
and $\bigwedge f.\ f \in qbs\text{-}to\text{-}measure\ X \rightarrow_M real\text{-}borel$
$\Longrightarrow (\int x.\ f\ (\alpha\ x)\ \partial\ \mu) = (\int x.\ f\ (\beta\ x)\ \partial\ \nu)$
shows $qbs\text{-}prob\text{-}eq2\ (X,\alpha,\mu)\ (Y,\beta,\nu)$

⟨*proof*⟩

lemma(**in** *pair-qbs-prob*) *qbs-prob-eq3-intro*:
assumes $X = Y$
and $\bigwedge f.\ f \in$ *qbs-to-measure* $X \rightarrow_M$ *real-borel* $\Longrightarrow (\forall\ k \in$ *qbs-space* $X.\ 0 \leq f$
$k)$
$\Longrightarrow (\int x.\ f\ (\alpha\ x)\ \partial\ \mu) = (\int x.\ f\ (\beta\ x)\ \partial\ \nu)$
shows *qbs-prob-eq3* $(X,\alpha,\mu)\ (Y,\beta,\nu)$
⟨*proof*⟩

lemma(**in** *pair-qbs-prob*) *qbs-prob-eq4-intro*:
assumes $X = Y$
and $\bigwedge f.\ f \in$ *qbs-to-measure* $X \rightarrow_M$ *ennreal-borel*
$\Longrightarrow (\int^+ x.\ f\ (\alpha\ x)\ \partial\ \mu) = (\int^+ x.\ f\ (\beta\ x)\ \partial\ \nu)$
shows *qbs-prob-eq4* $(X,\alpha,\mu)\ (Y,\beta,\nu)$
⟨*proof*⟩

lemma *qbs-prob-eq-dest*:
assumes *qbs-prob-eq* $(X,\alpha,\mu)\ (Y,\beta,\nu)$
shows *qbs-prob* $X\ \alpha\ \mu$
qbs-prob $Y\ \beta\ \nu$
$Y = X$
and *distr* μ (*qbs-to-measure* X) α = *distr* ν (*qbs-to-measure* X) β
⟨*proof*⟩

lemma *qbs-prob-eq2-dest*:
assumes *qbs-prob-eq2* $(X,\alpha,\mu)\ (Y,\beta,\nu)$
shows *qbs-prob* $X\ \alpha\ \mu$
qbs-prob $Y\ \beta\ \nu$
$Y = X$
and $\bigwedge f.\ f \in$ *qbs-to-measure* $X \rightarrow_M$ *real-borel*
$\Longrightarrow (\int x.\ f\ (\alpha\ x)\ \partial\ \mu) = (\int x.\ f\ (\beta\ x)\ \partial\ \nu)$
⟨*proof*⟩

lemma *qbs-prob-eq3-dest*:
assumes *qbs-prob-eq3* $(X,\alpha,\mu)\ (Y,\beta,\nu)$
shows *qbs-prob* $X\ \alpha\ \mu$
qbs-prob $Y\ \beta\ \nu$
$Y = X$
and $\bigwedge f.\ f \in$ *qbs-to-measure* $X \rightarrow_M$ *real-borel* $\Longrightarrow (\forall\ k \in$ *qbs-space* $X.\ 0 \leq f$
$k)$
$\Longrightarrow (\int x.\ f\ (\alpha\ x)\ \partial\ \mu) = (\int x.\ f\ (\beta\ x)\ \partial\ \nu)$
⟨*proof*⟩

lemma *qbs-prob-eq4-dest*:
assumes *qbs-prob-eq4* $(X,\alpha,\mu)\ (Y,\beta,\nu)$
shows *qbs-prob* $X\ \alpha\ \mu$
qbs-prob $Y\ \beta\ \nu$

$Y = X$
and $\bigwedge f.\ f \in$ *qbs-to-measure* $X \to_M$ *ennreal-borel*
$\Longrightarrow (\int^+ x.\ f\ (\alpha\ x)\ \partial\ \mu) = (\int^+ x.\ f\ (\beta\ x)\ \partial\ \nu)$
⟨*proof*⟩

definition *qbs-prob-t-ennintegral* :: [′*a qbs-prob-t*, ′*a* ⇒ *ennreal*] ⇒ *ennreal* **where**
qbs-prob-t-ennintegral p f ≡
(*if f* ∈ (*fst p*) $\to_Q$ *ennreal-quasi-borel*
then $(\int^+ x.\ f\ (fst\ (snd\ p)\ x)\ \partial\ (snd\ (snd\ p)))$ *else 0*)

definition *qbs-prob-t-integral* :: [′*a qbs-prob-t*, ′*a* ⇒ *real*] ⇒ *real* **where**
qbs-prob-t-integral p f ≡
(*if f* ∈ (*fst p*) $\to_Q \mathbb{R}_Q$
then $(\int x.\ f\ (fst\ (snd\ p)\ x)\ \partial\ (snd\ (snd\ p)))$
else 0)

definition *qbs-prob-t-integrable* :: [′*a qbs-prob-t*, ′*a* ⇒ *real*] ⇒ *bool* **where**
qbs-prob-t-integrable p f ≡ *f* ∈ *fst p* $\to_Q$ *real-quasi-borel* ∧ *integrable* (*snd* (*snd p*)) (*f* ∘ (*fst* (*snd p*)))

definition *qbs-prob-t-measure* :: ′*a qbs-prob-t* ⇒ ′*a measure* **where**
qbs-prob-t-measure p ≡ *distr* (*snd* (*snd p*)) (*qbs-to-measure* (*fst p*)) (*fst* (*snd p*))

lemma *qbs-prob-eq-symp*:
symp qbs-prob-eq
⟨*proof*⟩

lemma *qbs-prob-eq-transp*:
transp qbs-prob-eq
⟨*proof*⟩

quotient-type ′*a qbs-prob-space* = ′*a qbs-prob-t* / *partial*: *qbs-prob-eq*
morphisms *rep-qbs-prob-space qbs-prob-space*
⟨*proof*⟩

interpretation *qbs-prob-space* : *quot-type qbs-prob-eq Abs-qbs-prob-space Rep-qbs-prob-space*
⟨*proof*⟩

lemma *qbs-prob-space-induct*:
assumes $\bigwedge X\ \alpha\ \mu.$ *qbs-prob* $X\ \alpha\ \mu \Longrightarrow P$ (*qbs-prob-space* (X,α,μ))
shows *P s*
⟨*proof*⟩

lemma *qbs-prob-space-induct*′:
assumes $\bigwedge X\ \alpha\ \mu.$ *qbs-prob* $X\ \alpha\ \mu \Longrightarrow s =$ *qbs-prob-space* $(X,\alpha,\mu) \Longrightarrow P$ (*qbs-prob-space* (X,α,μ))
shows *P s*
⟨*proof*⟩

lemma *rep-qbs-prob-space*:
$\exists X\ \alpha\ \mu.\ p = \textit{qbs-prob-space}\ (X,\ \alpha,\ \mu) \wedge \textit{qbs-prob}\ X\ \alpha\ \mu$
⟨*proof*⟩

lemma(**in** *qbs-prob*) *in-Rep*:
$(X,\ \alpha,\ \mu) \in \textit{Rep-qbs-prob-space}\ (\textit{qbs-prob-space}\ (X,\alpha,\mu))$
⟨*proof*⟩

lemma(**in** *qbs-prob*) *if-in-Rep*:
assumes $(X',\alpha',\mu') \in \textit{Rep-qbs-prob-space}\ (\textit{qbs-prob-space}\ (X,\alpha,\mu))$
shows $X' = X$
$\textit{qbs-prob}\ X'\ \alpha'\ \mu'$
$\textit{qbs-prob-eq}\ (X,\alpha,\mu)\ (X',\alpha',\mu')$
⟨*proof*⟩

lemma(**in** *qbs-prob*) *in-Rep-induct*:
assumes $\bigwedge Y\ \beta\ \nu.\ (Y,\beta,\nu) \in \textit{Rep-qbs-prob-space}\ (\textit{qbs-prob-space}\ (X,\alpha,\mu)) \Longrightarrow P\ (Y,\beta,\nu)$
shows $P\ (\textit{rep-qbs-prob-space}\ (\textit{qbs-prob-space}\ (X,\alpha,\mu)))$
⟨*proof*⟩

lemma *qbs-prob-eq-2-implies-3* :
assumes *qbs-prob-eq2 p1 p2*
shows *qbs-prob-eq3 p1 p2*
⟨*proof*⟩

lemma *qbs-prob-eq-3-implies-1* :
assumes *qbs-prob-eq3* (*p1* :: ′*a qbs-prob-t*) *p2*
shows *qbs-prob-eq p1 p2*
⟨*proof*⟩

lemma *qbs-prob-eq-1-implies-2* :
assumes *qbs-prob-eq p1* (*p2* :: ′*a qbs-prob-t*)
shows *qbs-prob-eq2 p1 p2*
⟨*proof*⟩

lemma *qbs-prob-eq-1-implies-4* :
assumes *qbs-prob-eq p1 p2*
shows *qbs-prob-eq4 p1 p2*
⟨*proof*⟩

lemma *qbs-prob-eq-4-implies-3* :
assumes *qbs-prob-eq4 p1 p2*
shows *qbs-prob-eq3 p1 p2*
⟨*proof*⟩

lemma *qbs-prob-eq-equiv12* :
qbs-prob-eq = *qbs-prob-eq2*

⟨*proof*⟩

lemma *qbs-prob-eq-equiv13* :
qbs-prob-eq = *qbs-prob-eq3*
⟨*proof*⟩

lemma *qbs-prob-eq-equiv14* :
qbs-prob-eq = *qbs-prob-eq4*
⟨*proof*⟩

lemma *qbs-prob-eq-equiv23* :
qbs-prob-eq2 = *qbs-prob-eq3*
⟨*proof*⟩

lemma *qbs-prob-eq-equiv24* :
qbs-prob-eq2 = *qbs-prob-eq4*
⟨*proof*⟩

lemma *qbs-prob-eq-equiv34*:
qbs-prob-eq3 = *qbs-prob-eq4*
⟨*proof*⟩

lemma *qbs-prob-eq-equiv31* :
qbs-prob-eq = *qbs-prob-eq3*
⟨*proof*⟩

lemma *qbs-prob-space-eq*:
assumes *qbs-prob-eq* (X,α,μ) (Y,β,ν)
shows *qbs-prob-space* (X,α,μ) = *qbs-prob-space* (Y,β,ν)
⟨*proof*⟩

lemma(**in** *pair-qbs-prob*) *qbs-prob-space-eq*:
assumes $Y = X$
and *distr* μ (*qbs-to-measure* X) α = *distr* ν (*qbs-to-measure* X) β
shows *qbs-prob-space* (X,α,μ) = *qbs-prob-space* (Y,β,ν)
⟨*proof*⟩

lemma(**in** *pair-qbs-prob*) *qbs-prob-space-eq2*:
assumes $Y = X$
and $\bigwedge f.\ f \in$ *qbs-to-measure* $X \rightarrow_M$ *real-borel*
$\Longrightarrow (\int x.\ f\ (\alpha\ x)\ \partial\ \mu) = (\int x.\ f\ (\beta\ x)\ \partial\ \nu)$
shows *qbs-prob-space* (X,α,μ) = *qbs-prob-space* (Y,β,ν)
⟨*proof*⟩

lemma(**in** *pair-qbs-prob*) *qbs-prob-space-eq3*:
assumes $Y = X$
and $\bigwedge f.\ f \in$ *qbs-to-measure* $X \rightarrow_M$ *real-borel* $\Longrightarrow$ ($\forall k \in$ *qbs-space* $X.\ 0 \le f$
k)
$\Longrightarrow (\int x.\ f\ (\alpha\ x)\ \partial\ \mu) = (\int x.\ f\ (\beta\ x)\ \partial\ \nu)$

shows *qbs-prob-space* (X,α,μ) = *qbs-prob-space* (Y,β,ν)
⟨*proof*⟩

lemma(**in** *pair-qbs-prob*) *qbs-prob-space-eq4*:
assumes $Y = X$
and $\bigwedge f.\ f \in$ *qbs-to-measure* $X \rightarrow_M$ *ennreal-borel*
$\Longrightarrow (\int^+ x.\ f\ (\alpha\ x)\ \partial\ \mu) = (\int^+ x.\ f\ (\beta\ x)\ \partial\ \nu)$
shows *qbs-prob-space* (X,α,μ) = *qbs-prob-space* (Y,β,ν)
⟨*proof*⟩

lemma(**in** *pair-qbs-prob*) *qbs-prob-space-eq-inverse*:
assumes *qbs-prob-space* (X,α,μ) = *qbs-prob-space* (Y,β,ν)
shows *qbs-prob-eq* (X,α,μ) (Y,β,ν)
and *qbs-prob-eq2* (X,α,μ) (Y,β,ν)
and *qbs-prob-eq3* (X,α,μ) (Y,β,ν)
and *qbs-prob-eq4* (X,α,μ) (Y,β,ν)
⟨*proof*⟩

lift-definition *qbs-prob-space-qbs* :: *'a qbs-prob-space* ⇒ *'a quasi-borel*
is *fst* ⟨*proof*⟩

lemma(**in** *qbs-prob*) *qbs-prob-space-qbs-computation*[*simp*]:
qbs-prob-space-qbs (*qbs-prob-space* (X,α,μ)) = X
⟨*proof*⟩

lemma *rep-qbs-prob-space'*:
assumes *qbs-prob-space-qbs* $s = X$
shows $\exists\,\alpha\ \mu.\ s$ = *qbs-prob-space* $(X,\alpha,\mu) \wedge$ *qbs-prob* $X\ \alpha\ \mu$
⟨*proof*⟩

lift-definition *qbs-prob-ennintegral* :: [*'a qbs-prob-space*, *'a* ⇒ *ennreal*] ⇒ *ennreal*
is *qbs-prob-t-ennintegral*
⟨*proof*⟩

lift-definition *qbs-prob-integral* :: [*'a qbs-prob-space*, *'a* ⇒ *real*] ⇒ *real*
is *qbs-prob-t-integral*
⟨*proof*⟩

syntax
-qbs-prob-ennintegral :: *pttrn* ⇒ *ennreal* ⇒ *'a qbs-prob-space* ⇒ *ennreal* (‹$\int^+{}_Q$((2 -./ -)/ ∂-)› [60,61] 110)

syntax-consts
-qbs-prob-ennintegral ⇌ *qbs-prob-ennintegral*

translations
$\int^+{}_Q\ x.\ f\ \partial p$ ⇌ *CONST qbs-prob-ennintegral* $p\ (\lambda x.\ f)$

syntax
-qbs-prob-integral :: *pttrn* ⇒ *real* ⇒ *'a qbs-prob-space* ⇒ *real* (‹$\int_Q$((2 -./ -)/ ∂-)›
[60,61] 110)

syntax-consts
-qbs-prob-integral ⇌ *qbs-prob-integral*

translations
$\int_Q$ *x. f* ∂*p* ⇌ *CONST qbs-prob-integral p* (λ*x. f*)

We define the function $l_X \in L(P(X)) \to_M G(X)$.

lift-definition *qbs-prob-measure* :: *'a qbs-prob-space* ⇒ *'a measure*
is *qbs-prob-t-measure*
⟨*proof*⟩

declare [[*coercion qbs-prob-measure*]]

lemma(**in** *qbs-prob*) *qbs-prob-measure-computation*[*simp*]:
qbs-prob-measure (*qbs-prob-space* (*X*,α,μ)) = *distr* μ (*qbs-to-measure X*) α
⟨*proof*⟩

definition *qbs-emeasure* ::*'a qbs-prob-space* ⇒ *'a set* ⇒ *ennreal* **where**
qbs-emeasure s ≡ *emeasure* (*qbs-prob-measure s*)

lemma(**in** *qbs-prob*) *qbs-emeasure-computation*[*simp*]:
assumes *U* ∈ *sets* (*qbs-to-measure X*)
shows *qbs-emeasure* (*qbs-prob-space* (*X*,α,μ)) *U* = *emeasure* μ (α −‘ *U*)
⟨*proof*⟩

definition *qbs-measure* ::*'a qbs-prob-space* ⇒ *'a set* ⇒ *real* **where**
qbs-measure s ≡ *measure* (*qbs-prob-measure s*)

interpretation *qbs-prob-measure-prob-space* : *prob-space qbs-prob-measure* (*s*::*'a qbs-prob-space*) **for** *s*
⟨*proof*⟩

lemma *qbs-prob-measure-space*:
qbs-space (*qbs-prob-space-qbs s*) = *space* (*qbs-prob-measure s*)
⟨*proof*⟩

lemma *qbs-prob-measure-sets*[*measurable-cong*]:
sets (*qbs-to-measure* (*qbs-prob-space-qbs s*)) = *sets* (*qbs-prob-measure s*)
⟨*proof*⟩

lemma(**in** *qbs-prob*) *qbs-prob-ennintegral-def*:
assumes $f \in X \to_Q \mathbb{R}_{Q\geq 0}$

shows *qbs-prob-ennintegral* (*qbs-prob-space* (X,α,μ)) f = ($\int^+ x.$ f (α x) ∂ μ)
⟨*proof*⟩

lemma(**in** *qbs-prob*) *qbs-prob-ennintegral-def2*:
assumes $f \in X \to_Q \mathbb{R}_{Q \geq 0}$
shows *qbs-prob-ennintegral* (*qbs-prob-space* (X,α,μ)) f = $integral^N$ (*distr* μ (*qbs-to-measure* X) α) f
⟨*proof*⟩

lemma (**in** *qbs-prob*) *qbs-prob-ennintegral-not-morphism*:
assumes $f \notin X \to_Q \mathbb{R}_{Q \geq 0}$
shows *qbs-prob-ennintegral* (*qbs-prob-space* (X,α,μ)) f = *0*
⟨*proof*⟩

lemma *qbs-prob-ennintegral-def2*:
assumes *qbs-prob-space-qbs* s = (X :: *'a quasi-borel*)
and $f \in X \to_Q \mathbb{R}_{Q \geq 0}$
shows *qbs-prob-ennintegral* s f = $integral^N$ (*qbs-prob-measure* s) f
⟨*proof*⟩

lemma(**in** *qbs-prob*) *qbs-prob-integral-def*:
assumes $f \in X \to_Q$ *real-quasi-borel*
shows *qbs-prob-integral* (*qbs-prob-space* (X,α,μ)) f = ($\int x.$ f (α x) ∂ μ)
⟨*proof*⟩

lemma(**in** *qbs-prob*) *qbs-prob-integral-def2*:
qbs-prob-integral (*qbs-prob-space* (X,α,μ)) f = $integral^L$ (*distr* μ (*qbs-to-measure* X) α) f
⟨*proof*⟩

lemma *qbs-prob-integral-def2*:
qbs-prob-integral (s::*'a qbs-prob-space*) f = $integral^L$ (*qbs-prob-measure* s) f
⟨*proof*⟩

definition *qbs-prob-var* :: *'a qbs-prob-space* ⇒ (*'a* ⇒ *real*) ⇒ *real* **where**
qbs-prob-var s f ≡ *qbs-prob-integral* s ($\lambda x.$ (f x − *qbs-prob-integral* s f)^2)

lemma(**in** *qbs-prob*) *qbs-prob-var-computation*:
assumes $f \in X \to_Q$ *real-quasi-borel*
shows *qbs-prob-var* (*qbs-prob-space* (X,α,μ)) f = ($\int x.$ (f (α x) − ($\int x.$ f (α x) ∂ μ))^2 ∂ μ)
⟨*proof*⟩

lift-definition *qbs-integrable* :: [*'a qbs-prob-space*, *'a* ⇒ *real*] ⇒ *bool*
is *qbs-prob-t-integrable*
⟨*proof*⟩

lemma(**in** *qbs-prob*) *qbs-integrable-def*:
qbs-integrable (*qbs-prob-space* (X, α, μ)) f = ($f \in X \to_Q \mathbb{R}_Q$ ∧ *integrable* μ (f ∘

α))
⟨*proof*⟩

lemma *qbs-integrable-morphism*:
assumes *qbs-prob-space-qbs s* = *X*
and *qbs-integrable s f*
shows $f \in X \rightarrow_Q \mathbb{R}_Q$
⟨*proof*⟩

lemma(**in** *qbs-prob*) *qbs-integrable-measurable*[*simp*,*measurable*]:
assumes *qbs-integrable* (*qbs-prob-space* (X,α,μ)) *f*
shows $f \in$ *qbs-to-measure* $X \rightarrow_M$ *real-borel*
⟨*proof*⟩

lemma *qbs-integrable-iff-integrable*:
(*qbs-integrable* (*s*::′*a qbs-prob-space*) *f*) = (*integrable* (*qbs-prob-measure s*) *f*)
⟨*proof*⟩

lemma(**in** *qbs-prob*) *qbs-integrable-iff-integrable-distr*:
qbs-integrable (*qbs-prob-space* (X,α,μ)) *f* = *integrable* (*distr* μ (*qbs-to-measure X*) α) *f*
⟨*proof*⟩

lemma(**in** *qbs-prob*) *qbs-integrable-iff-integrable*:
assumes $f \in$ *qbs-to-measure* $X \rightarrow_M$ *real-borel*
shows *qbs-integrable* (*qbs-prob-space* (X,α,μ)) *f* = *integrable* μ (λx. *f* (α *x*))
⟨*proof*⟩

lemma *qbs-integrable-if-integrable*:
assumes *integrable* (*qbs-prob-measure s*) *f*
shows *qbs-integrable* (*s*::′*a qbs-prob-space*) *f*
⟨*proof*⟩

lemma *integrable-if-qbs-integrable*:
assumes *qbs-integrable* (*s*::′*a qbs-prob-space*) *f*
shows *integrable* (*qbs-prob-measure s*) *f*
⟨*proof*⟩

lemma *qbs-integrable-iff-bounded*:
assumes *qbs-prob-space-qbs s* = *X*
shows *qbs-integrable s f* $\longleftrightarrow$ $f \in X \rightarrow_Q \mathbb{R}_Q$ $\wedge$ *qbs-prob-ennintegral s* (λx. *ennreal* $|f\ x\ |$) $< \infty$
(**is** *?lhs* = *?rhs*)
⟨*proof*⟩

lemma *qbs-integrable-cong*:
assumes *qbs-prob-space-qbs s* = *X*
$\bigwedge x.\ x \in$ *qbs-space X* $\Longrightarrow f\ x = g\ x$
and *qbs-integrable s f*

shows *qbs-integrable s g*
⟨*proof*⟩

lemma *qbs-integrable-const*[*simp*]:
qbs-integrable s ($\lambda x.\ c$)
⟨*proof*⟩

lemma *qbs-integrable-add*[*simp*]:
assumes *qbs-integrable s f*
and *qbs-integrable s g*
shows *qbs-integrable s* ($\lambda x.\ f\ x + g\ x$)
⟨*proof*⟩

lemma *qbs-integrable-diff*[*simp*]:
assumes *qbs-integrable s f*
and *qbs-integrable s g*
shows *qbs-integrable s* ($\lambda x.\ f\ x - g\ x$)
⟨*proof*⟩

lemma *qbs-integrable-mult-iff*[*simp*]:
(*qbs-integrable s* ($\lambda x.\ c * f\ x$)) = ($c = 0 \lor$ *qbs-integrable s f*)
⟨*proof*⟩

lemma *qbs-integrable-mult*[*simp*]:
assumes *qbs-integrable s f*
shows *qbs-integrable s* ($\lambda x.\ c * f\ x$)
⟨*proof*⟩

lemma *qbs-integrable-abs*[*simp*]:
assumes *qbs-integrable s f*
shows *qbs-integrable s* ($\lambda x.\ |f\ x|$)
⟨*proof*⟩

lemma *qbs-integrable-sq*[*simp*]:
assumes *qbs-integrable s f*
and *qbs-integrable s* ($\lambda x.\ (f\ x)^2$)
shows *qbs-integrable s* ($\lambda x.\ (f\ x - c)^2$)
⟨*proof*⟩

lemma *qbs-ennintegral-eq-qbs-integral*:
assumes *qbs-prob-space-qbs s* = X
qbs-integrable s f
and $\bigwedge x.\ x \in$ *qbs-space* $X \Longrightarrow 0 \le f\ x$
shows *qbs-prob-ennintegral s* ($\lambda x.$ *ennreal* ($f\ x$)) = *ennreal* (*qbs-prob-integral s f*)
⟨*proof*⟩

lemma *qbs-prob-ennintegral-cong*:
assumes *qbs-prob-space-qbs s* = X

and $\bigwedge x.\ x \in$ *qbs-space* $X \Longrightarrow f\, x = g\, x$
shows *qbs-prob-ennintegral* $s\, f$ = *qbs-prob-ennintegral* $s\, g$
⟨*proof*⟩

lemma *qbs-prob-ennintegral-const*:
qbs-prob-ennintegral (s::$'a$ *qbs-prob-space*) ($\lambda x.\ c$) = c
⟨*proof*⟩

lemma *qbs-prob-ennintegral-add*:
assumes *qbs-prob-space-qbs* $s = X$
$f \in$ (X::$'a$ *quasi-borel*) $\rightarrow_Q \mathbb{R}_{Q\geq 0}$
and $g \in X \rightarrow_Q \mathbb{R}_{Q\geq 0}$
shows *qbs-prob-ennintegral* s ($\lambda x.\ f\, x + g\, x$) = *qbs-prob-ennintegral* $s\, f$ + *qbs-prob-ennintegral* $s\, g$
⟨*proof*⟩

lemma *qbs-prob-ennintegral-cmult*:
assumes *qbs-prob-space-qbs* $s = X$
and $f \in X \rightarrow_Q \mathbb{R}_{Q\geq 0}$
shows *qbs-prob-ennintegral* s ($\lambda x.\ c * f\, x$) = c * *qbs-prob-ennintegral* $s\, f$
⟨*proof*⟩

lemma *qbs-prob-ennintegral-cmult-noninfty*:
assumes $c \neq \infty$
shows *qbs-prob-ennintegral* s ($\lambda x.\ c * f\, x$) = c * *qbs-prob-ennintegral* $s\, f$
⟨*proof*⟩

lemma *qbs-prob-integral-cong*:
assumes *qbs-prob-space-qbs* $s = X$
and $\bigwedge x.\ x \in$ *qbs-space* $X \Longrightarrow f\, x = g\, x$
shows *qbs-prob-integral* $s\, f$ = *qbs-prob-integral* $s\, g$
⟨*proof*⟩

lemma *qbs-prob-integral-nonneg*:
assumes *qbs-prob-space-qbs* $s = X$
and $\bigwedge x.\ x \in$ *qbs-space* $X \Longrightarrow 0 \leq f\, x$
shows $0 \leq$ *qbs-prob-integral* $s\, f$
⟨*proof*⟩

lemma *qbs-prob-integral-mono*:
assumes *qbs-prob-space-qbs* $s = X$
qbs-integrable (s :: $'a$ *qbs-prob-space*) f
qbs-integrable $s\, g$
and $\bigwedge x.\ x \in$ *qbs-space* $X \Longrightarrow f\, x \leq g\, x$
shows *qbs-prob-integral* $s\, f \leq$ *qbs-prob-integral* $s\, g$
⟨*proof*⟩

lemma *qbs-prob-integral-const*:

qbs-prob-integral (*s*::'*a qbs-prob-space*) (*λx. c*) = *c*
⟨*proof*⟩

lemma *qbs-prob-integral-add*:
assumes *qbs-integrable* (*s*::'*a qbs-prob-space*) *f*
and *qbs-integrable s g*
shows *qbs-prob-integral s* (*λx. f x* + *g x*) = *qbs-prob-integral s f* + *qbs-prob-integral s g*
⟨*proof*⟩

lemma *qbs-prob-integral-diff*:
assumes *qbs-integrable* (*s*::'*a qbs-prob-space*) *f*
and *qbs-integrable s g*
shows *qbs-prob-integral s* (*λx. f x* − *g x*) = *qbs-prob-integral s f* − *qbs-prob-integral s g*
⟨*proof*⟩

lemma *qbs-prob-integral-cmult*:
qbs-prob-integral s (*λx. c* ∗ *f x*) = *c* ∗ *qbs-prob-integral s f*
⟨*proof*⟩

lemma *real-qbs-prob-integral-def*:
assumes *qbs-integrable* (*s*::'*a qbs-prob-space*) *f*
shows *qbs-prob-integral s f* = *enn2real* (*qbs-prob-ennintegral s* (*λx. ennreal* (*f x*))) − *enn2real* (*qbs-prob-ennintegral s* (*λx. ennreal* (− *f x*)))
⟨*proof*⟩

lemma *qbs-prob-var-eq*:
assumes *qbs-integrable* (*s*::'*a qbs-prob-space*) *f*
and *qbs-integrable s* (*λx.* $(f\ x)^2$)
shows *qbs-prob-var s f* = *qbs-prob-integral s* (*λx.* $(f\ x)^2$) − $(qbs\text{-}prob\text{-}integral\ s\ f)^2$
⟨*proof*⟩

lemma *qbs-prob-var-affine*:
assumes *qbs-integrable s f*
shows *qbs-prob-var s* (*λx. a* ∗ *f x* + *b*) = a^2 ∗ *qbs-prob-var s f*
(**is** *?lhs* = *?rhs*)
⟨*proof*⟩

lemma *qbs-prob-integral-Markov-inequality*:
assumes *qbs-prob-space-qbs s* = *X*
and *qbs-integrable s f*
⋀*x. x* ∈ *qbs-space X* ⟹ *0* ≤ *f x*
and *0* < *c*
shows *qbs-emeasure s* {*x* ∈ *qbs-space X*. *c* ≤ *f x*} ≤ *ennreal* (*1*/*c* ∗ *qbs-prob-integral s f*)
⟨*proof*⟩

lemma *qbs-prob-integral-Markov-inequality'*:
assumes *qbs-prob-space-qbs s* = *X*
qbs-integrable s f
$\bigwedge x.\ x \in$ *qbs-space* (*qbs-prob-space-qbs s*) $\Longrightarrow 0 \le f\,x$
and $0 < c$
shows *qbs-measure s* $\{x \in$ *qbs-space* (*qbs-prob-space-qbs s*). $c \le f\,x\} \le (1/c\ *$ *qbs-prob-integral s f*)
⟨*proof*⟩

lemma *qbs-prob-integral-Markov-inequality-abs*:
assumes *qbs-prob-space-qbs s* = *X*
qbs-integrable s f
and $0 < c$
shows *qbs-emeasure s* $\{x \in$ *qbs-space X*. $c \le |f\,x|\} \le$ *ennreal* ($1/c\ *$ *qbs-prob-integral s* ($\lambda x.\ |f\,x|$))
⟨*proof*⟩

lemma *qbs-prob-integral-Markov-inequality-abs'*:
assumes *qbs-prob-space-qbs s* = *X*
qbs-integrable s f
and $0 < c$
shows *qbs-measure s* $\{x \in$ *qbs-space X*. $c \le |f\,x|\} \le (1/c\ *$ *qbs-prob-integral s* ($\lambda x.\ |f\,x|$))
⟨*proof*⟩

lemma *qbs-prob-integral-real-Markov-inequality*:
assumes *qbs-prob-space-qbs s* = $\mathbb{R}_Q$
qbs-integrable s f
and $0 < c$
shows *qbs-emeasure s* $\{r.\ c \le |f\,r|\} \le$ *ennreal* ($1/c\ *$ *qbs-prob-integral s* ($\lambda x.\ |f\,x|$))
⟨*proof*⟩

lemma *qbs-prob-integral-real-Markov-inequality'*:
assumes *qbs-prob-space-qbs s* = $\mathbb{R}_Q$
qbs-integrable s f
and $0 < c$
shows *qbs-measure s* $\{r.\ c \le |f\,r|\} \le 1/c\ *$ *qbs-prob-integral s* ($\lambda x.\ |f\,x|$)
⟨*proof*⟩

lemma *qbs-prob-integral-Chebyshev-inequality*:
assumes *qbs-prob-space-qbs s* = *X*
qbs-integrable s f
qbs-integrable s ($\lambda x.\ (f\,x)^2$)
and $0 < b$
shows *qbs-measure s* $\{x \in$ *qbs-space X*. $b \le |f\,x -$ *qbs-prob-integral s f*$|\} \le 1\ /\ b^2\ *$ *qbs-prob-var s f*
⟨*proof*⟩

end

3.2 The Probability Monad

theory *Monad-QuasiBorel*
  imports *Probability-Space-QuasiBorel*
begin

3.2.1 The Probability Monad P

definition *monadP-qbs-Px* :: *′a quasi-borel* ⇒ *′a qbs-prob-space set* **where**
monadP-qbs-Px X ≡ {*s. qbs-prob-space-qbs s* = *X*}

locale *in-Px* =
  fixes *X* :: *′a quasi-borel* **and** *s* :: *′a qbs-prob-space*
  assumes *in-Px*:*s* ∈ *monadP-qbs-Px X*
begin

lemma *qbs-prob-space-X*[*simp*]:
 qbs-prob-space-qbs s = *X*
  ⟨*proof*⟩

end

locale *in-MPx* =
  fixes *X* :: *′a quasi-borel* **and** β :: *real* ⇒ *′a qbs-prob-space*
  assumes *ex*:∃ α∈ *qbs-Mx X*. ∃ *g* ∈ *real-borel* $\rightarrow_M$ *prob-algebra real-borel*.
                ∀ *r*. β *r* = *qbs-prob-space* (*X*,α,*g r*)
begin

lemma *rep-inMPx*:
 ∃ α *g*. α ∈ *qbs-Mx X* ∧ *g* ∈ *real-borel* $\rightarrow_M$ *prob-algebra real-borel* ∧
    β = (λ*r*. *qbs-prob-space* (*X*,α,*g r*))
⟨*proof*⟩

end

definition *monadP-qbs-MPx* :: *′a quasi-borel* ⇒ (*real* ⇒ *′a qbs-prob-space*) *set*
where
monadP-qbs-MPx X ≡ {β. *in-MPx X* β}

definition *monadP-qbs* :: *′a quasi-borel* ⇒ *′a qbs-prob-space quasi-borel* **where**
monadP-qbs X ≡ *Abs-quasi-borel* (*monadP-qbs-Px X*, *monadP-qbs-MPx X*)

lemma(**in** *qbs-prob*) *qbs-prob-space-in-Px*:
  qbs-prob-space (*X*,α,μ) ∈ *monadP-qbs-Px X*
  ⟨*proof*⟩

lemma *rep-monadP-qbs-Px*:
  assumes *s* ∈ *monadP-qbs-Px X*

shows $\exists\alpha\ \mu.\ s = $ *qbs-prob-space* $(X,\ \alpha,\ \mu) \wedge$ *qbs-prob* $X\ \alpha\ \mu$
⟨*proof*⟩

lemma *rep-monadP-qbs-MPx*:
assumes $\beta \in$ *monadP-qbs-MPx* X
shows $\exists\alpha\ g.\ \alpha \in$ *qbs-Mx* $X \wedge g \in$ *real-borel* $\rightarrow_M$ *prob-algebra real-borel* $\wedge$
$\beta = (\lambda r.$ *qbs-prob-space* $(X,\alpha,g\ r))$
⟨*proof*⟩

lemma *qbs-prob-MPx*:
assumes $\alpha \in$ *qbs-Mx* X
and $g \in$ *real-borel* $\rightarrow_M$ *prob-algebra real-borel*
shows *qbs-prob* $X\ \alpha\ (g\ r)$
⟨*proof*⟩

lemma *monadP-qbs-f*[*simp*]: *monadP-qbs-MPx* $X \subseteq$ *UNIV* $\rightarrow$ *monadP-qbs-Px* X
⟨*proof*⟩

lemma *monadP-qbs-closed1*: *qbs-closed1* (*monadP-qbs-MPx* X)
⟨*proof*⟩

lemma *monadP-qbs-closed2*: *qbs-closed2* (*monadP-qbs-Px* X) (*monadP-qbs-MPx* X)
⟨*proof*⟩

lemma *monadP-qbs-closed3*: *qbs-closed3* (*monadP-qbs-MPx* (X :: *'a quasi-borel*))
⟨*proof*⟩

lemma *monadP-qbs-correct*: *Rep-quasi-borel* (*monadP-qbs* X) = (*monadP-qbs-Px* X, *monadP-qbs-MPx* X)
⟨*proof*⟩

lemma *monadP-qbs-space*[*simp*] : *qbs-space* (*monadP-qbs* X) = *monadP-qbs-Px* X
⟨*proof*⟩

lemma *monadP-qbs-Mx*[*simp*] : *qbs-Mx* (*monadP-qbs* X) = *monadP-qbs-MPx* X
⟨*proof*⟩

lemma *monadP-qbs-empty-iff*:
qbs-space $X = \{\} \longleftrightarrow$ *qbs-space* (*monadP-qbs* X) $= \{\}$
⟨*proof*⟩

If $\beta \in MPx$, there exists $X\ \alpha\ g$ s.t.$\beta\ r = [X,\alpha,g\ r]$. We define a function which picks $X\ \alpha\ g$ from $\beta \in MPx$.

definition *rep-monadP-qbs-MPx* :: (*real* $\Rightarrow$ *'a qbs-prob-space*) $\Rightarrow$ *'a quasi-borel* $\times$ (*real* $\Rightarrow$ *'a*) $\times$ (*real* $\Rightarrow$ *real measure*) **where**
rep-monadP-qbs-MPx $\beta \equiv$ *let* $X =$ *qbs-prob-space-qbs* (β *undefined*);
$\alpha g = ($*SOME* $k.$ (*fst* k) $\in$ *qbs-Mx* $X \wedge$ (*snd* k) $\in$ *real-borel* $\rightarrow_M$ *prob-algebra real-borel*

$\wedge\ \beta = (\lambda r.\ \textit{qbs-prob-space}\ (X,\textit{fst}\ k,\textit{snd}\ k\ r)))$
in $(X,\alpha g)$

lemma *qbs-prob-measure-measurable*[*measurable*]:
qbs-prob-measure ∈ *qbs-to-measure* (*monadP-qbs* (*X* :: *'a quasi-borel*)) $\to_M$ *prob-algebra* (*qbs-to-measure X*)
⟨*proof*⟩

lemma *qbs-l-inj*:
inj-on qbs-prob-measure (*monadP-qbs-Px X*)
⟨*proof*⟩

lemma *qbs-prob-measure-measurable'*[*measurable*]:
qbs-prob-measure ∈ *qbs-to-measure* (*monadP-qbs* (*X* :: *'a quasi-borel*)) $\to_M$ *sub-prob-algebra* (*qbs-to-measure X*)
⟨*proof*⟩

3.2.2 Return

definition *qbs-return* :: [*'a quasi-borel*, *'a*] ⇒ *'a qbs-prob-space* **where**
qbs-return X x ≡ *qbs-prob-space* (*X*,λ*r*. *x*,*Eps real-distribution*)

lemma(**in** *real-distribution*) *qbs-return-qbs-prob*:
assumes *x* ∈ *qbs-space X*
shows *qbs-prob X* (λ*r*. *x*) *M*
⟨*proof*⟩

lemma(**in** *real-distribution*) *qbs-return-computation* :
assumes *x* ∈ *qbs-space X*
shows *qbs-return X x* = *qbs-prob-space* (*X*,λ*r*. *x*,*M*)
⟨*proof*⟩

lemma *qbs-return-morphism*:
qbs-return X ∈ $X \to_Q$ *monadP-qbs X*
⟨*proof*⟩

lemma *qbs-return-morphism'*:
assumes *f* ∈ $X \to_Q Y$
shows (λ*x*. *qbs-return Y* (*f x*)) ∈ $X \to_Q$ *monadP-qbs Y*
⟨*proof*⟩

3.2.3 Bind

definition *qbs-bind* :: *'a qbs-prob-space* ⇒ (*'a* ⇒ *'b qbs-prob-space*) ⇒ *'b qbs-prob-space* **where**
qbs-bind s f ≡ (*let* (*qbsx*,α,μ) = *rep-qbs-prob-space s*;
(*qbsy*,β,*g*) = *rep-monadP-qbs-MPx* (*f* ∘ α)
in qbs-prob-space (*qbsy*,β,μ ⋙= *g*))

adhoc-overloading *Monad-Syntax.bind* ⇌ *qbs-bind*

lemma(**in** *qbs-prob*) *qbs-bind-computation*:
assumes$s = $ *qbs-prob-space* (X,α,μ)
$f \in X \rightarrow_Q$ *monadP-qbs* Y
$\beta \in$ *qbs-Mx* Y
and [*measurable*]: $g \in$ *real-borel* $\rightarrow_M$ *prob-algebra real-borel*
and $(f \circ \alpha) = (\lambda r.$ *qbs-prob-space* $(Y,\beta,\ g\ r))$
shows *qbs-prob* $Y\ \beta\ (\mu \gg= g)$
$s \gg= f = $ *qbs-prob-space* $(Y,\beta,\mu \gg= g)$
⟨*proof*⟩

lemma *qbs-bind-morphism′*:
assumes $f \in X \rightarrow_Q$ *monadP-qbs* Y
shows $(\lambda x.\ x \gg= f) \in$ *monadP-qbs* $X \rightarrow_Q$ *monadP-qbs* Y
⟨*proof*⟩

lemma *qbs-return-comp*:
assumes $\alpha \in$ *qbs-Mx* X
shows (*qbs-return* $X \circ \alpha) = (\lambda r.$ *qbs-prob-space* $(X,\alpha,$*return real-borel* $r))$
⟨*proof*⟩

lemma *qbs-bind-return′*:
assumes $x \in$ *monadP-qbs-Px* X
shows $x \gg=$ *qbs-return* $X = x$
⟨*proof*⟩

lemma *qbs-bind-return*:
assumes $f \in X \rightarrow_Q$ *monadP-qbs* Y
and $x \in$ *qbs-space* X
shows *qbs-return* $X\ x \gg= f = f\ x$
⟨*proof*⟩

lemma *qbs-bind-assoc*:
assumes $s \in$ *monadP-qbs-Px* X
$f \in X \rightarrow_Q$ *monadP-qbs* Y
and $g \in Y \rightarrow_Q$ *monadP-qbs* Z
shows $s \gg= (\lambda x.\ f\ x \gg= g) = (s \gg= f) \gg= g$
⟨*proof*⟩

lemma *qbs-bind-cong*:
assumes $s \in$ *monadP-qbs-Px* X
$\bigwedge x.\ x \in$ *qbs-space* $X \Longrightarrow f\ x = g\ x$
and $f \in X \rightarrow_Q$ *monadP-qbs* Y
shows $s \gg= f = s \gg= g$
⟨*proof*⟩

3.2.4 The Functorial Action $P(f)$

definition *monadP-qbs-Pf* :: [*'a quasi-borel*, *'b quasi-borel*,*'a* ⇒ *'b*,*'a qbs-prob-space*] ⇒ *'b qbs-prob-space* **where**
monadP-qbs-Pf - Y f sx ≡ *sx* ⪢= *qbs-return Y* ∘ *f*

lemma *monadP-qbs-Pf-morphism*:
  assumes $f \in X \rightarrow_Q Y$
  shows *monadP-qbs-Pf X Y f* ∈ *monadP-qbs X* $\rightarrow_Q$ *monadP-qbs Y*
  ⟨*proof*⟩

lemma(**in** *qbs-prob*) *monadP-qbs-Pf-computation*:
  assumes *s* = *qbs-prob-space* (*X*,α,μ)
    and $f \in X \rightarrow_Q Y$
   shows *qbs-prob Y* (*f* ∘ α) μ
    and *monadP-qbs-Pf X Y f s* = *qbs-prob-space* (*Y*,*f* ∘ α,μ)
  ⟨*proof*⟩

We show that P is a functor i.e. P preserves identity and composition.

lemma *monadP-qbs-Pf-id*:
  assumes *s* ∈ *monadP-qbs-Px X*
  shows *monadP-qbs-Pf X X id s* = *s*
  ⟨*proof*⟩

lemma *monadP-qbs-Pf-comp*:
  assumes *s* ∈ *monadP-qbs-Px X*
      $f \in X \rightarrow_Q Y$
    and $g \in Y \rightarrow_Q Z$
   shows ((*monadP-qbs-Pf Y Z g*) ∘ (*monadP-qbs-Pf X Y f*)) *s* = *monadP-qbs-Pf X Z* (*g* ∘ *f*) *s*
⟨*proof*⟩

3.2.5 Join

definition *qbs-join* :: *'a qbs-prob-space qbs-prob-space* ⇒ *'a qbs-prob-space* **where**
qbs-join ≡ (λ*sst*. *sst* ⪢= *id*)

lemma *qbs-join-morphism*:
  qbs-join ∈ *monadP-qbs* (*monadP-qbs X*) $\rightarrow_Q$ *monadP-qbs X*
  ⟨*proof*⟩

lemma *qbs-join-computation*:
  assumes *qbs-prob* (*monadP-qbs X*) β μ
      ssx = *qbs-prob-space* (*monadP-qbs X*,β,μ)
      α ∈ *qbs-Mx X*
      g ∈ *real-borel* $\rightarrow_M$ *prob-algebra real-borel*
    and β =(λ*r*. *qbs-prob-space* (*X*,α,*g r*))
   shows *qbs-prob X* α (μ ⪢= *g*) *qbs-join ssx* = *qbs-prob-space* (*X*,α, μ ⪢= *g*)
  ⟨*proof*⟩

3.2.6 Strength

definition *qbs-strength* :: $['a\ quasi\text{-}borel,'b\ quasi\text{-}borel,'a \times 'b\ qbs\text{-}prob\text{-}space] \Rightarrow ('a \times 'b)\ qbs\text{-}prob\text{-}space$ **where**
$qbs\text{-}strength\ W\ X = (\lambda(w,sx).\ let\ (\text{-},\alpha,\mu) = rep\text{-}qbs\text{-}prob\text{-}space\ sx$
$in\ qbs\text{-}prob\text{-}space\ (W \bigotimes_Q X,\ \lambda r.\ (w,\alpha\ r),\ \mu))$

lemma(**in** *qbs-prob*) *qbs-strength-computation*:
assumes $w \in qbs\text{-}space\ W$
and $sx = qbs\text{-}prob\text{-}space\ (X,\alpha,\mu)$
shows $qbs\text{-}prob\ (W \bigotimes_Q X)\ (\lambda r.\ (w,\alpha\ r))\ \mu$
$qbs\text{-}strength\ W\ X\ (w,sx) = qbs\text{-}prob\text{-}space\ (W \bigotimes_Q X,\ \lambda r.\ (w,\alpha\ r),\ \mu)$
⟨*proof*⟩

lemma *qbs-strength-natural*:
assumes $f \in X \rightarrow_Q X'$
$g \in Y \rightarrow_Q Y'$
$x \in qbs\text{-}space\ X$
and $sy \in monadP\text{-}qbs\text{-}Px\ Y$
shows $(monadP\text{-}qbs\text{-}Pf\ (X \bigotimes_Q Y)\ (X' \bigotimes_Q Y')\ (map\text{-}prod\ f\ g) \circ qbs\text{-}strength\ X\ Y)\ (x,sy) = (qbs\text{-}strength\ X'\ Y' \circ map\text{-}prod\ f\ (monadP\text{-}qbs\text{-}Pf\ Y\ Y'\ g))\ (x,sy)$
(**is** *?lhs* = *?rhs*)
⟨*proof*⟩

lemma *qbs-strength-ab-r*:
assumes $\alpha \in qbs\text{-}Mx\ X$
$\beta \in monadP\text{-}qbs\text{-}MPx\ Y$
$\gamma \in qbs\text{-}Mx\ Y$
and $[measurable]$:$g \in real\text{-}borel \rightarrow_M prob\text{-}algebra\ real\text{-}borel$
and $\beta = (\lambda r.\ qbs\text{-}prob\text{-}space\ (Y,\gamma,g\ r))$
shows $qbs\text{-}prob\ (X \bigotimes_Q Y)\ (map\text{-}prod\ \alpha\ \gamma \circ real\text{-}real.g)\ (distr\ (return\ real\text{-}borel\ r \bigotimes_M g\ r)\ real\text{-}borel\ real\text{-}real.f)$
$qbs\text{-}strength\ X\ Y\ (\alpha\ r,\ \beta\ r) = qbs\text{-}prob\text{-}space\ (X \bigotimes_Q Y,\ map\text{-}prod\ \alpha\ \gamma \circ real\text{-}real.g,\ distr\ (return\ real\text{-}borel\ r \bigotimes_M g\ r)\ real\text{-}borel\ real\text{-}real.f)$
⟨*proof*⟩

lemma *qbs-strength-morphism*:
$qbs\text{-}strength\ X\ Y \in X \bigotimes_Q monadP\text{-}qbs\ Y \rightarrow_Q monadP\text{-}qbs\ (X \bigotimes_Q Y)$
⟨*proof*⟩

lemma *qbs-bind-morphism''*:
$(\lambda(f,x).\ x \gg= f) \in exp\text{-}qbs\ X\ (monadP\text{-}qbs\ Y) \bigotimes_Q (monadP\text{-}qbs\ X) \rightarrow_Q (monadP\text{-}qbs\ Y)$
⟨*proof*⟩

lemma *qbs-bind-morphism'''*:
$(\lambda f\ x.\ x \gg= f) \in exp\text{-}qbs\ X\ (monadP\text{-}qbs\ Y) \rightarrow_Q exp\text{-}qbs\ (monadP\text{-}qbs\ X)\ (monadP\text{-}qbs\ Y)$
⟨*proof*⟩

lemma *qbs-bind-morphism*:
assumes $f \in X \rightarrow_Q$ *monadP-qbs Y*
and $g \in X \rightarrow_Q$ *exp-qbs Y* (*monadP-qbs Z*)
shows ($\lambda x.\ f\ x \gg= g\ x$) $\in X \rightarrow_Q$ *monadP-qbs Z*
⟨*proof*⟩

lemma *qbs-bind-morphism''''*:
assumes $x \in$ *monadP-qbs-Px X*
shows ($\lambda f.\ x \gg= f$) $\in$ *exp-qbs X* (*monadP-qbs Y*) $\rightarrow_Q$ *monadP-qbs Y*
⟨*proof*⟩

lemma *qbs-strength-law1*:
assumes $x \in$ *qbs-space* (*unit-quasi-borel* $\bigotimes_Q$ *monadP-qbs X*)
shows *snd x* = (*monadP-qbs-Pf* (*unit-quasi-borel* $\bigotimes_Q$ *X*) *X snd* ∘ *qbs-strength unit-quasi-borel X*) *x*
⟨*proof*⟩

lemma *qbs-strength-law2*:
assumes $x \in$ *qbs-space* ((*X* $\bigotimes_Q$ *Y*) $\bigotimes_Q$ *monadP-qbs Z*)
shows (*qbs-strength X* (*Y* $\bigotimes_Q$ *Z*) ∘ (*map-prod id* (*qbs-strength Y Z*)) ∘ ($\lambda((x,y),z).\ (x,(y,z))$)) *x* =
(*monadP-qbs-Pf* ((*X* $\bigotimes_Q$ *Y*) $\bigotimes_Q$ *Z*) (*X* $\bigotimes_Q$ (*Y* $\bigotimes_Q$ *Z*)) ($\lambda((x,y),z).\ (x,(y,z))$) ∘ *qbs-strength* (*X* $\bigotimes_Q$ *Y*) *Z*) *x*
(**is** *?lhs* = *?rhs*)
⟨*proof*⟩

lemma *qbs-strength-law3*:
assumes $x \in$ *qbs-space* (*X* $\bigotimes_Q$ *Y*)
shows *qbs-return* (*X* $\bigotimes_Q$ *Y*) *x* = (*qbs-strength X Y* ∘ (*map-prod id* (*qbs-return Y*))) *x*
⟨*proof*⟩

lemma *qbs-strength-law4*:
assumes $x \in$ *qbs-space* (*X* $\bigotimes_Q$ *monadP-qbs* (*monadP-qbs Y*))
shows (*qbs-strength X Y* ∘ *map-prod id qbs-join*) *x* = (*qbs-join* ∘ *monadP-qbs-Pf* (*X* $\bigotimes_Q$ *monadP-qbs Y*) (*monadP-qbs* (*X* $\bigotimes_Q$ *Y*))(*qbs-strength X Y*) ∘ *qbs-strength X* (*monadP-qbs Y*)) *x*
(**is** *?lhs* = *?rhs*)
⟨*proof*⟩

lemma *qbs-return-Mxpair*:
assumes $\alpha \in$ *qbs-Mx X*
and $\beta \in$ *qbs-Mx Y*
shows *qbs-return* (*X* $\bigotimes_Q$ *Y*) (α *r*, β *k*) = *qbs-prob-space* (*X* $\bigotimes_Q$ *Y*,*map-prod* α β ∘ *real-real.g*, *distr* (*return real-borel r* $\bigotimes_M$ *return real-borel k*) *real-borel real-real.f*)
qbs-prob (*X* $\bigotimes_Q$ *Y*) (*map-prod* α β ∘ *real-real.g*) (*distr* (*return real-borel*

$r \bigotimes_M$ *return real-borel k*) *real-borel real-real.f*)
⟨*proof*⟩

lemma *pair-return-return*:
  assumes $l \in$ *space M*
    and $r \in$ *space N*
   shows *return M l* $\bigotimes_M$ *return N r* = *return* ($M \bigotimes_M N$) (l,r)
⟨*proof*⟩

lemma *bind-bind-return-distr*:
  assumes *real-distribution* μ
    and *real-distribution* ν
   shows $\mu \gg= (\lambda r.\ \nu \gg= (\lambda l.$ *distr* (*return real-borel r* $\bigotimes_M$ *return real-borel l*) *real-borel real-real.f*))
      = *distr* ($\mu \bigotimes_M \nu$) *real-borel real-real.f*
   (**is** *?lhs* = *?rhs*)
⟨*proof*⟩

lemma(**in** *pair-qbs-probs*) *qbs-bind-return-qp*:
  shows *qbs-prob-space* (Y, β, ν) $\gg=$ (λy. *qbs-prob-space* (X, α, μ) $\gg=$ (λx. *qbs-return* ($X \bigotimes_Q Y$) (x,y))) = *qbs-prob-space* ($X \bigotimes_Q Y$, *map-prod* α β $\circ$ *real-real.g*, *distr* ($\mu \bigotimes_M \nu$) *real-borel real-real.f*)
    qbs-prob ($X \bigotimes_Q Y$) (*map-prod* α β $\circ$ *real-real.g*) (*distr* ($\mu \bigotimes_M \nu$) *real-borel real-real.f*)
⟨*proof*⟩

lemma(**in** *pair-qbs-probs*) *qbs-bind-return-pq*:
  shows *qbs-prob-space* (X, α, μ) $\gg=$ (λx. *qbs-prob-space* (Y, β, ν) $\gg=$ (λy. *qbs-return* ($X \bigotimes_Q Y$) (x,y))) = *qbs-prob-space* ($X \bigotimes_Q Y$, *map-prod* α β $\circ$ *real-real.g*, *distr* ($\mu \bigotimes_M \nu$) *real-borel real-real.f*)
    qbs-prob ($X \bigotimes_Q Y$) (*map-prod* α β $\circ$ *real-real.g*) (*distr* ($\mu \bigotimes_M \nu$) *real-borel real-real.f*)
⟨*proof*⟩

lemma *qbs-bind-return-rotate*:
  assumes $p \in$ *monadP-qbs-Px X*
    and $q \in$ *monadP-qbs-Px Y*
   shows $q \gg= (\lambda y.\ p \gg= (\lambda x.$ *qbs-return* ($X \bigotimes_Q Y$) (x,y))) = $p \gg= (\lambda x.\ q \gg= (\lambda y.$ *qbs-return* ($X \bigotimes_Q Y$) (x,y)))
⟨*proof*⟩

lemma *qbs-pair-bind-return1*:
  assumes $f \in X \bigotimes_Q Y \rightarrow_Q$ *monadP-qbs Z*
      $p \in$ *monadP-qbs-Px X*
    and $q \in$ *monadP-qbs-Px Y*
   shows $q \gg= (\lambda y.\ p \gg= (\lambda x.\ f\ (x,y)))$ = ($q \gg= (\lambda y.\ p \gg= (\lambda x.$ *qbs-return* ($X \bigotimes_Q Y$) (x,y)))) $\gg= f$
      (**is** *?lhs* = *?rhs*)

⟨*proof*⟩

lemma *qbs-pair-bind-return2*:
assumes $f \in X \bigotimes_Q Y \to_Q$ *monadP-qbs* Z
$p \in$ *monadP-qbs-Px* X
and $q \in$ *monadP-qbs-Px* Y
shows $p \gg= (\lambda x.\ q \gg= (\lambda y.\ f\ (x,y))) = (p \gg= (\lambda x.\ q \gg= (\lambda y.$ *qbs-return* $(X \bigotimes_Q Y)\ (x,y)))) \gg= f$
(**is** *?lhs* = *?rhs*)
⟨*proof*⟩

lemma *qbs-bind-rotate*:
assumes $f \in X \bigotimes_Q Y \to_Q$ *monadP-qbs* Z
$p \in$ *monadP-qbs-Px* X
and $q \in$ *monadP-qbs-Px* Y
shows $q \gg= (\lambda y.\ p \gg= (\lambda x.\ f\ (x,y))) = p \gg= (\lambda x.\ q \gg= (\lambda y.\ f\ (x,y)))$
⟨*proof*⟩

lemma(**in** *pair-qbs-probs*) *qbs-bind-bind-return*:
assumes $f \in X \bigotimes_Q Y \to_Q Z$
shows *qbs-prob* Z $(f \circ$ (*map-prod* $\alpha\ \beta \circ$ *real-real.g*)) (*distr* $(\mu \bigotimes_M \nu)$ *real-borel* *real-real.f*)
and *qbs-prob-space* $(X,\alpha,\mu) \gg= (\lambda x.$ *qbs-prob-space* $(Y,\beta,\nu) \gg= (\lambda y.$ *qbs-return* $Z\ (f\ (x,y)))) =$ *qbs-prob-space* $(Z,f \circ$ (*map-prod* $\alpha\ \beta \circ$ *real-real.g*),*distr* $(\mu \bigotimes_M \nu)$ *real-borel real-real.f*)
(**is** *?lhs* = *?rhs*)
⟨*proof*⟩

3.2.7 Properties of Return and Bind

lemma *qbs-prob-measure-return*:
assumes $x \in$ *qbs-space* X
shows *qbs-prob-measure* (*qbs-return* $X\ x$) = *return* (*qbs-to-measure* X) x
⟨*proof*⟩

lemma *qbs-prob-measure-bind*:
assumes $s \in$ *monadP-qbs-Px* X
and $f \in X \to_Q$ *monadP-qbs* Y
shows *qbs-prob-measure* $(s \gg= f)$ = *qbs-prob-measure* $s \gg=$ *qbs-prob-measure* $\circ f$
(**is** *?lhs* = *?rhs*)
⟨*proof*⟩

lemma *qbs-of-return*:
assumes $x \in$ *qbs-space* X
shows *qbs-prob-space-qbs* (*qbs-return* $X\ x$) = X
⟨*proof*⟩

lemma *qbs-of-bind*:
assumes $s \in$ *monadP-qbs-Px* X
and $f \in X \to_Q$ *monadP-qbs* Y
shows *qbs-prob-space-qbs* $(s \gg= f) = Y$
⟨*proof*⟩

3.2.8 Properties of Integrals

lemma *qbs-integrable-return*:
assumes $x \in$ *qbs-space* X
and $f \in X \to_Q \mathbb{R}_Q$
shows *qbs-integrable* (*qbs-return* X x) f
⟨*proof*⟩

lemma *qbs-integrable-bind-return*:
assumes $s \in$ *monadP-qbs-Px* Y
$f \in Z \to_Q \mathbb{R}_Q$
and $g \in Y \to_Q Z$
shows *qbs-integrable* $(s \gg= (\lambda y.$ *qbs-return* Z $(g\ y)))$ f = *qbs-integrable* s $(f \circ g)$
⟨*proof*⟩

lemma *qbs-prob-ennintegral-morphism*:
assumes $L \in X \to_Q$ *monadP-qbs* Y
and $f \in X \to_Q$ *exp-qbs* Y $\mathbb{R}_{Q\geq 0}$
shows $(\lambda x.$ *qbs-prob-ennintegral* $(L\ x)\ (f\ x)) \in X \to_Q \mathbb{R}_{Q\geq 0}$
⟨*proof*⟩

lemma *qbs-morphism-ennintegral-fst*:
assumes $q \in$ *monadP-qbs-Px* Y
and $f \in X \bigotimes_Q Y \to_Q \mathbb{R}_{Q\geq 0}$
shows $(\lambda x. \int^{+}{}_Q\ y.\ f\ (x, y)\ \partial q) \in X \to_Q \mathbb{R}_{Q\geq 0}$
⟨*proof*⟩

lemma *qbs-morphism-ennintegral-snd*:
assumes $p \in$ *monadP-qbs-Px* X
and $f \in X \bigotimes_Q Y \to_Q \mathbb{R}_{Q\geq 0}$
shows $(\lambda y. \int^{+}{}_Q\ x.\ f\ (x, y)\ \partial p) \in Y \to_Q \mathbb{R}_{Q\geq 0}$
⟨*proof*⟩

lemma *qbs-prob-ennintegral-morphism'*:
assumes $f \in X \to_Q \mathbb{R}_{Q\geq 0}$
shows $(\lambda s.$ *qbs-prob-ennintegral* s $f) \in$ *monadP-qbs* $X \to_Q \mathbb{R}_{Q\geq 0}$
⟨*proof*⟩

lemma *qbs-prob-ennintegral-return*:
assumes $f \in X \to_Q \mathbb{R}_{Q\geq 0}$
and $x \in$ *qbs-space* X

shows *qbs-prob-ennintegral* (*qbs-return X x*) $f = f\ x$
⟨*proof*⟩

lemma *qbs-prob-ennintegral-bind*:
assumes $s \in$ *monadP-qbs-Px X*
$f \in X \to_Q$ *monadP-qbs Y*
and $g \in Y \to_Q \mathbb{R}_{Q\geq 0}$
shows *qbs-prob-ennintegral* ($s \gg= f$) g = *qbs-prob-ennintegral* s (λy. (*qbs-prob-ennintegral* ($f\ y$) g))
(**is** *?lhs* = *?rhs*)
⟨*proof*⟩

lemma *qbs-prob-ennintegral-bind-return*:
assumes $s \in$ *monadP-qbs-Px Y*
$f \in Z \to_Q \mathbb{R}_{Q\geq 0}$
and $g \in Y \to_Q Z$
shows *qbs-prob-ennintegral* ($s \gg=$ (λy. *qbs-return* Z ($g\ y$))) f = *qbs-prob-ennintegral* s ($f \circ g$)
⟨*proof*⟩

lemma *qbs-prob-integral-morphism'*:
assumes $f \in X \to_Q \mathbb{R}_Q$
shows (λs. *qbs-prob-integral* $s\ f$) $\in$ *monadP-qbs X* $\to_Q \mathbb{R}_Q$
⟨*proof*⟩

lemma *qbs-morphism-integral-fst*:
assumes $q \in$ *monadP-qbs-Px Y*
and $f \in X \bigotimes_Q Y \to_Q \mathbb{R}_Q$
shows (λx. $\int_Q y$. f (x, y) ∂q) $\in X \to_Q \mathbb{R}_Q$
⟨*proof*⟩

lemma *qbs-morphism-integral-snd*:
assumes $p \in$ *monadP-qbs-Px X*
and $f \in X \bigotimes_Q Y \to_Q \mathbb{R}_Q$
shows (λy. $\int_Q x$. f (x, y) ∂p) $\in Y \to_Q \mathbb{R}_Q$
⟨*proof*⟩

lemma *qbs-prob-integral-morphism*:
assumes $L \in X \to_Q$ *monadP-qbs Y*
$f \in X \to_Q$ *exp-qbs Y* $\mathbb{R}_Q$
and $\bigwedge x$. $x \in$ *qbs-space X* $\Longrightarrow$ *qbs-integrable* ($L\ x$) ($f\ x$)
shows (λx. *qbs-prob-integral* ($L\ x$) ($f\ x$)) $\in X \to_Q \mathbb{R}_Q$
⟨*proof*⟩

lemma *qbs-prob-integral-morphism''*:
assumes $f \in X \to_Q \mathbb{R}_Q$
and $L \in Y \to_Q$ *monadP-qbs X*
shows (λy. *qbs-prob-integral* ($L\ y$) f) $\in Y \to_Q \mathbb{R}_Q$
⟨*proof*⟩

lemma *qbs-prob-integral-return*:
  assumes $f \in X \rightarrow_Q \mathbb{R}_Q$
    and $x \in$ *qbs-space* X
  shows *qbs-prob-integral* (*qbs-return* X x) $f = f\,x$
  ⟨*proof*⟩

lemma *qbs-prob-integral-bind*:
  assumes $s \in$ *monadP-qbs-Px* X
    $f \in X \rightarrow_Q$ *monadP-qbs* Y
    $g \in Y \rightarrow_Q \mathbb{R}_Q$
    and $\exists K. \forall y \in$ *qbs-space* $Y. |g\ y| \le K$
  shows *qbs-prob-integral* ($s \gg= f$) g = *qbs-prob-integral* s ($\lambda y.$ (*qbs-prob-integral* ($f\ y$) g))
    (**is** *?lhs* = *?rhs*)
⟨*proof*⟩

lemma *qbs-prob-integral-bind-return*:
  assumes $s \in$ *monadP-qbs-Px* Y
    $f \in Z \rightarrow_Q \mathbb{R}_Q$
    and $g \in Y \rightarrow_Q Z$
  shows *qbs-prob-integral* ($s \gg=$ ($\lambda y.$ *qbs-return* Z ($g\ y$))) f = *qbs-prob-integral* s ($f \circ g$)
⟨*proof*⟩

lemma *qbs-prob-var-bind-return*:
  assumes $s \in$ *monadP-qbs-Px* Y
    $f \in Z \rightarrow_Q \mathbb{R}_Q$
    and $g \in Y \rightarrow_Q Z$
  shows *qbs-prob-var* ($s \gg=$ ($\lambda y.$ *qbs-return* Z ($g\ y$))) f = *qbs-prob-var* s ($f \circ g$)
⟨*proof*⟩

end

3.3 Binary Product Measure

theory *Pair-QuasiBorel-Measure*
  imports *Monad-QuasiBorel*
begin

3.3.1 Binary Product Measure

Special case of [1] Proposition 23 where $\Omega = \mathbb{R} \times \mathbb{R}$ and $X = X \times Y$. Let $[\alpha, \mu] \in P(X)$ and $[\beta, \nu] \in P(Y)$. $\alpha \times \beta$ is the α in Proposition 23.

definition *qbs-prob-pair-measure-t* :: [$'a$ *qbs-prob-t*, $'b$ *qbs-prob-t*] ⇒ ($'a \times 'b$) *qbs-prob-t* **where**
qbs-prob-pair-measure-t p q ≡ (*let* $(X,\alpha,\mu) = p$;
    $(Y,\beta,\nu) = q$ *in*

($X \bigotimes_Q Y$, *map-prod* α β $\circ$ *real-real.g*, *distr* ($\mu \bigotimes_M$ ν) *real-borel real-real.f*))

lift-definition *qbs-prob-pair-measure* :: ['*a qbs-prob-space*, '*b qbs-prob-space*] $\Rightarrow$ ('*a* $\times$ '*b*) *qbs-prob-space* (**infix** ‹$\bigotimes_{Qmes}$› *80*)
is *qbs-prob-pair-measure-t*
⟨*proof*⟩

lemma(**in** *pair-qbs-probs*) *qbs-prob-pair-measure-computation*:
(*qbs-prob-space* (X,α,μ)) $\bigotimes_{Qmes}$ (*qbs-prob-space* (Y,β,ν)) = *qbs-prob-space* (X $\bigotimes_Q$ Y, *map-prod* α β $\circ$ *real-real.g* , *distr* ($\mu \bigotimes_M \nu$) *real-borel real-real.f*)
qbs-prob ($X \bigotimes_Q Y$) (*map-prod* α β $\circ$ *real-real.g*) (*distr* ($\mu \bigotimes_M \nu$) *real-borel real-real.f*)
⟨*proof*⟩

lemma *qbs-prob-pair-measure-qbs*:
qbs-prob-space-qbs ($p \bigotimes_{Qmes} q$) = *qbs-prob-space-qbs* p $\bigotimes_Q$ *qbs-prob-space-qbs* q
⟨*proof*⟩

lemma(**in** *pair-qbs-probs*) *qbs-prob-pair-measure-measure*:
shows *qbs-prob-measure* (*qbs-prob-space* (X,α,μ) $\bigotimes_{Qmes}$ *qbs-prob-space* (Y,β,ν)) = *distr* ($\mu \bigotimes_M \nu$) (*qbs-to-measure* ($X \bigotimes_Q Y$)) (*map-prod* α β)
⟨*proof*⟩

lemma *qbs-prob-pair-measure-morphism*:
case-prod qbs-prob-pair-measure $\in$ *monadP-qbs* $X \bigotimes_Q$ *monadP-qbs* $Y \to_Q$ *monadP-qbs* ($X \bigotimes_Q Y$)
⟨*proof*⟩

lemma(**in** *pair-qbs-probs*) *qbs-prob-pair-measure-nnintegral*:
assumes $f \in X \bigotimes_Q Y \to_Q \mathbb{R}_{Q\geq 0}$
shows ($\int^+_Q$ z. f z ∂(*qbs-prob-space* (X,α,μ) $\bigotimes_{Qmes}$ *qbs-prob-space* (Y,β,ν))) = ($\int^+$ z. (f $\circ$ *map-prod* α β) z ∂($\mu \bigotimes_M \nu$))
(**is** *?lhs* = *?rhs*)
⟨*proof*⟩

lemma(**in** *pair-qbs-probs*) *qbs-prob-pair-measure-integral*:
assumes $f \in X \bigotimes_Q Y \to_Q \mathbb{R}_Q$
shows ($\int_Q$ z. f z ∂(*qbs-prob-space* (X,α,μ) $\bigotimes_{Qmes}$ *qbs-prob-space* (Y,β,ν))) = ($\int$ z. (f $\circ$ *map-prod* α β) z ∂($\mu \bigotimes_M \nu$))
(**is** *?lhs* = *?rhs*)
⟨*proof*⟩

lemma *qbs-prob-pair-measure-eq-bind*:
assumes $p \in$ *monadP-qbs-Px* X
and $q \in$ *monadP-qbs-Px* Y
shows $p \bigotimes_{Qmes} q = p \gg= (\lambda x.\ q \gg= (\lambda y.$ *qbs-return* ($X \bigotimes_Q Y$) (x,y)))
⟨*proof*⟩

3.3.2 Fubini Theorem

lemma *qbs-prob-ennintegral-Fubini-fst*:
assumes $p \in monadP\text{-}qbs\text{-}Px\ X$
$q \in monadP\text{-}qbs\text{-}Px\ Y$
and $f \in X \bigotimes_Q Y \to_Q \mathbb{R}_{Q\geq 0}$
shows $(\int^+_Q x.\ \int^+_Q y.\ f\ (x,y)\ \partial q\ \partial p) = (\int^+_Q z.\ f\ z\ \partial(p \bigotimes_{Qmes} q))$
(**is** $?lhs = ?rhs$)
⟨*proof*⟩

lemma *qbs-prob-ennintegral-Fubini-snd*:
assumes $p \in monadP\text{-}qbs\text{-}Px\ X$
$q \in monadP\text{-}qbs\text{-}Px\ Y$
and $f \in X \bigotimes_Q Y \to_Q \mathbb{R}_{Q\geq 0}$
shows $(\int^+_Q y.\ \int^+_Q x.\ f\ (x,y)\ \partial p\ \partial q) = (\int^+_Q x.\ f\ x\ \partial(p \bigotimes_{Qmes} q))$
(**is** $?lhs = ?rhs$)
⟨*proof*⟩

lemma *qbs-prob-ennintegral-indep1*:
assumes $p \in monadP\text{-}qbs\text{-}Px\ X$
and $f \in X \to_Q \mathbb{R}_{Q\geq 0}$
shows $(\int^+_Q z.\ f\ (fst\ z)\ \partial(p \bigotimes_{Qmes} q)) = (\int^+_Q x.\ f\ x\ \partial p)$
(**is** $?lhs = -$)
⟨*proof*⟩

lemma *qbs-prob-ennintegral-indep2*:
assumes $q \in monadP\text{-}qbs\text{-}Px\ Y$
and $f \in Y \to_Q \mathbb{R}_{Q\geq 0}$
shows $(\int^+_Q z.\ f\ (snd\ z)\ \partial(p \bigotimes_{Qmes} q)) = (\int^+_Q y.\ f\ y\ \partial q)$
(**is** $?lhs = -$)
⟨*proof*⟩

lemma *qbs-ennintegral-indep-mult*:
assumes $p \in monadP\text{-}qbs\text{-}Px\ X$
$q \in monadP\text{-}qbs\text{-}Px\ Y$
$f \in X \to_Q \mathbb{R}_{Q\geq 0}$
and $g \in Y \to_Q \mathbb{R}_{Q\geq 0}$
shows $(\int^+_Q z.\ f\ (fst\ z) * g\ (snd\ z)\ \partial(p \bigotimes_{Qmes} q)) = (\int^+_Q x.\ f\ x\ \partial p) * (\int^+_Q y.\ g\ y\ \partial q)$
(**is** $?lhs = ?rhs$)
⟨*proof*⟩

lemma(**in** *pair-qbs-probs*) *qbs-prob-pair-measure-integrable*:
assumes *qbs-integrable* $(qbs\text{-}prob\text{-}space\ (X,\alpha,\mu) \bigotimes_{Qmes} qbs\text{-}prob\text{-}space\ (Y,\beta,\nu))\ f$
shows $f \in X \bigotimes_Q Y \to_Q \mathbb{R}_Q$
integrable $(\mu \bigotimes_M \nu)\ (f \circ (map\text{-}prod\ \alpha\ \beta))$
⟨*proof*⟩

lemma(**in** *pair-qbs-probs*) *qbs-prob-pair-measure-integrable'*:
assumes $f \in X \bigotimes_Q Y \to_Q \mathbb{R}_Q$
and *integrable* $(\mu \bigotimes_M \nu)$ $(f \circ (map\text{-}prod\ \alpha\ \beta))$
shows *qbs-integrable* (*qbs-prob-space* $(X,\alpha,\mu) \bigotimes_{Qmes}$ *qbs-prob-space* (Y,β,ν))
f
⟨*proof*⟩

lemma *qbs-integrable-pair-swap*:
assumes *qbs-integrable* $(p \bigotimes_{Qmes} q)$ f
shows *qbs-integrable* $(q \bigotimes_{Qmes} p)$ $(\lambda(x,y).\ f\ (y,x))$
⟨*proof*⟩

lemma *qbs-integrable-pair1*:
assumes $p \in$ *monadP-qbs-Px* X
$q \in$ *monadP-qbs-Px* Y
$f \in X \bigotimes_Q Y \to_Q \mathbb{R}_Q$
qbs-integrable p $(\lambda x.\ \int_Q y.\ |f\ (x,y)|\ \partial q)$
and $\bigwedge x.\ x \in$ *qbs-space* $X \Longrightarrow$ *qbs-integrable* q $(\lambda y.\ f\ (x,y))$
shows *qbs-integrable* $(p \bigotimes_{Qmes} q)$ f
⟨*proof*⟩

lemma *qbs-integrable-pair2*:
assumes $p \in$ *monadP-qbs-Px* X
$q \in$ *monadP-qbs-Px* Y
$f \in X \bigotimes_Q Y \to_Q \mathbb{R}_Q$
qbs-integrable q $(\lambda y.\ \int_Q x.\ |f\ (x,y)|\ \partial p)$
and $\bigwedge y.\ y \in$ *qbs-space* $Y \Longrightarrow$ *qbs-integrable* p $(\lambda x.\ f\ (x,y))$
shows *qbs-integrable* $(p \bigotimes_{Qmes} q)$ f
⟨*proof*⟩

lemma *qbs-integrable-fst*:
assumes *qbs-integrable* $(p \bigotimes_{Qmes} q)$ f
shows *qbs-integrable* p $(\lambda x.\ \int_Q y.\ f\ (x,y)\ \partial q)$
⟨*proof*⟩

lemma *qbs-integrable-snd*:
assumes *qbs-integrable* $(p \bigotimes_{Qmes} q)$ f
shows *qbs-integrable* q $(\lambda y.\ \int_Q x.\ f\ (x,y)\ \partial p)$
⟨*proof*⟩

lemma *qbs-integrable-indep-mult*:
assumes *qbs-integrable* p f
and *qbs-integrable* q g
shows *qbs-integrable* $(p \bigotimes_{Qmes} q)$ $(\lambda x.\ f\ (fst\ x) * g\ (snd\ x))$
⟨*proof*⟩

lemma *qbs-integrable-indep1*:
assumes *qbs-integrable* p f
shows *qbs-integrable* $(p \bigotimes_{Qmes} q)$ $(\lambda x.\ f\ (fst\ x))$

⟨*proof*⟩

lemma *qbs-integrable-indep2*:
assumes *qbs-integrable q g*
shows *qbs-integrable* $(p \bigotimes_{Qmes} q)$ $(\lambda x.\ g\ (snd\ x))$
⟨*proof*⟩

lemma *qbs-prob-integral-Fubini-fst*:
assumes *qbs-integrable* $(p \bigotimes_{Qmes} q)$ f
shows $(\int_Q x.\ \int_Q y.\ f\ (x,y)\ \partial q\ \partial p) = (\int_Q z.\ f\ z\ \partial(p \bigotimes_{Qmes} q))$
(**is** *?lhs* = *?rhs*)
⟨*proof*⟩

lemma *qbs-prob-integral-Fubini-snd*:
assumes *qbs-integrable* $(p \bigotimes_{Qmes} q)$ f
shows $(\int_Q y.\ \int_Q x.\ f\ (x,y)\ \partial p\ \partial q) = (\int_Q z.\ f\ z\ \partial(p \bigotimes_{Qmes} q))$
(**is** *?lhs* = *?rhs*)
⟨*proof*⟩

lemma *qbs-prob-integral-indep1*:
assumes *qbs-integrable p f*
shows $(\int_Q z.\ f\ (fst\ z)\ \partial(p \bigotimes_{Qmes} q)) = (\int_Q x.\ f\ x\ \partial p)$
⟨*proof*⟩

lemma *qbs-prob-integral-indep2*:
assumes *qbs-integrable q g*
shows $(\int_Q z.\ g\ (snd\ z)\ \partial(p \bigotimes_{Qmes} q)) = (\int_Q y.\ g\ y\ \partial q)$
⟨*proof*⟩

lemma *qbs-prob-integral-indep-mult*:
assumes *qbs-integrable p f*
and *qbs-integrable q g*
shows $(\int_Q z.\ f\ (fst\ z) * g\ (snd\ z)\ \partial(p \bigotimes_{Qmes} q)) = (\int_Q x.\ f\ x\ \partial p) * (\int_Q y.\ g\ y\ \partial q)$
(**is** *?lhs* = *?rhs*)
⟨*proof*⟩

lemma *qbs-prob-var-indep-plus*:
assumes *qbs-integrable* $(p \bigotimes_{Qmes} q)$ f
qbs-integrable $(p \bigotimes_{Qmes} q)$ $(\lambda z.\ (f\ z)^2)$
qbs-integrable $(p \bigotimes_{Qmes} q)$ g
qbs-integrable $(p \bigotimes_{Qmes} q)$ $(\lambda z.\ (g\ z)^2)$
qbs-integrable $(p \bigotimes_{Qmes} q)$ $(\lambda z.\ (f\ z) * (g\ z))$
and $(\int_Q z.\ f\ z * g\ z\ \partial(p \bigotimes_{Qmes} q)) = (\int_Q z.\ f\ z\ \partial(p \bigotimes_{Qmes} q)) * (\int_Q z.\ g\ z\ \partial(p \bigotimes_{Qmes} q))$
shows *qbs-prob-var* $(p \bigotimes_{Qmes} q)$ $(\lambda z.\ f\ z + g\ z)$ = *qbs-prob-var* $(p \bigotimes_{Qmes} q)$ f + *qbs-prob-var* $(p \bigotimes_{Qmes} q)$ g
⟨*proof*⟩

lemma *qbs-prob-var-indep-plus'*:
assumes *qbs-integrable p f*
qbs-integrable p $(\lambda x.\ (f\ x)^2)$
qbs-integrable q g
and *qbs-integrable q* $(\lambda x.\ (g\ x)^2)$
shows *qbs-prob-var* $(p \bigotimes_{Qmes} q)$ $(\lambda z.\ f\ (fst\ z) + g\ (snd\ z))$ = *qbs-prob-var p f* + *qbs-prob-var q g*
⟨*proof*⟩

end

3.4 Measure as QBS Measure

theory *Measure-as-QuasiBorel-Measure*
imports *Pair-QuasiBorel-Measure*

begin

lemma *distr-id'*:
assumes *sets N = sets M*
$f \in N \to_M N$
and $\bigwedge x.\ x \in space\ N \Longrightarrow f\ x = x$
shows *distr N M f = N*
⟨*proof*⟩

Every probability measure on a standard Borel space can be represented as a measure on a quasi-Borel space [1], Proposition 23.

locale *standard-borel-prob-space = standard-borel P + p:prob-space P*
for *P* :: *'a measure*
begin

sublocale *qbs-prob measure-to-qbs P g distr P real-borel f*
⟨*proof*⟩

lift-definition *as-qbs-measure* :: *'a qbs-prob-space* **is**
(*measure-to-qbs P*, *g*, *distr P real-borel f*)
⟨*proof*⟩

lemma *as-qbs-measure-retract*:
assumes [*measurable*]:$a \in P \to_M$ *real-borel*
and [*measurable*]:$b \in$ *real-borel* $\to_M P$
and [*simp*]:$\bigwedge x.\ x \in space\ P \Longrightarrow (b \circ a)\ x = x$
shows *qbs-prob* (*measure-to-qbs P*) *b* (*distr P real-borel a*)
as-qbs-measure = qbs-prob-space (*measure-to-qbs P*, *b*, *distr P real-borel a*)
⟨*proof*⟩

lemma *measure-as-qbs-measure-qbs*:
qbs-prob-space-qbs as-qbs-measure = measure-to-qbs P

⟨*proof*⟩

lemma *measure-as-qbs-measure-image*:
as-qbs-measure ∈ *monadP-qbs-Px* (*measure-to-qbs P*)
⟨*proof*⟩

lemma *as-qbs-measure-as-measure*[*simp*]:
distr (*distr P real-borel f*) (*qbs-to-measure* (*measure-to-qbs P*)) *g* = *P*
⟨*proof*⟩

lemma *measure-as-qbs-measure-recover*:
qbs-prob-measure as-qbs-measure = *P*
⟨*proof*⟩

end

lemma(**in** *standard-borel*) *qbs-prob-measure-recover*:
assumes *q* ∈ *monadP-qbs-Px* (*measure-to-qbs M*)
shows *standard-borel-prob-space.as-qbs-measure* (*qbs-prob-measure q*) = *q*
⟨*proof*⟩

lemma(**in** *standard-borel-prob-space*) *ennintegral-as-qbs-ennintegral*:
assumes *k* ∈ *borel-measurable P*
shows ($\int^{+}_{Q}$ *x*. *k x* ∂*as-qbs-measure*) = ($\int^{+}$ *x*. *k x* ∂*P*)
⟨*proof*⟩

lemma(**in** *standard-borel-prob-space*) *integral-as-qbs-integral*:
($\int_{Q}$ *x*. *k x* ∂*as-qbs-measure*) = ($\int$ *x*. *k x* ∂*P*)
⟨*proof*⟩

lemma(**in** *standard-borel*) *measure-with-args-morphism*:
assumes [*measurable*]:μ ∈ *X* $\rightarrow_M$ *prob-algebra M*
shows *standard-borel-prob-space.as-qbs-measure* ∘ μ ∈ *measure-to-qbs X* $\rightarrow_Q$ *monadP-qbs* (*measure-to-qbs M*)
⟨*proof*⟩

lemma(**in** *standard-borel*) *measure-with-args-recover*:
assumes μ ∈ *space X* → *space* (*prob-algebra M*)
and *x* ∈ *space X*
shows *qbs-prob-measure* (*standard-borel-prob-space.as-qbs-measure* (μ *x*)) = μ *x*
⟨*proof*⟩

3.5 Example of Probability Measures

Probability measures on ℝ can be represented as probability measures on the quasi-Borel space ℝ.

3.5.1 Normal Distribution

definition *normal-distribution* :: *real* × *real* ⇒ *real measure* **where**
normal-distribution $\mu\sigma$ = (*if 0* < (*snd* $\mu\sigma$) *then density lborel* (λx. *ennreal* (*normal-density* (*fst* $\mu\sigma$) (*snd* $\mu\sigma$) *x*))
else return lborel 0)

lemma *normal-distribution-measurable*:
normal-distribution ∈ *real-borel* $\bigotimes_M$ *real-borel* $\rightarrow_M$ *prob-algebra real-borel*
⟨*proof*⟩

definition *qbs-normal-distribution* :: *real* ⇒ *real* ⇒ *real qbs-prob-space* **where**
qbs-normal-distribution ≡ *curry* (*standard-borel-prob-space.as-qbs-measure* ∘ *normal-distribution*)

lemma *qbs-normal-distribution-morphism*:
qbs-normal-distribution ∈ $\mathbb{R}_Q$ $\rightarrow_Q$ *exp-qbs* $\mathbb{R}_Q$ (*monadP-qbs* $\mathbb{R}_Q$)
⟨*proof*⟩

context
fixes μ σ :: *real*
assumes *sigma*:σ > *0*
begin

interpretation *n-dist*:*standard-borel-prob-space normal-distribution* (μ,σ)
⟨*proof*⟩

lemma *qbs-normal-distribution-def2*:
qbs-normal-distribution μ σ = *n-dist.as-qbs-measure*
⟨*proof*⟩

lemma *qbs-normal-distribution-integral*:
($\int_Q$ *x*. *f x* ∂ (*qbs-normal-distribution* μ σ)) = ($\int$ *x*. *f x* ∂ (*density lborel* (λx. *ennreal* (*normal-density* μ σ *x*))))
⟨*proof*⟩

lemma *qbs-normal-distribution-expectation*:
assumes *f* ∈ *real-borel* $\rightarrow_M$ *real-borel*
shows ($\int_Q$ *x*. *f x* ∂ (*qbs-normal-distribution* μ σ)) = ($\int$ *x*. *normal-density* μ σ *x* ∗ *f x* ∂ *lborel*)
⟨*proof*⟩

end

3.5.2 Uniform Distribution

definition *interval-uniform-distribution* :: *real* ⇒ *real* ⇒ *real measure* **where**
interval-uniform-distribution a b ≡ (*if a* < *b then uniform-measure lborel* {*a*<..<*b*}
else return lborel 0)

lemma *sets-interval-uniform-distribution*[*measurable-cong*]:
sets (*interval-uniform-distribution a b*) = *borel*
⟨*proof*⟩

lemma *interval-uniform-distribution-meaurable*:
(λr. *interval-uniform-distribution* (*fst r*) (*snd r*)) ∈ *real-borel* $\bigotimes_M$ *real-borel* $\rightarrow_M$ *prob-algebra real-borel*
⟨*proof*⟩

definition *qbs-interval-uniform-distribution* :: *real* ⇒ *real* ⇒ *real qbs-prob-space* **where**
qbs-interval-uniform-distribution ≡ *curry* (*standard-borel-prob-space.as-qbs-measure* ∘ (λr. *interval-uniform-distribution* (*fst r*) (*snd r*)))

lemma *qbs-interval-uniform-distribution-morphism*:
qbs-interval-uniform-distribution ∈ $\mathbb{R}_Q \rightarrow_Q$ *exp-qbs* $\mathbb{R}_Q$ (*monadP-qbs* $\mathbb{R}_Q$)
⟨*proof*⟩

context
fixes *a b* :: *real*
assumes *a-less-than-b*: $a < b$
begin

definition *ab-qbs-uniform-distribution* ≡ *qbs-interval-uniform-distribution a b*

interpretation *ab-u-dist*: *standard-borel-prob-space interval-uniform-distribution a b*
⟨*proof*⟩

lemma *qbs-interval-uniform-distribution-def2*:
ab-qbs-uniform-distribution = *ab-u-dist.as-qbs-measure*
⟨*proof*⟩

lemma *qbs-uniform-distribution-expectation*:
assumes $f \in \mathbb{R}_Q \rightarrow_Q \mathbb{R}_{Q\geq 0}$
shows ($\int^{+}_Q x.\ f\ x\ \partial$*ab-qbs-uniform-distribution*) = ($\int^{+} x \in \{a<..<b\}.\ f\ x$ ∂*lborel*) / $(b - a)$
(**is** *?lhs* = *?rhs*)
⟨*proof*⟩

end

3.5.3 Bernoulli Distribution

definition *qbs-bernoulli* :: *real* ⇒ *bool qbs-prob-space* **where**
qbs-bernoulli ≡ *standard-borel-prob-space.as-qbs-measure* ∘ (λx. *measure-pmf* (*bernoulli-pmf x*))

lemma *bernoulli-measurable*:
$(\lambda x.\ \textit{measure-pmf}\ (\textit{bernoulli-pmf}\ x)) \in \textit{real-borel} \rightarrow_M \textit{prob-algebra bool-borel}$
⟨*proof*⟩

lemma *qbs-bernoulli-morphism*:
$\textit{qbs-bernoulli} \in \mathbb{R}_Q \rightarrow_Q \textit{monadP-qbs}\ \mathbb{B}_Q$
⟨*proof*⟩

lemma *qbs-bernoulli-measure*:
$\textit{qbs-prob-measure}\ (\textit{qbs-bernoulli}\ p) = \textit{measure-pmf}\ (\textit{bernoulli-pmf}\ p)$
⟨*proof*⟩

context
fixes $p :: \textit{real}$
assumes *pgeq-0*[*simp*]:$0 \le p$ **and** *pleq-1*[*simp*]:$p \le 1$
begin

lemma *qbs-bernoulli-expectation*:
$(\int_Q x.\ f\ x\ \partial \textit{qbs-bernoulli}\ p) = f\ \textit{True} * p + f\ \textit{False} * (1 - p)$
⟨*proof*⟩

end

end

3.6 Bayesian Linear Regression

theory *Bayesian-Linear-Regression*
imports *Measure-as-QuasiBorel-Measure*
begin

We formalize the Bayesian linear regression presented in [1] section VI.

3.6.1 Prior

abbreviation $\nu \equiv \textit{density lborel}\ (\lambda x.\ \textit{ennreal}\ (\textit{normal-density}\ 0\ 3\ x))$

interpretation ν: *standard-borel-prob-space* ν
⟨*proof*⟩

term $\nu.\textit{as-qbs-measure} :: \textit{real qbs-prob-space}$
definition *prior* :: (*real* ⇒ *real*) *qbs-prob-space* **where**
$\textit{prior} \equiv \textit{do}\ \{\ s \leftarrow \nu.\textit{as-qbs-measure}\ ;$
$b \leftarrow \nu.\textit{as-qbs-measure}\ ;$
$\textit{qbs-return}\ (\mathbb{R}_Q \Rightarrow_Q \mathbb{R}_Q)\ (\lambda r.\ s * r + b)\}$

lemma ν*-as-qbs-measure-eq*:
$\nu.\textit{as-qbs-measure} = \textit{qbs-prob-space}\ (\mathbb{R}_Q, \textit{id}, \nu)$

⟨*proof*⟩

interpretation *ν-qp*: *pair-qbs-prob* $\mathbb{R}_Q$ *id* *ν* $\mathbb{R}_Q$ *id* *ν*
⟨*proof*⟩

lemma *ν-as-qbs-measure-in-Pr*:
ν.as-qbs-measure ∈ *monadP-qbs-Px* $\mathbb{R}_Q$
⟨*proof*⟩

lemma *sets-real-real-real*[*measurable-cong*]:
sets (*qbs-to-measure* (($\mathbb{R}_Q \bigotimes_Q \mathbb{R}_Q$) $\bigotimes_Q$ $\mathbb{R}_Q$)) = *sets* ((*borel* $\bigotimes_M$ *borel*) $\bigotimes_M$ *borel*)
⟨*proof*⟩

lemma *lin-morphism*:
(λ(*s*, *b*) *r*. *s* ∗ *r* + *b*) ∈ $\mathbb{R}_Q \bigotimes_Q \mathbb{R}_Q \to_Q \mathbb{R}_Q \Rightarrow_Q \mathbb{R}_Q$
⟨*proof*⟩

lemma *lin-measurable*[*measurable*]:
(λ(*s*, *b*) *r*. *s* ∗ *r* + *b*) ∈ *real-borel* $\bigotimes_M$ *real-borel* $\to_M$ *qbs-to-measure* ($\mathbb{R}_Q \Rightarrow_Q \mathbb{R}_Q$)
⟨*proof*⟩

lemma *prior-computation*:
qbs-prob ($\mathbb{R}_Q \Rightarrow_Q \mathbb{R}_Q$) ((λ(*s*, *b*) *r*. *s* ∗ *r* + *b*) ∘ *real-real.g*) (*distr* (*ν* $\bigotimes_M$ *ν*) *real-borel real-real.f*)
prior = *qbs-prob-space* ($\mathbb{R}_Q \Rightarrow_Q \mathbb{R}_Q$, (λ(*s*, *b*) *r*. *s* ∗ *r* + *b*) ∘ *real-real.g*, *distr* (*ν* $\bigotimes_M$ *ν*) *real-borel real-real.f*)
⟨*proof*⟩

The following lemma corresponds to the equation (5).

lemma *prior-measure*:
qbs-prob-measure prior = *distr* (*ν* $\bigotimes_M$ *ν*) (*qbs-to-measure* (*exp-qbs* $\mathbb{R}_Q$ $\mathbb{R}_Q$)) (λ(*s*,*b*) *r*. *s* ∗ *r* + *b*)
⟨*proof*⟩

lemma *prior-in-space*:
prior ∈ *qbs-space* (*monadP-qbs* ($\mathbb{R}_Q \Rightarrow_Q \mathbb{R}_Q$))
⟨*proof*⟩

3.6.2 Likelihood

abbreviation *d* *μ* *x* ≡ *normal-density* *μ* (*1/2*) *x*

lemma *d-positive* : *0* < *d* *μ* *x*
⟨*proof*⟩

definition *obs* :: (*real* ⇒ *real*) ⇒ *ennreal* **where**
obs f ≡ *d* (*f 1*) *2.5* ∗ *d* (*f 2*) *3.8* ∗ *d* (*f 3*) *4.5* ∗ *d* (*f 4*) *6.2* ∗ *d* (*f 5*) *8*

lemma *obs-morphism*:
$obs \in \mathbb{R}_Q \Rightarrow_Q \mathbb{R}_Q \rightarrow_Q \mathbb{R}_{Q\geq 0}$
⟨*proof*⟩

lemma *obs-measurable*[*measurable*]:
$obs \in$ *qbs-to-measure* (*exp-qbs* $\mathbb{R}_Q$ $\mathbb{R}_Q$) $\rightarrow_M$ *ennreal-borel*
⟨*proof*⟩

3.6.3 Posterior

lemma *id-obs-morphism*:
$(\lambda f.\ (f, obs\ f)) \in \mathbb{R}_Q \Rightarrow_Q \mathbb{R}_Q \rightarrow_Q (\mathbb{R}_Q \Rightarrow_Q \mathbb{R}_Q) \bigotimes_Q \mathbb{R}_{Q\geq 0}$
⟨*proof*⟩

lemma *push-forward-measure-in-space*:
monadP-qbs-Pf $(\mathbb{R}_Q \Rightarrow_Q \mathbb{R}_Q)$ $((\mathbb{R}_Q \Rightarrow_Q \mathbb{R}_Q) \bigotimes_Q \mathbb{R}_{Q\geq 0})$ $(\lambda f.\ (f, obs\ f))$ *prior* $\in$ *qbs-space* (*monadP-qbs* $((\mathbb{R}_Q \Rightarrow_Q \mathbb{R}_Q) \bigotimes_Q \mathbb{R}_{Q\geq 0})$)
⟨*proof*⟩

lemma *push-forward-measure-computation*:
qbs-prob $((\mathbb{R}_Q \Rightarrow_Q \mathbb{R}_Q) \bigotimes_Q \mathbb{R}_{Q\geq 0})$ $(\lambda l.\ (((\lambda(s,\ b)\ r.\ s * r + b) \circ real\text{-}real.g)\ l,\ ((obs \circ (\lambda(s,\ b)\ r.\ s * r + b)) \circ real\text{-}real.g)\ l))$ $(distr\ (\nu \bigotimes_M \nu)\ real\text{-}borel\ real\text{-}real.f)$
monadP-qbs-Pf $(\mathbb{R}_Q \Rightarrow_Q \mathbb{R}_Q)$ $((\mathbb{R}_Q \Rightarrow_Q \mathbb{R}_Q) \bigotimes_Q \mathbb{R}_{Q\geq 0})$ $(\lambda f.\ (f,\ obs\ f))$ *prior* = *qbs-prob-space* $((\mathbb{R}_Q \Rightarrow_Q \mathbb{R}_Q) \bigotimes_Q \mathbb{R}_{Q\geq 0},\ (\lambda l.\ (((\lambda(s,\ b)\ r.\ s * r + b) \circ real\text{-}real.g)\ l,\ ((obs \circ (\lambda(s,\ b)\ r.\ s * r + b)) \circ real\text{-}real.g)\ l)),\ distr\ (\nu \bigotimes_M \nu)\ real\text{-}borel\ real\text{-}real.f)$
⟨*proof*⟩

3.6.4 Normalizer

We use the unit space for an error.

definition *norm-qbs-measure* :: $('a \times ennreal)$ *qbs-prob-space* $\Rightarrow$ $'a$ *qbs-prob-space* + *unit* **where**
norm-qbs-measure p $\equiv$ (*let* $(XR, \alpha\beta, \nu)$ = *rep-qbs-prob-space p in*
if emeasure (*density* ν (*snd* $\circ$ $\alpha\beta$)) *UNIV* = *0 then Inr* ()
else if emeasure (*density* ν (*snd* $\circ$ $\alpha\beta$)) *UNIV* = ∞ *then Inr* ()
else Inl (*qbs-prob-space* (*map-qbs fst XR*, *fst* $\circ$ $\alpha\beta$, *density* ν ($\lambda r.$ *snd* ($\alpha\beta$ *r*) / *emeasure* (*density* ν (*snd* $\circ$ $\alpha\beta$)) *UNIV*))))

lemma *norm-qbs-measure-qbs-prob*:
assumes *qbs-prob* $(X \bigotimes_Q \mathbb{R}_{Q\geq 0})$ $(\lambda r.\ (\alpha\ r,\ \beta\ r))$ μ
emeasure (*density* μ β) *UNIV* $\neq$ *0*
and *emeasure* (*density* μ β) *UNIV* $\neq \infty$
shows *qbs-prob X* α (*density* μ ($\lambda r.$ (β *r*) / *emeasure* (*density* μ β) *UNIV*))
⟨*proof*⟩

lemma *norm-qbs-measure-computation*:
  assumes *qbs-prob* $(X \bigotimes_Q \mathbb{R}_{Q\geq 0})$ $(\lambda r.\ (\alpha\ r,\ \beta\ r))$ μ
  shows *norm-qbs-measure* (*qbs-prob-space* $(X \bigotimes_Q \mathbb{R}_{Q\geq 0}$, $(\lambda r.\ (\alpha\ r,\ \beta\ r))$, $\mu))$ = (*if emeasure* (*density* μ β) *UNIV* = *0 then Inr* ()
else if emeasure (*density* μ β) *UNIV* = ∞ *then Inr* ()
else Inl (*qbs-prob-space* (X, α, *density* μ $(\lambda r.\ (\beta\ r)$ / *emeasure* (*density* μ β) *UNIV*))))
⟨*proof*⟩

lemma *norm-qbs-measure-morphism*:
norm-qbs-measure $\in$ *monadP-qbs* $(X \bigotimes_Q \mathbb{R}_{Q\geq 0}) \to_Q$ *monadP-qbs* $X <+>_Q 1_Q$
⟨*proof*⟩

The following is the semantics of the entire program.

definition *program* :: (*real* $\Rightarrow$ *real*) *qbs-prob-space* + *unit* **where**
program $\equiv$ *norm-qbs-measure* (*monadP-qbs-Pf* $(\mathbb{R}_Q \Rightarrow_Q \mathbb{R}_Q)$ $((\mathbb{R}_Q \Rightarrow_Q \mathbb{R}_Q) \bigotimes_Q \mathbb{R}_{Q\geq 0})$ $(\lambda f.\ (f, obs\ f))$ *prior*)

lemma *program-in-space*:
program $\in$ *qbs-space* (*monadP-qbs* $(\mathbb{R}_Q \Rightarrow_Q \mathbb{R}_Q) <+>_Q 1_Q)$
⟨*proof*⟩

We calculate the normalizing constant.

lemma *complete-the-square*:
  fixes *a b c x* :: *real*
  assumes $a \neq 0$
  shows $a*x^2 + b*x + c = a*(x + (b\ /\ (2*a)))^2 - ((b^2 - 4*a*c)/(4*a))$
  ⟨*proof*⟩

lemma *complete-the-square2'*:
  fixes *a b c x* :: *real*
  assumes $a \neq 0$
  shows $a*x^2 - 2*b*x + c = a*(x - (b\ /\ a))^2 - ((b^2 - a*c)/a)$
  ⟨*proof*⟩

lemma *normal-density-mu-x-swap*:
  normal-density μ σ x = *normal-density* x σ μ
  ⟨*proof*⟩

lemma *normal-density-plus-shift*:
normal-density μ σ $(x + y)$ = *normal-density* $(\mu - x)$ σ y
  ⟨*proof*⟩

lemma *normal-density-times*:
  assumes $\sigma > 0$ $\sigma' > 0$
  shows *normal-density* μ σ x $*$ *normal-density* μ' σ' x = $(1\ /\ sqrt\ (2 * pi * (\sigma^2 + \sigma'^2))) * exp\ (-(\mu - \mu')^2\ /\ (2 * (\sigma^2 + \sigma'^2)))$ $*$ *normal-density* $((\mu*\sigma'^2 +$

$\mu'*\sigma^2)/(\sigma^2 + \sigma'^2))$ $(\sigma * \sigma' / sqrt (\sigma^2 + \sigma'^2))$ x
(**is** *?lhs = ?rhs*)
⟨*proof*⟩

lemma *normal-density-times'*:
assumes $\sigma > 0$ $\sigma' > 0$
shows $a * normal\text{-}density\ \mu\ \sigma\ x * normal\text{-}density\ \mu'\ \sigma'\ x = a * (1 / sqrt (2 * pi * (\sigma^2 + \sigma'^2))) * exp (- (\mu - \mu')^2 / (2 * (\sigma^2 + \sigma'^2))) * normal\text{-}density ((\mu*\sigma'^2 + \mu'*\sigma^2)/(\sigma^2 + \sigma'^2)) (\sigma * \sigma' / sqrt (\sigma^2 + \sigma'^2))\ x$
⟨*proof*⟩

lemma *normal-density-times-minusx*:
assumes $\sigma > 0$ $\sigma' > 0$ $a \neq a'$
shows $normal\text{-}density (\mu - a*x)\ \sigma\ y * normal\text{-}density (\mu' - a'*x)\ \sigma'\ y = (1 / |a' - a|) * normal\text{-}density ((\mu' - \mu)/(a' - a)) (sqrt ((\sigma^2 + \sigma'^2)/(a' - a)^2))\ x * normal\text{-}density (((\mu - a*x)*\sigma'^2 + (\mu' - a'*x)*\sigma^2)/(\sigma^2 + \sigma'^2)) (\sigma * \sigma' / sqrt (\sigma^2 + \sigma'^2))\ y$
⟨*proof*⟩

The following is the normalizing constant of the program.

abbreviation $C \equiv ennreal ((4 * sqrt\ 2 / (pi^2 * sqrt (66961 * pi))) * (exp (-(1674761 / 1674025))))$

lemma *program-normalizing-constant*:
$emeasure (density (distr (\nu \bigotimes_M \nu)\ real\text{-}borel\ real\text{-}real.f) (obs \circ (\lambda(s, b)\ r.\ s * r + b) \circ real\text{-}real.g))\ UNIV = C$
(**is** *?lhs = ?rhs*)
⟨*proof*⟩

The program returns a probability measure, rather than error.

lemma *program-result*:
$qbs\text{-}prob (\mathbb{R}_Q \Rightarrow_Q \mathbb{R}_Q) ((\lambda(s, b)\ r.\ s * r + b) \circ real\text{-}real.g) (density (distr (\nu \bigotimes_M \nu)\ real\text{-}borel\ real\text{-}real.f) (\lambda r. (obs \circ (\lambda(s, b)\ r.\ s * r + b) \circ real\text{-}real.g)\ r / C))$
$program = Inl (qbs\text{-}prob\text{-}space (\mathbb{R}_Q \Rightarrow_Q \mathbb{R}_Q, (\lambda(s, b)\ r.\ s * r + b) \circ real\text{-}real.g, density (distr (\nu \bigotimes_M \nu)\ real\text{-}borel\ real\text{-}real.f) (\lambda r. (obs \circ (\lambda(s, b)\ r.\ s * r + b) \circ real\text{-}real.g)\ r / C)))$
⟨*proof*⟩

lemma *program-inl*:
$program \in Inl$ ‘ $(qbs\text{-}space (monadP\text{-}qbs (\mathbb{R}_Q \Rightarrow_Q \mathbb{R}_Q)))$
⟨*proof*⟩

lemma *program-result-measure*:
$qbs\text{-}prob\text{-}measure (qbs\text{-}prob\text{-}space (\mathbb{R}_Q \Rightarrow_Q \mathbb{R}_Q, (\lambda(s, b)\ r.\ s * r + b) \circ real\text{-}real.g, density (distr (\nu \bigotimes_M \nu)\ real\text{-}borel\ real\text{-}real.f) (\lambda r. (obs \circ (\lambda(s, b)\ r.\ s * r + b) \circ real\text{-}real.g)\ r / C)))$
$= density (qbs\text{-}prob\text{-}measure\ prior) (\lambda k.\ obs\ k / C)$
(**is** *?lhs = ?rhs*)
⟨*proof*⟩

lemma *program-result-measure'*:
qbs-prob-measure (*qbs-prob-space* (*exp-qbs* $\mathbb{R}_Q$ $\mathbb{R}_Q$, (λ(*s*, *b*) *r*. *s* $*$ *r* + *b*) $\circ$ *real-real.g*, *density* (*distr* (ν $\bigotimes_M$ ν) *real-borel real-real.f*) (λ*r*. (*obs* $\circ$ (λ(*s*, *b*) *r*. *s* $*$ *r* + *b*) $\circ$ *real-real.g*) *r* / *C*)))
= *distr* (*density* (ν $\bigotimes_M$ ν) (λ(*s*,*b*). *obs* (λ*r*. *s* $*$ *r* + *b*) / *C*)) (*qbs-to-measure* (*exp-qbs* $\mathbb{R}_Q$ $\mathbb{R}_Q$)) (λ(*s*, *b*) *r*. *s* $*$ *r* + *b*)
⟨*proof*⟩

end

References

[1] C. Heunen, O. Kammar, S. Staton, and H. Yang. A convenient category for higher-order probability theory. In *Proceedings of the 32nd Annual ACM/IEEE Symposium on Logic in Computer Science*, LICS '17. IEEE Press, 2017.