論文 / 著書情報
Article / Book Information

題目(和文)	高階関数を扱う確率論基礎の形式化
Title(English)	Formalized Foundations for Higher-Order Probability Theory
著者(和文)	平田路和
Author(English)	Michikazu Hirata
出典(和文)	学位:博士(理学), 学位授与機関:東京科学大学, 報告番号:甲第235号, 授与年月日:2025年3月26日, 学位の種別:課程博士, 審査員:南出 靖彦,荒井 迅,増原 英彦,三好 直人,脇田 建,Affeldt Reynald
Citation(English)	Degree:Doctor (Science), Conferring organization: Institute of Science Tokyo, Report number:甲第235号, Conferred date:2025/3/26, Degree Type:Course doctor, Examiner:,,,,,
学位種別(和文)	博士論文
Type(English)	Doctoral Thesis

Institute of Science Tokyo
Department of Mathematical and Computing Science

Formalized Foundations for Higher-Order Probability Theory

Michikazu Hirata

Supervised by Prof. Yasuhiko Minamide

A doctoral thesis submitted to Institute of Science Tokyo.

February 7, 2025

Abstract

Formalization of probability theory plays important roles as a foundation of formal verification for systems with probabilistic behaviors such as probabilistic programs, stochastic processes, and machine learning algorithms. Although basic probability theory has already been formalized, existing library does not cover advanced theories used in some latest researches.

This thesis aims to formalize advanced topics of probability theory in the interactive theorem prover Isabelle/HOL. Especially, we focus on the basis of higher-order probability theory. We formalize the three theories: standard Borel spaces, the Lévy-Prokhorov metric, and quasi-Borel spaces. The theory of standard Borel spaces is often used in applied probability theory and theoretical basis of quasi-Borel spaces. We formalize standard Borel spaces and prove Kuratowski's theorem: any standard Borel space is either a countable discrete space or isomorphic to $\mathbb{R}$. The Lévy-Prokhorov metric is a metric between finite measures on a metric space. The metric was introduced to analyze weak convergence of measures. We formalize the Lévy-Prokhorov metric and prove Prokhorov's theorem, giving an equivalent condition for the relative compactness of sets of finite measures, and the Riesz representation theorem, a theorem in functional analysis. The theory of quasi-Borel spaces is a new denotational model for higher-order probabilistic programs. We formalize quasi-Borel spaces and the s-finite measure monad on it. We also apply our formalization to verify probabilistic programs, including the Monte Carlo method and the Gaussian mean learning algorithm, and differential privacy.

Acknowledgement

I would firstly like to thank my supervisor, Professor Yasuhiko Minamide, for giving me the opportunity to work in his research group and continued encouragement and support. He has provided a number of helpful pieces of advice, valuable comments, and suggestions. I would also like to thank Dr. Tetsuya Sato and all other members in Minamide's research group. I have benefited greatly from Dr. Tetsuya Sato's comments and deep knowledge of probability theory and category theory. I would also like to thank my friends for their encouragement, support, and sometimes having discussions about technical topics. I would also express my gratitude to my family for their continued encouragement, support, and delicious meals.

Contents

Chapter 1

Introduction

Formalization of mathematics is one of the important foundations of formal verification with interactive theorem provers because formal methods rely on mathematics such as discrete mathematics, algebra, analysis, and category theory. Probability theory is a fundamental tool to analyze probabilistic programs, stochastic processes, and machine learning algorithms. The semantics of probabilistic programs is naturally based on probability theory. The semantics is applied to verify the implementation of probabilistic programming languages and reason about probabilistic programs. Stochastic processes are usually used to model probabilistic systems whose state changes depending on time. Probability theory is also applied to evaluate machine learning algorithms by assuming that data are obtained from probability distributions.

The theory of higher-order probabilistic programs has been an actively studied research area. The word "higher-order probabilistic programs" means that probabilistic programs supporting higher-order functions, i.e., the program can receive functions as arguments. In general, the semantics of probabilistic programs is based on measurable spaces and measurable functions. However, semantics based on measure theory faces a technical difficulty when we try to denote higher-order programs. It is known that there is no σ-algebra on $\mathbb{R}^{\mathbb{R}}$ (the set of all measurable functions from $\mathbb{R}$ to $\mathbb{R}$) that makes the evaluation function $\mathrm{ev}\colon \mathbb{R}^{\mathbb{R}} \times \mathbb{R} \to \mathbb{R}$, $\mathrm{ev}(f, x) = f(x)$ measurable [3]. This result suggests that the standard semantics cannot treat higher-order programs. In order to overcome this difficulty, Heunen et al. [25] introduced the notion of quasi-Borel spaces. The theory enables us to denote higher-order probabilistic programs because function spaces always exist and the s-finite measure monad on quasi-Borel spaces is used to denote the type of probability distributions. Although the construction of the s-finite measure is non-trivial, no paper provides its detailed proofs. In addition, the definitions of the monad vary among prior studies [44, 52, 56]. Those situations make it difficult to use the theory and check the correctness of the research results.

Basic probability theory (and also measure theory) has already been formalized in major interactive theorem provers, e.g., Coq [1, 12], Isabelle/HOL [4, 8, 18, 26], and Lean [50, 53]. However, the existing library still does not cover advanced theories used in some recent research works, such as the theory of quasi-Borel spaces. It is also sometimes difficult to check the correctness of mathematical statements in new research results because the proof is complicated or omitted even for non-trivial statements. Hence, formalizing advanced probability theories plays important roles for studies related to probability theory because we can ensure the correctness of the theory and easily check that theorems are valid by just executing proof

scripts without reading complicated and lengthy proofs by ourselves.

This thesis aims to formalize advanced topics of probability theory in the interactive theorem prover Isabelle/HOL. Especially, we focus on the basis of higher-order probability theory. We formalize the three theories: standard Borel spaces, the Lévy-Prokhorov metric, and quasi-Borel spaces. We also verify probabilistic programs using quasi-Borel spaces.

Standard Borel spaces Standard Borel spaces are a certain class of measurable spaces. Standard Borel spaces are defined through Polish spaces, which are a class of topological spaces. Those spaces are often used in applied probability theory and statistics area, including the theory of quasi-Borel spaces, because they have good properties: any Polish space is embedded into a compact space (Lemma 3.6) and any standard Borel space is either countable discrete space or isomorphic to $\mathbb{R}$ (Kuratowski's theorem, Theorem 3.12). Although those spaces are commonly used, they were not formalized until these days. Recently, Gouëzel formalized Polish and standard Borel spaces in Lean [50, 21]. Isabelle/HOL's standard library also has a formalization of the Polish spaces based on type classes. However, their formalization has restrictions that it cannot treat the cases where there is no natural metric on the type or the natural topology on the type does not form a Polish space.

In this thesis, we formalize the notion of Polish spaces and standard Borel spaces. We also prove their useful properties: the fact that any Polish space is embedded into a compact space (Lemma 3.6), and Kuratowski's theorem (Theorem 3.12). Those theorems are used later in this thesis. We finally apply Kuratowski's theorem to prove the disintegration theorem, which ensures the existence of a *conditional probability kernel*.

Compared to the existing Isabelle/HOL's formalization of Polish spaces, our definition can treat Polish spaces with any career sets and include advanced theorems. In general, a Polish space is defined as a separable completely metrizable space, while the existing formalization defines the Polish spaces as separable complete metric spaces using type classes. The existing formalization cannot treat subspaces directly because type classes cannot do so. In addition, their definition requires us to define a metric on the set. Hence, it is not suitable to interpret a type as a Polish space when there is no natural metric on the type, e.g., $\mathbb{R} \cup \{\infty, -\infty\}$, or the natural metric is not complete. Our formalization does not have those restrictions because we define Polish spaces in the same manner as the usual mathematical definition using topological spaces defined by type definition (see Section 2.3.3) in Isabelle/HOL.

Lévy-Prokhorov metric Although this theory is independent of higher-order probability theory, the formalization of the theory includes important results such as the Riesz representation theorem and Prokhorov's theorem. The Lévy-Prokhorov metric is a mathematical tool to analyze asymptotic behaviors of distributions or measures in terms of weak convergence. Such analysis is one of the important aspects of probability theory and a foundation of statistics because the knowledge on asymptotic behaviors provides insights of what will be likely to happen when we collect large data. One of the important consequence related to the Lévy-Prokhorov metric is Prokhorov's theorem. The theorem provides a condition for the relative compactness of sets of finite measures: a set of (uniformly bounded) finite measures is relatively compact if and only if it is *tight*. The theorem plays essential roles in proofs of the central limit theorem, Sanov's theorem in large deviation theory, and the existence of optimal coupling in transportation theory.

In this thesis, we formalize the Lévy-Prokhorov metric and related notions such as weak convergence. We apply the Lévy-Prokhorov metric to formalize Prokhorov's theorem and show that the space of finite measures on a standard Borel space is also a standard Borel space. We first formalize the notion of weak convergence including the Portmanteau theorem, equivalent conditions of weak convergence, and the topology of weak convergence. We define the notion of weak convergence using *filters* as convergence in Isabelle/HOL. We then formalize the Lévy-Prokhorov metric. We prove the equivalence of the topology of weak convergence and the topology induced by the Lévy-Prokhorov metric. Our proof is different from the common textbook proofs (e.g. [10, 15]). We obtain a simpler proof thanks to the generalization of weak convergence by filters. We then formalize Prokhorov's theorem using the Lévy-Prokhorov metric. In order to formalize Prokhorov's theorem, we also prove (a special case of) Alaoglu's theorem and the Riesz representation theorem. The Riesz representation theorem is an important result in functional analysis. While its proof, including related lemmas, consists of around nine pages in Rudin's book [38], our formalization takes more than 2,100 lines of proofs. We finally show that the measurable space of finite measures on a standard Borel space is a standard Borel space. The measurable space of measures on some measurable space is used in stochastic processes and the semantics of probabilistic programs. The measurable space of measures is defined independently from metrics or topologies. We prove that the measurable space of finite measures is generated from the Lévy-Prokhorov metric. As a consequence, we obtain that the measurable space of finite measures on a standard Borel space is a standard Borel space.

Avigad et al. formalized the notion of weak convergence of probability measures on $\mathbb{R}$ and a special case of Prokhorov's theorem during the proof of the central limit theorem in Isabelle/HOL [4]. Compared to their work, our formalization of weak convergence treats finite measures on any metric spaces, and convergence is generalized by filters. While there is a simpler proof for the special case of Prokhorov's theorem that they formalized, Prokhorov's theorem that we formalize needs tools in functional analysis, such as the Riesz representation theorem, and thus requires more effort. The Lean mathematical library, mathlib [50], includes ongoing formalization of the weak convergence and the Lévy-Prokhorov metric by Kytölä [30]. Their definition of the weak convergence is also generalized by filters and treats not only probability measures but also finite measures. They showed that the Lévy-Prokhorov metric on the set of finite measures on a pseudo-metric space is a pseudo-metric. They proved the equivalence of the topology of weak convergence and the topology induced by the Lévy-Prokhorov metric on the space of probability measures. Our work contains more results than their work, such as Prokhorov's theorem (Theorem 4.20).

Quasi-Borel spaces The theory of quasi-Borel spaces provides a new denotational model for higher-order probabilistic programs. Heunen et al. [25] introduced the notion of quasi-Borel spaces and the probability monad on it. Later, the s-finite measure monad was developed [44, 52, 56]. The s-finite measure monad treats s-finite measures, while the probability monad treats only probability measures. The s-finite measure monad enables us to denote probabilistic programs with conditioning.

In this thesis, we formalize the notion of quasi-Borel spaces and the s-finite measure monad on it. This is the first formalization of the theory of quasi-Borel spaces, to our best knowledge. The probability monad is obtained by taking a subspace. We also formalize s-finite kernels, which are used to denote first-order probabilistic programs because the s-finite mea-

sure monad depends on s-finite kernels. The construction of the s-finite measure monad is non-trivial. However, no paper provides its detailed explanation. Furthermore, the details of the definition vary among previous studies [44, 52, 56]. We recover the omitted details during formalization. In addition to formalizing the s-finite measure monad, we also implement *qbs prover*, an automation for checking whether the term is a member of the quasi-Borel space or not. In a typical situation of formal proofs in measure theory, we need to check measurability of functions to apply equations or theorems because most of the theorems require that functions are measurable. Working with quasi-Borel spaces also faces similar situations. The qbs prover reduces the cost of proof by automatically solving this kind of side conditions. In the usual definition, the morphisms (structure-preserving functions) between quasi-Borel spaces are defined first, then the function spaces are constructed through morphisms. In our formalization, we first construct the function spaces, then morphisms are defined through the function spaces. This design works better with our proof automation.

Probabilistic program verification We apply quasi-Borel spaces to verify probabilistic programs. The s-finite measure monad on quasi-Borel spaces enables us to denote probabilistic programs which have three major functionalities: sampling, higher-order functions, and conditioning. Other works based on measure theories, e.g., the s-finite kernel in Coq by Affeldt et al. [2] or the Giry monad in Isabelle/HOL by Eberl et al. [18], cannot support probabilistic programs with higher-order functions.

In this thesis, we verify four example programs including the Monte Carlo method and the Gaussian mean learning algorithm. We prove that the distribution of the average of samples converges in probability to the expected value in the Monte Carlo method example. In the example of the Gaussian mean learning algorithm, we prove convergence and stability under change of priors. We also formalize differential privacy using quasi-Borel spaces and show a simple example. The definitions and properties of differential privacy using quasi-Borel spaces are easily derived from the ones using measurable spaces.

1.1 Publications

The content of this thesis is based on the following publications.

Journal Article

J1 Michikazu Hirata, Yasuhiko Minamide, Tetsuya Sato, *Program Logic for Higher-Order Probabilistic Programs in Isabelle/HOL*, Science of Computer Programming, Volume 230, 102993, Elsevier, 2023 (special issue of FLOPS2022).

Conference Papers

C1 Michikazu Hirata, *A Formalization of the Lévy-Prokhorov Metric in Isabelle/HOL*, In 15th International Conference on Interactive Theorem Proving (ITP 2024). Leibniz International Proceedings in Informatics (LIPIcs), Volume 309, pp.21:1–21:18, Schloss Dagstuhl – Leibniz-Zentrum für Informatik (2024).

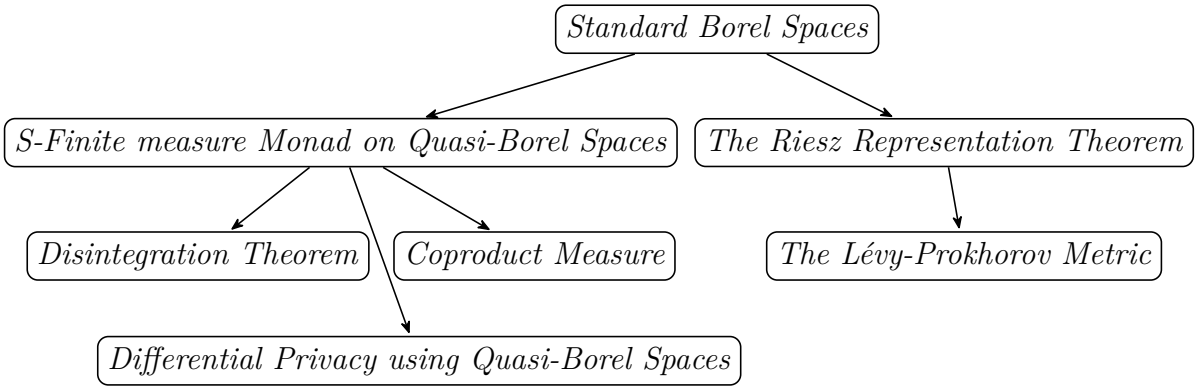


Fig. 1.1: Dependencies of AFP entries.

C2 Michikazu Hirata, Yasuhiko Minamide, Tetsuya Sato, *Semantic Foundations of Higher-Order Probabilistic Programs in Isabelle/HOL*, In 14th International Conference on Interactive Theorem Proving (ITP 2023). Leibniz International Proceedings in Informatics (LIPIcs), Volume 268, pp.18:1–18:18, Schloss Dagstuhl – Leibniz-Zentrum für Informatik (2023).

C3 Michikazu Hirata, Yasuhiko Minamide, Tetsuya Sato, *Program Logic for Higher-Order Probabilistic Programs in Isabelle/HOL*, In 16th International Symposium on Functional and Logic Programming (FLOPS 2022). Lecture Notes in Computer Science, vol 13215, pp.57–74, Springer, Cham (2022).

Chapter 3 is based on C2, Chapter 4 is based on C1, Chapter 5 is based on J1, C2, and C3, and Chapter 6 is based on C2.

Archive of Formal Proofs The proof scripts are available on the official proof libraries of Isabelle, the Archive of Formal Proofs (AFP). Theory dependencies are shown in Fig 1.1.

AFP1 Michikazu Hirata, *Differential Privacy using Quasi-Borel Spaces*, 2025.

AFP2 Michikazu Hirata, *Coproduct Measure*, 2024.

AFP3 Michikazu Hirata, *The Lévy-Prokhorov Metric*, 2024.

AFP4 Michikazu Hirata, *The Riesz Representation Theorem*, 2024.

AFP5 Michikazu Hirata, *Disintegration Theorem*, 2023.

AFP6 Michikazu Hirata and Yasuhiko Minamide, *S-Finite Measure Monad on Quasi-Borel Spaces*, 2023.

AFP7 Michikazu Hirata, *Standard Borel Spaces*, 2023.

AFP8 Michikazu Hirata, Yasuhiko Minamide, and Tetsuya Sato, *Quasi-Borel Spaces*, 2022.

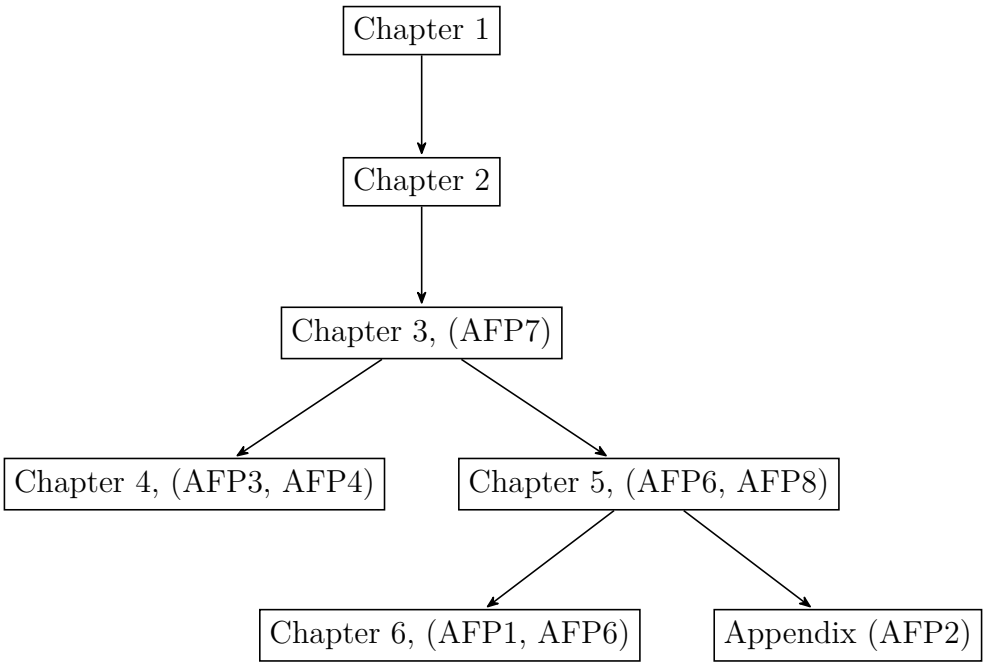


Fig. 1.2: Chapter dependencies.

1.2 Thesis Outline

In Chapter 2, we review basic mathematics used in this thesis and Isabelle/HOL. In Chapter 3, we formalize Polish spaces and standard Borel spaces. We also apply them to prove the disintegration theorem. In Chapter 4, we formalize the Lévy-Prokhorov metric and show Prokhorov's theorem. We also prove that the space of all finite measures on a standard Borel space is also a standard Borel space. In Chapter 5, we formalize quasi-Borel spaces and the s-finite measure monad. We then implement the qbs prover, a proof automation for quasi-Borel spaces. In Chapter 6, we apply quasi-Borel spaces to verify probabilistic programs. In Appendix, we formalize the coproduct measurable spaces and coproduct measures. Chapter dependencies and corresponding formalization in AFP entries are shown in Fig 1.2.

Chapter 2

Preliminaries

In this chapter, we introduce mathematical notions we use throughout this thesis and Isabelle/HOL. We first review the basis of topological, metric, and measurable spaces. Then, we give an overview of Isabelle/HOL. Finally, we explain how the mathematical notions are defined in Isabelle/HOL's standard library.

2.1 Mathematical Preliminaries

We first review mathematical notions.

2.1.1 Topological Spaces

Topology is a way of expressing *nearness* of points in a set. Let X be a set and $\mathcal{O}_X$ a set of subsets of X. The pair $(X, \mathcal{O}_X)$ is called a topological space when $\varnothing \in \mathcal{O}_X$, $X \in \mathcal{O}_X$, and $\mathcal{O}_X$ is closed under finite intersections and arbitrary unions. We sometimes write just X for $(X, \mathcal{O}_X)$ when the structure is obvious from the context. We follow the standard definitions of topology, such as,

- $U \subseteq X$ is an open set of $X \overset{\text{def}}{\iff} U \in \mathcal{O}_X$,
- $C \subseteq X$ is a closed set of $X \overset{\text{def}}{\iff} X - C$ is open, and
- $f : X \to Y$ is a continuous map $\overset{\text{def}}{\iff} \forall U \in \mathcal{O}_Y.\, f^{-1}(U) \in \mathcal{O}_X$

for topological spaces X and Y.

2.1.2 Filters and Limits

We introduce the notion of filter and limit according to the Isabelle/HOL's standard library. The notion of convergence is defined through *filters*. Filters describe "large" or "eventually" elements. Let I be a set and $\mathcal{F}$ be a set of subsets of I, then we call $\mathcal{F}$ a filter on I when the following hold.

1. $I \in \mathcal{F}$.
2. If $A \in \mathcal{F}$ and $B \in \mathcal{F}$, then $A \cap B \in \mathcal{F}$.

3. If $A \in \mathcal{F}$ and $A \subseteq B \subseteq I$, then $B \in \mathcal{F}$.

A filter $\mathcal{F}$ is called *proper* if $\varnothing \notin \mathcal{F}$ and *non-proper* otherwise. This definition allows the non-proper filter. The non-proper filter is trivial, meaning that it is the power set of I from the third axiom of filters.

We define a new quantifier $\forall_{\mathrm{F}}$ using filters[1].

$$\forall_{\mathrm{F}}\, x \text{ in } \mathcal{F}.\, P(x) \overset{\text{def}}{\iff} \{x \in I.\, P(x)\} \in \mathcal{F}.$$

The predicate $\forall_{\mathrm{F}}\, x$ in $\mathcal{F}.\, P(x)$ means that $P(x)$ eventually holds with respect to $\mathcal{F}$. The quantifier $\forall_{\mathrm{F}}$ is used to express notions such as "for sufficiently large n" and "for x sufficiently close to a". The axioms of filter correspond to the following rules.

1. $\forall_{\mathrm{F}}\, x$ in $\mathcal{F}.\, True$.
2. If $\forall_{\mathrm{F}}\, x$ in $\mathcal{F}.\, P(x)$ and $\forall_{\mathrm{F}}\, x$ in $\mathcal{F}.\, Q(x)$, then $\forall_{\mathrm{F}}\, x$ in $\mathcal{F}.\, P(x) \wedge Q(x)$.
3. If $\forall_{\mathrm{F}}\, x$ in $\mathcal{F}.\, P(x)$ and $(\forall x \in I.\, P(x) \Longrightarrow Q(x))$, then $\forall_{\mathrm{F}}\, x$ in $\mathcal{F}.\, Q(x)$.

For instance, let us define a filter on $\mathbb{N}$ as follows.

$$\mathcal{F}_{\mathrm{seq}} = \bigcup_{n \in \mathbb{N}} \{A \subseteq \mathbb{N}.\, \{n, n+1, n+2, \dots\} \subseteq A\}.$$

It is easy to check that $\mathcal{F}_{\mathrm{seq}}$ forms a filter. Then,

$$\begin{aligned} \forall_{\mathrm{F}}\, n \text{ in } \mathcal{F}_{\mathrm{seq}}.\, P(n) &\iff \{n \in \mathbb{N}.\, P(n)\} \in \mathcal{F}_{\mathrm{seq}} \\ &\iff \exists N.\, \{N, N+1, N+2, \dots\} \subseteq \{n \in \mathbb{N}.\, P(n)\} \\ &\iff \exists N.\, \forall n \geq N.\, P(n). \end{aligned}$$

For $a \in \mathbb{R}$, there exists a filter $\mathcal{F}_a$ on $\mathbb{R}$ expressing "for all x sufficiently close to a". The filter $\mathcal{F}_a$ has the following property.

$$\forall_{\mathrm{F}}\, x \text{ in } \mathcal{F}_a.\, P(x) \iff (\exists \delta > 0.\, \forall x.\, x \neq a \wedge |x - a| < \delta \Longrightarrow P(x)).$$

Next, we define the notion of convergence. Let I be a set, $\mathcal{F}$ a filter on I, X a topological space, $\{x_i\}_{i \in I}$ a sequence, and $x \in X$. The notion of convergence is defined as follows.

$$(x_i \longrightarrow x)\ \mathcal{F} \text{ in } X \overset{\text{def}}{\iff} (\forall U\colon \text{ open in } X.\, x \in U \Longrightarrow (\forall_{\mathrm{F}}\, i \text{ in } \mathcal{F}.\, x_i \in U)).$$

This definition means that for all neighborhoods of x, x_i eventually belongs to the neighborhood. Intuitively, x_i is eventually *close* to x in X. If $(x_i \longrightarrow x)\ \mathcal{F}$ in X, x is called the limit. When the topology is trivial from the context, we omit the topological space and denote the convergence by $(x_i \longrightarrow x)\ \mathcal{F}$. If (X, d) is a metric space, the convergence has the following equivalent condition.

$$(x_i \longrightarrow x)\ \mathcal{F} \text{ in } X \iff \forall \varepsilon > 0.\, \forall_{\mathrm{F}}\, i \text{ in } \mathcal{F}.\, d(x_i, x) < \varepsilon.$$

[1]Note that the subscript F of $\forall_{\mathrm{F}}$ does not bind filters.

For instance, the limit of a sequence and the limit of a function at a are expressed as the following familiar forms.

$$\begin{aligned}
\lim_{n\to\infty} x_n = x &\iff (x_n \longrightarrow x)\ \mathcal{F}_{\mathrm{seq}} \text{ in } \mathbb{R} \\
&\iff \forall \varepsilon > 0.\, \exists N.\, \forall n \geq N.\, |x_n - x| < \varepsilon. \\
\lim_{x\to a} f(x) = L &\iff (f \longrightarrow L)\ \mathcal{F}_a \text{ in } \mathbb{R} \\
&\iff (\forall \varepsilon > 0.\, \exists \delta > 0.\, \forall x.\, x \neq a \wedge |x - a| < \delta \Longrightarrow |f(x) - L| < \varepsilon)\,.
\end{aligned}$$

In addition to the limit, limit inferior and limit superior are also defined using filters.

$$\mathrm{Liminf}_{\mathcal{F}}\{x_i\}_{i\in I} \stackrel{\mathrm{def}}{=} \sup_{A\in\mathcal{F}} \inf_{i\in A} x_i, \quad \mathrm{Limsup}_{\mathcal{F}}\{x_i\}_{i\in I} \stackrel{\mathrm{def}}{=} \inf_{A\in\mathcal{F}} \sup_{i\in A} x_i.$$

It is easy to check that those limit inferior and limit superior have the same meaning as the usual definitions for $\mathcal{F}_{\mathrm{seq}}$, that is,

$$\mathrm{Liminf}_{\mathcal{F}_{\mathrm{seq}}}\{x_n\}_{n\in\mathbb{N}} = \sup_{N\in\mathbb{N}} \inf_{n\geq N} x_n, \quad \mathrm{Limsup}_{\mathcal{F}_{\mathrm{seq}}}\{x_n\}_{n\in\mathbb{N}} = \inf_{N\in\mathbb{N}} \sup_{n\geq N} x_n.$$

2.1.3 Metric Spaces

A metric space is a pair of a set X and a function $d : X \times X \to \mathbb{R}$ such that

- For all $x, y \in X$, $d(x, y) \geq 0$.
- For all $x, y \in X$, $d(x, y) = d(y, x)$.
- For all $x, y \in X$, $d(x, y) = 0 \iff x = y$.
- For all $x, y, z \in X$, $d(x, z) \leq d(x, y) + d(y, z)$.

We sometimes write just X for (X, d). Let (X, d) be a metric space, $x \in X$ and $\varepsilon > 0$. The set $ball_X(x, \varepsilon) = \{y \in X.\ d(x, y) < \varepsilon\}$ is called the open ball with center x and radius ε. For $x \in X$ and $\varepsilon \geq 0$, the set $cBall_X(x, \varepsilon) = \{y \in X.\ d(x, y) \leq \varepsilon\}$ is called the closed ball with center x and radius ε. We assume that $\mathbb{R}$ is equipped with the standard distance $d(x, y) = |x - y|$ in this thesis.

Metric space X induces the topological space $(X, \mathcal{O}_d)$ where $\mathcal{O}_d$ consists of arbitrary unions of open balls.

Definition 2.1.

- A sequence $\{x_n\}_{n\in\mathbb{N}}$ on a metric space X is called a *Cauchy sequence* if $\forall \varepsilon > 0.\, \exists N.\, \forall n, m \geq N.\, d(x_n, x_m) < \varepsilon$.
- A metric space is *complete* if every Cauchy sequence has a limit.
- A topological space X is *metrizable* if there exists a metric d on X which induces X.
- A topological space X is called a *completely metrizable space* if there exists a complete metric on X which induces X.

2.1.4 Measure theory

Measure theory is a basis of modern probability theory. Let M be a set and Σ_M a set of subsets of M. A pair (M, Σ_M) is called a *measurable space* if Σ_M is non-empty and closed under complement and countable unions. We sometimes write M for a measurable space (M, Σ_M). A member $A \in \Sigma_M$ is called a *measurable set*. A function f from a measurable space M to a measurable space N is *measurable* if $f^{-1}(A) \in \Sigma_M$ for all $A \in \Sigma_N$. Let M be a measurable space, $\mu : \Sigma_M \to [0, \infty]$ is a *measure* on M if $\mu(\varnothing) = 0$ and $\mu(\bigcup_{n\in\mathbb{N}} A_n) = \sum_{n=0}^{\infty} \mu(A_n)$ for any disjoint family $\{A_n\}_{n\in\mathbb{N}} \subseteq \Sigma_M$. A measure μ on M is called a *probability measure* if $\mu(M) = 1$, a *sub-probability measure* if $\mu(M) \leq 1$, a *finite measure* if $\mu(M) < \infty$, and a σ-finite measure if there exists a disjoint family $\{A_n\}_{n\in\mathbb{N}} \subseteq \Sigma_M$ such that $\bigcup_{n\in\mathbb{N}} A_n = M$ and $\mu(A_n) < \infty$ for all n. For a measure μ on M and a measurable function $f : M \to \mathbb{R}$, $\int f \mathrm{d}\mu$ denotes the Lebesgue integral of f with respect to μ. When we bind the argument explicitly, we write $\int f \mathrm{d}\mu$ for $\int f(x)\mu(\mathrm{d}x)$. For a measurable function $f : M \to N$ and a measure μ on N, the *push-forward measure* (or *image measure*) is a measure on M defined by $f_*\mu(A) = \mu(f^{-1}(A))$ for $A \in \Sigma_M$.

We use topological space $(X, \mathcal{O}_X)$ as the measurable space $(X, \sigma[\mathcal{O}_X])$ where $\sigma[\mathcal{O}_X]$ is the least σ-algebra including all open sets of X. The measurable space $(X, \sigma[\mathcal{O}_X])$ is called the Borel space. Notice that a metric space is also treated as the measurable space since a metric space induces a topological space. The Borel space induced by a metric space (X, d) is denoted by (X, Σ_d).

Measurable spaces and measurable functions form the category **Meas**. The Giry monad G is a monad on **Meas** [20] which expresses stochastic processes by means of measurable functions. The monad is also applied to semantics of probabilistic programs. For a measurable space M, $G(M)$ is the measurable space of all probability measures on M. For $x \in M$, the unit (return) operator assigns the Dirac measure[2] δ_x centered at x. For $\mu \in G(M)$ and a measurable function $f : M \to G(N)$, the bind $\mu \gg\!=_G f$ is the probability measure defined by $(\mu \gg\!=_G f)(A) = \int f(x)(A)\mu(\mathrm{d}x)$.

2.2 Isabelle/HOL

Formalization in this thesis are done with the interactive theorem prover Isabelle/HOL [35]. In this section, we give an overview and introduce the functionalities of Isabelle/HOL.

The syntax of Isabelle/HOL follows λ-calculus and functional programming languages. The notation $t :: \tau$ means that t has the type τ. Isabelle/HOL supports type variables as in ML and Haskell. Type variables are denoted by $'a$, $'b$, ... We denote the type of Boolean by *bool*, the type of natural numbers by *nat*, and the type of real numbers by *real*. Some of the types are built from existing types. We denote the function type from $'a$ to $'b$ by $'a \Rightarrow 'b$, the product type of $'a$ and $'b$ by $'a \times 'b$, and the type of set on $'a$ by $'a$ *set*. Both of $\Rightarrow$ and $\times$ are right-associative. The function type $\tau_1 \Rightarrow \tau_2 \Rightarrow \cdots \Rightarrow \tau_n \Rightarrow \tau$ is abbreviated to $[\tau_1, \tau_2, \ldots, \tau_n] \Rightarrow \tau$.

The **definition** command introduces a new constant.

```
definition doublex-plus-y :: [real, real] ⇒ real where
doublex-plus-y ≡ (λx y. x * 2 + y)
```

[2] $\delta_x(U) = 1$ is $x \in U$ and 0 otherwise.

The following also defines the same function without explicit λ-abstraction.

definition *doublex-plus-y* :: [*real*, *real*] ⇒ *real* **where**
doublex-plus-y x y ≡ *x* ∗ *2* + *y*

The **lemma** command introduces a new theorem.

lemma *conj*:
assumes *A* **and** *B*
shows *A* ∧ *B*
— Proof scripts follow.

Although we must give a formal proof to introduce new theorems, we usually omit proof scripts in this thesis. The **proposition**, **corollary**, and **theorem** commands have the same meaning as **lemma**.

Standard Constants

UNIV :: *'a set*	The set of all elements on type *'a*. That is, *UNIV* = {*x*::*'a*. *True*}.
A → *B*	The set of all functions from *A* to *B*. That is, *A* → *B* = {*f*. ∀ *x*∈ *A*. *f x* ∈*B*}.
f ‘ *A*	The image of *A* under *f*. That is, *f* ‘ *A* = {*y*. ∃ *a*∈ *A*. *y* = *f a*}.
f −‘ *B*	The inverse image of *B* under *f*. That is, *f* −‘ *B* = {*x*. *f x* ∈ *B*}.
undefined :: *'a*	An arbitrary element of type *'a*.
SOME x. *P x*	Some element *x* for which *P x* holds. We have the following rule: If ∃ *x*. *P x* , then *P* (*SOME x*. *P x*).
∑ *i*∈*I*. *a i*	The finite sum of *a i*.
∑ *n*. *a n*	The sum over all natural numbers. That is, ∑ *n*. *a n* = $\sum_{n=0}^{\infty}$ *a n*.

2.3 Mathematical Structures in Isabelle/HOL

There are several ways of defining mathematical structures in Isabelle/HOL. We first compare three ways by observing the definition of metric spaces. Then, we explain the quotient type definition.

2.3.1 Type classes

Isabelle provides Haskell-like type classes [22]. The type class of metric spaces is defined as follows[3].

class *metric-space* =
fixes *dist* :: *'a* ⇒ *'a* ⇒ *real*
assumes *dist-nonneg*: *dist x y* ≥ *0*
and *dist-commute*: *dist x y* = *dist y x*
and *dist-eq-0-iff*: *dist x y* = *0* ⟷ *x* = *y*
and *dist-triangle*: *dist x z* ≤ *dist x y* + *dist y z*

[3]This definition is different from the standard library's one.

Type variables are annotated with (finitely many) classes such as $'a :: metric\text{-}space$. Once we define the class *metric-space*, the constant $dist :: ('a :: metric\text{-}space) \Rightarrow 'a \Rightarrow real$ is introduced in the global context. The formalization of some structure using type class is called *type-based*.

Type-based constructions work well in proof automation. However, we face a difficulty when we want to define a metric space on subsets, e.g. $[0, 1]$, which is not the set of the whole elements of reals because type classes only work for types. In addition, there can only be one *metric-space* instance. Hence, it is not suitable when there are no *canonical* metric or we want to change the metric during the proof.

2.3.2 Locale

The **locale** command introduces a context. Isabelle/HOL's standard library defines metric spaces as follows.

```
locale Metric-space =
  fixes M :: 'a set and d :: 'a ⇒ 'a ⇒ real
  assumes nonneg: ⋀x y. 0 ≤ d x y
  assumes commute: ⋀x y. d x y = d y x
  assumes zero: ⋀x y. ⟦x ∈ M; y ∈ M⟧ ⟹ d x y = 0 ⟷ x=y
  assumes triangle: ⋀x y z. ⟦x ∈ M; y ∈ M; z ∈ M⟧ ⟹ d x z ≤ d x y + d y z
```

In this case, a set M and a function d are fixed and the four assumptions hold, that is, (M, d) forms a metric space in the context of *Metric-space*. This style of definition is called *set-based* because we specify the underlying set of the metric space unlike type-based definition.

The locale declaration introduces the predicate $Metric\text{-}space :: 'a\ set \Rightarrow ('a \Rightarrow 'a \Rightarrow real) \Rightarrow bool$ where $Metric\text{-}space\ M\ d$ means that (M, d) is a metric space in the sense of the above definition. The following are examples of how to define new constants and theorems inside the context.

```
definition(in Metric-space) mball :: 'a ⇒ real ⇒ 'a set where
 mball x r ≡ {y. x ∈ M ∧ y ∈ M ∧ d x y < r}

lemma(in Metric-space) mball-subset-concentric:
  assumes r ≤ s
  shows mball x r ⊆ mball x s
```

In contrast to the type-based definition, the set-based definition enables us to define the metric space on any carrier set. However, we often need to show the membership relation $x \in M$ which we do not need to show in the type-based development.

2.3.3 Type Definition

The **typedef** command allows users to define a new type which denotes a non-empty subset of an existing type. The type of metric space is defined as the subset of pairs forming metric spaces.

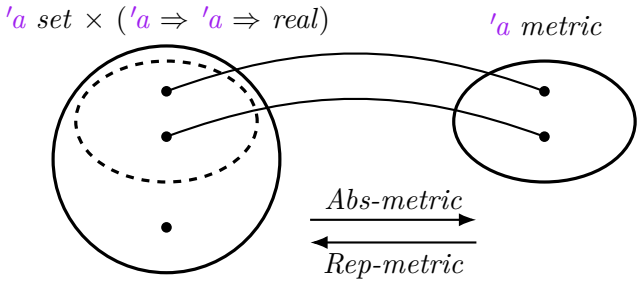
```
typedef 'a metric = {(M::'a set,d). Metric-space M d}
— Need to show that this type is non-empty.
```

The **typedef** command automatically generates the following two constants (the abstraction morphism and the representation morphism).

$$\textit{Abs-metric} :: {'a}\ \textit{set} \times ({'a} \Rightarrow {'a} \Rightarrow \textit{real}) \Rightarrow {'a}\ \textit{metric}$$
$$\textit{Rep-metric} :: {'a}\ \textit{metric} \Rightarrow {'a}\ \textit{set} \times ({'a} \Rightarrow {'a} \Rightarrow \textit{real})$$

The **typedef** command introduces the following laws:

- For all $m :: {'a}\ \textit{metric}$, $\textit{Abs-metric}\ (\textit{Rep-metric}\ m) = m$.
- If $\textit{Metric-space}\ M\ d$, then $\textit{Rep-metric}\ (\textit{Abs-metric}\ (M,d)) = (M,d)$.



Each component of metric spaces is obtained by following projection functions.

definition *mspace* **where** $\textit{mspace}\ m \equiv \textit{fst}\ (\textit{Rep-metric}\ m)$
definition *mdist* **where** $\textit{mdist}\ m \equiv \textit{snd}\ (\textit{Rep-metric}\ m)$

Same as the set-based definition, the type definition allows us to use metric spaces on any carrier set. However, it is tedious to write projection functions *mspace* and *mdist* many times. For instance, the triangle axiom is stated as follows.

lemma
assumes $x \in \textit{mspace}\ m$ **and** $y \in \textit{mspace}\ m$ **and** $z \in \textit{mspace}\ m$
shows $\textit{mdist}\ m\ x\ z \leq \textit{mdist}\ m\ x\ y + \textit{mdist}\ m\ y\ z$

2.3.4 Quotient Type

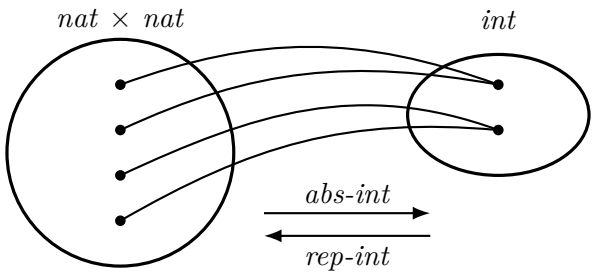
The **quotient-type** command [28] defines a quotient type when an equivalence relation over a raw type is given. For instance, the type of integers is defined as follows.

quotient-type $\textit{int} = \textit{nat}\times\textit{nat}\ /\ \textit{intrel}$
— Need to prove that *intrel* is an equivalent relation.

The relation *intrel* is an equivalent relation over the type $\textit{nat}\times\textit{nat}$ defined by $\textit{intrel}\ (x,y)\ (u,v)$ if and only if $x + v = y + u$. The representative of a pair of natural numbers (x, y) denotes the integer $x - y$. The **quotient-type** command generates the following constants and lemmas related to those constants.

$$\textit{abs-int} :: \textit{nat}\times\textit{nat} \Rightarrow \textit{int}, \quad \textit{rep-int} :: \textit{int} \Rightarrow \textit{nat}\times\textit{nat}.$$

Intuitively, *rep-int* is a function that takes an equivalence class (an integer number in this example) and computes one of its representatives and *abs-int* is the natural surjection, which maps an element to its equivalence class.



The **lift-definition** defines constants related to types defined by the **quotient-type**.

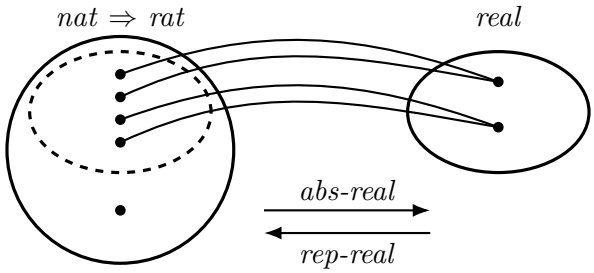
lift-definition *zero-int* :: *int* **is** (*0*, *0*) .
lift-definition *plus-int* :: *int* ⇒ *int* ⇒ *int*
is λ(*x*, *y*) (*u*, *v*). (*x* + *u*, *y* + *v*)
— Need to prove that *plus-int* is *well-defined*.

Note that + in the definition of *plus-int* is the addition on *nat*.

In some situations, users want to define a quotient type over a subset of some type. For instance, the type of real is obtained as a quotient type on the set of all Cauchy sequences on rational numbers. In this case, the base set (the set of all Cauchy sequences) is a subset of *nat* ⇒ *rat*. The **quotient-type** also provides a way to define such a kind of type. The type of real numbers is defined in Isabelle/HOL as follows.

quotient-type *real* = *nat* ⇒ *rat* / *partial*: *realrel*
— Need to prove that *realrel* is a partial equivalent relation.

The relation *realrel* is a partial equivalence relation, that is, there exists a *r* such that *realrel* *r* *r* and *realrel* is symmetric and transitive. The **quotient-type** command generates the abstraction/representation functions and related lemmas as in the previous example.



2.4 Topology, Metric Spaces, and Measure Theory in Isabelle/HOL

Isabelle/HOL's standard library includes formalization of topology, metric spaces, and measure theory. The following table shows how each structure is defined in Isabelle/HOL.

	Type classes	Locale	Type definition
Topology	✓		✓
Metric spaces	✓	✓	✓
Measure theory			✓

The type-based topological spaces and metric spaces were formalized by Hölzl et al. [27]. The topological space library defined by type definition is ported from HOL-Light. Paulson implemented metric spaces based on locale and type definition [36] by porting the HOL-Light's metric space library [34]. The current measure theory library was first formalized by Hölzl and Heller [26] and has been extended by several other works [4, 18].

Topological spaces

There are two kinds of topological spaces: the type class *'a*::*topological-space* and the type of topological space *'a topology*. In this thesis, we mainly work with not type-based topological spaces but the type of topological spaces because we use topological spaces whose carrier sets are not the type universe in general. We list basic operators for topological spaces.

topspace	::	*'a topology* ⇒ *'a set*
topspace X	=	The carrier set of X.
openin	::	*'a topology* ⇒ *'a set* ⇒ *bool*
openin X U	⟺	U is an open set of X.
closedin	::	*'a topology* ⇒ *'a set* ⇒ *bool*
closedin X C	⟺	C is a closed set of X.
compactin	::	*'a topology* ⇒ *'a set* ⇒ *bool*
compactin X K	⟺	K is a compact set of X.
topology-generated-by	::	*'a set set* ⇒ *'a topology*
topology-generated-by $\mathcal{O}$	=	The topological space generated from $\mathcal{O}$.
continuous-map	::	*'a topology* ⇒ *'b topology* ⇒ (*'a* ⇒ *'b*) ⇒ *bool*
continuous-map X Y f	⟺	f is a continuous map from X to Y.
euclidean	::	*'a*::*topological-space topology*
euclidean	=	The abstract topology obtained from its type.
closure-of	::	*'a topology* ⇒ *'a set* ⇒ *'a set*
X closure-of A	=	The closure of A in X.

Filter and Limits

The type *'a filter* denotes the type of filters.

sequentially	::	*nat filter*
sequentially	=	$\mathcal{F}_{\text{seq}}$
eventually	::	(*'a* ⇒ *bool*) ⇒ *'a filter* ⇒ *bool*
eventually P $\mathcal{F}$	⟺	$\forall_{\text{F}}\ i \text{ in } \mathcal{F}.\ P\ i$
$\forall_F$ *i in* $\mathcal{F}$. *P i*	=	*eventually P* $\mathcal{F}$
$(\square \longrightarrow \square)\ \square$	::	(*'a* ⇒ *'b*::*topological-space*) ⇒ *'b* ⇒ *'a filter* ⇒ *bool*
$(xn \longrightarrow x)\ \mathcal{F}$	⟺	$(xn \longrightarrow x)\ \mathcal{F}$
limitin	::	*'a topology* ⇒ (*'b* ⇒ *'a*) ⇒ *'a* ⇒ *'b filter* ⇒ *bool*
limitin X xn x $\mathcal{F}$	⟺	$(xn \longrightarrow x)\ \mathcal{F}$ in X

Note that there are two kinds of limits $(\square \longrightarrow \square)\ \square$ and *limitin*. While the former one determines the topology from type classes, we specify the topology for the latter limit.

Metric Spaces

We use the set-based metric spaces library because we will use metrics whose carrier sets are not the set of all elements of the type and change metrics during proofs. The context of metric space is defined as follows.

locale *Metric-space* =
  fixes $M :: 'a\ set$ **and** $d :: 'a \Rightarrow 'a \Rightarrow real$
  assumes *nonneg*: $\bigwedge x\ y.\ 0 \leq d\ x\ y$
  assumes *commute*: $\bigwedge x\ y.\ d\ x\ y = d\ y\ x$
  assumes *zero*: $\bigwedge x\ y.\ [\![x \in M;\ y \in M]\!] \Longrightarrow d\ x\ y = 0 \longleftrightarrow x{=}y$
  assumes *triangle*: $\bigwedge x\ y\ z.\ [\![x \in M;\ y \in M;\ z \in M]\!] \Longrightarrow d\ x\ z \leq d\ x\ y + d\ y\ z$

Notice that the non-negativity and commutativity must hold on not only M but the whole type. These assumptions make it easier to use non-negativity and commutativity in proofs, and do not change the essential structure of the metric space. Due to these assumptions, we need to take care of non-negativity and commutativity even outside of the carrier set when we define a metric space. We list standard constants in the context of *Metric-space*.

mtopology	::	$'a\ topology$
mtopology	=	The topological space induced by (M, d).
mball	::	$'a \Rightarrow real \Rightarrow 'a\ set$
mball x e	=	The open ball with center x and radius e.
mcball	::	$'a \Rightarrow real \Rightarrow 'a\ set$
mcball x e	=	The closed ball with center x and radius e.

Measure Theory

The type $'a\ measure$ denotes the type of measures. A measure consists of the measure and the measurable space on which the measure is defined. We also use a measure as a measurable space.

space	::	$'a\ measure \Rightarrow 'a\ set$
space M	=	The carrier set of M.
sets	::	$'a\ measure \Rightarrow 'a\ set\ set$
sets M	=	The σ-algebra of M.
emeasure	::	$'a\ measure \Rightarrow 'a\ set \Rightarrow ennreal$
emeasure M A	=	$M(A)$
measure	::	$'a\ measure \Rightarrow 'a\ set \Rightarrow real$
measure M A	=	$M(A)$ as a real number[4].
borel	::	$'a$::*topological-space measure*
borel	=	The borel space.
lborel	::	$'a$::*euclidean-space measure*
lborel	=	The Lebesgue measure[5].
count-space	::	$'a\ set \Rightarrow 'a\ measure$

[4] If *emeasure M A* $= \infty$, then *measure M A* $= 0$.
[5] Strictly speaking, *completion lborel* is the Lebesgue measure.

count-space I	=	The counting measure on I.
distr	::	$'a$ *measure* $\Rightarrow$ $'b$ *measure* $\Rightarrow$ ($'a \Rightarrow 'b$) $\Rightarrow$ $'b$ *measure*
distr M N f	=	The push-forward measure f_*M on N.
$\square \rightarrow_M \square$	::	$'a$ *measure* $\Rightarrow$ $'b$ *measure* $\Rightarrow$ ($'a \Rightarrow 'b$) *set*
$M \rightarrow_M N$	=	The set of all measurable functions from M to N.
$\square \bigotimes_M \square$	::	$'a$ *measure* $\Rightarrow$ $'b$ *measure* $\Rightarrow$ ($'a \times 'b$) *measure*
$M \bigotimes_M N$	=	The binary product measure of M and N.
PiM	::	$'i$ *set* $\Rightarrow$ ($'i \Rightarrow$ $'a$ *measure*) $\Rightarrow$ ($'i \Rightarrow 'a$) *measure*
$\Pi_M\ i{\in}I.\ M\ i$	=	*PiM I M*
$\Pi_M\ i{\in}I.\ M\ i$	=	The product measure.
$integral^N$	::	$'a \Rightarrow ('a \Rightarrow ennreal) \Rightarrow ennreal$
$\int^+ x.\ f\ x\ \partial M$	=	$integral^N\ M\ f$
$\int^+ x.\ f\ x\ \partial M$	=	The non-negative Lebesgue integral of f with respect to M.
integrable	::	$'a \Rightarrow ('a \Rightarrow 'b{::}\{second\text{-}countable\text{-}topology, real\text{-}normed\text{-}vector\}) \Rightarrow bool$
integrable M f	$\Longleftrightarrow$	f is integrable with respect to M.
$integral^L$	::	$'a \Rightarrow ('a \Rightarrow 'b{::}\{second\text{-}countable\text{-}topology, real\text{-}normed\text{-}vector\}) \Rightarrow 'b$
$\int x.\ f\ x\ \partial M$	=	$integral^L\ M\ f$
$\int x.\ f\ x\ \partial M$	=	The Bochner integral of f with respect to M.
prob-space	::	$'a$ *measure* $\Rightarrow$ *bool*
prob-space M	=	M is a probability measure.
subprob-space	::	$'a$ *measure* $\Rightarrow$ *bool*
subprob-space M	=	M is a sub-probability measure.
prob-algebra	::	$'a$ *measure* $\Rightarrow$ $'a$ *measure measure*
prob-algebra M	=	The space of all probability measures on M.
subprob-algebra	::	$'a$ *measure* $\Rightarrow$ $'a$ *measure measure*
subprob-algebra M	=	The space of all sub-probability measures on M.
return	::	$'a$ *measure* $\Rightarrow$ $'a \Rightarrow$ $'a$ *measure*
return M x	=	The Dirac measure δ_x on M.
$\square \gg\!= \square$	::	$'a$ *measure* $\Rightarrow$ ($'a \Rightarrow$ $'b$ *measure*) $\Rightarrow$ $'b$ *measure*
$M \gg\!= f$	=	$M \gg\!=_G f$.

We sometimes write $M\ A$ for *emeasure M A* using coercion. The Bochner integral is equal to the Lebesgue integral when $'b$ is *real* or *complex*.

2.5 Remark on Source Code

Throughout this thesis, we sometimes use usual mathematical symbols in Isabelle source code for readability. For instance, we might write *openin* $\mathbb{R}$ U and $f \in \mathbb{R} \rightarrow_M \mathbb{R}$ instead of *openin euclidean U* and $f \in$ *borel* $\rightarrow_M$ *borel*.

Chapter 3

Standard Borel Spaces

Standard Borel spaces and Polish spaces play an important role in applied probability theory because they have useful properties (e.g., Kuratowski's theorem), which do not hold for arbitrary measurable spaces and many practical spaces, e.g., $\mathbb{N}$, $\mathbb{R}$, and their countable products, are both of Polish and standard Borel spaces. Especially, the theory of standard Borel spaces is a theoretical basis of the theory of quasi-Borel spaces which we will discuss in Chapter 5.

We formalize Polish and standard Borel spaces and show Kuratowski's theorem, one of the important consequences related to standard Borel spaces. In Section 3.4, we apply Kuratowski's theorem to prove the disintegration theorem, which guarantees the existence of a *conditional probability kernel*.

Contributions

The main contributions are formalization of Polish spaces and standard Borel spaces. The existing formalization of Polish spaces in Isabelle/HOL, which is built as a type class, has restrictions such as it works only with types and still lacks theorems required in the later sections. Our formalization works with any carrier sets and includes advanced theorems such as the fact that any Polish space is homeomorphic to a G_δ set of the Hilbert cube, and Kuratowski's theorem. In order to prove Kuratowski's theorem, we use the product metric spaces. The definitions of product metric vary depending on contexts, e.g., the metric could be $\sum_{n\in\mathbb{N}}(1/2)^n d_n(x_n, y_n)$ or $\sum_{n\in\mathbb{N}}(1/3)^n d'_{g(n)}(x_{g(n)}, y_{g(n)})$, where d' is indexed by a set I and $g : \mathbb{N} \to I$ is a bijection. We define the product metric with the coefficient, I, and g parameterized, so that we can accommodate these definitions.

Reference

We refer to the textbook by Srivastava [46] and the lecture note by Biskup [11] for Polish spaces, Standard Borel spaces, and Kuratowski's theorem. The proof of disintegration theorem is based on Chapter 14.D of the book by Baccelli et al. [5].

3.1 Polish Spaces

Standard Borel space is defined through Polish spaces. We explain standard definitions and lemmas of Polish spaces. We also discuss our implementation of the product metric used in

the proof of Kuratowski's theorem.

Definition 3.1 (Polish Space)**.** A topological space X is called a *Polish space* if X is separable and completely metrizable.

For instance, $\mathbb{R}$ and countable discrete topologies are Polish spaces. In general, we need to provide a specific metric to prove that a topological space is a Polish space. The following lemma is convenient when we want to show that some topological space is a Polish space without giving a metric.

Lemma 3.2. If X and Y are homeomorphic and X is a Polish space, then Y is a Polish space.

Although there are no standard metric on extended real numbers $\overline{\mathbb{R}}$, $\overline{\mathbb{R}}$ is a Polish space because $\overline{\mathbb{R}}$ is homeomorphic to $[0,1]$.

The following lemma tells us when the subspace topology of a Polish space is a Polish space.

Lemma 3.3. Let X be a Polish space and $A \subseteq X$, then the subspace topology A is a Polish space if and only if A is a G_δ subset[1] of X.

Remember that if (X, d) is a complete metric space and $C \subseteq X$, then (C, d) is complete if and only if C is closed. Hence, the metric on a G_δ subset of a Polish space Y which is given as an evidence of the Polishness might be different from the one on Y. For instance, the subspace topology $(0, 1)$ of $\mathbb{R}$ is a Polish space. However, $(0, 1)$ is not complete with the standard metric because the Cauchy sequence $a_n = \frac{1}{n+1}$ does not have a limit in $(0, 1)$.

Lemma 3.4. Let I be a countable set and $\{X_i\}_{i \in I}$ are Polish spaces. Then, $\prod_{i \in I} X_i$ is Polish space.

Let $\mathscr{H} = [0,1]^{\mathbb{N}}$ be the *Hilbert cube* and $\mathscr{C} = \{0,1\}^{\mathbb{N}}$ the *Cantor space*. The following two lemmas are used in the proof of Kuratowski's theorem.

Lemma 3.5. If X is a separable metrizable space, then X is homeomorphic to a subset of $\mathscr{H}$. Furthermore, if X is a Polish space, then X is homeomorphic to a G_δ subset of $\mathscr{H}$.

Lemma 3.6. If a Polish space X is uncountable, then $\mathscr{C}$ is homeomorphic to a G_δ subset of X.

Polish Spaces in Isabelle/HOL

The definition of Polish space is a direct translation from the mathematical statement by using existing constants.

definition *Polish-space* $X \equiv$ *completely-metrizable-space* $X \wedge$ *separable-space* X

Isabelle/HOL's library includes the type class of Polish spaces. While the type-based Polish spaces require us to define a specific metric, our definition only requires the metrizability. Our definition is useful when there is no standard metric on a topological space, such as extended reals and a set of all finite measures.

[1] A subset A of a topological space X is a G_δ subset of X if it is a countable intersections of open sets.

Formalization of Product Metric

We define the product metric on product spaces with a countable index set. In order to enumerate elements of a countable set, Isabelle/HOL defines a function embedding a countable set into $\mathbb{N}$ and its inverse function.

$$\textit{to-nat-on} :: {'a}\ \textit{set} \Rightarrow {'a} \Rightarrow \textit{nat}, \quad \textit{from-nat-into} :: {'a}\ \textit{set} \Rightarrow \textit{nat} \Rightarrow {'a}.$$

These two constants have the following properties.

- If I is countable and $i \in I$, then $\textit{from-nat-into}\ I\ (\textit{to-nat-on}\ I\ i) = i$.
- If I is finite and $n < \textit{card}\ I$, then $\textit{to-nat-on}\ I\ (\textit{from-nat-into}\ I\ n) = n$.
- If I is countably infinite, then $\textit{to-nat-on}\ I\ (\textit{from-nat-into}\ I\ n) = n$ for all n.

The type-based metric space defines the metric on product spaces as follows.

$$\begin{aligned} \textit{dist} &:: ({'i} :: \textit{countable} \Rightarrow {'a} :: \textit{metric-space}) \Rightarrow ({'i} \Rightarrow {'a}) \Rightarrow \textit{real} \\ \textit{dist}\ x\ y &= (\sum n.\ (1\ /\ 2)^n * \textit{min}\ (\textit{dist}\ (x\ (\textit{from-nat}\ n))\ (y\ (\textit{from-nat}\ n)))\ 1) \end{aligned}$$

where $\textit{from-nat} = \textit{from-nat-into}\ (\textit{UNIV} :: {'i}\ \textit{set})$. This definition uses the smaller value of the distance and 1 because every coefficient of $(1/2)^n$ needs to be uniformly bounded to make the sequence summable. If we define the set-based metric on product spaces the same as the type-based one, we face some problems.

- In a typical situation, we use the product metric with $I = \mathbb{N}$. However, the product metric is not so intuitive, meaning that the metric does not take the sum of $(1/2)^n d_n(x_n, y_n)$ but the sum of terms including $\textit{from-nat-into}\ I\ n$.
- In the proof of Kuratowski's theorem in a later section, we want to use $(1/3)^n$ instead of $(1/2)^n$ for the coefficient of distance functions.

From these points, we define the set-based product metric using the **locale** command.

```
locale Product-metric =
  fixes r :: real
    and I :: 'i set
    and f :: 'i ⇒ nat
    and g :: nat ⇒ 'i
    and Mi :: 'i ⇒ 'a set
    and di :: 'i ⇒ 'a ⇒ 'a ⇒ real
    and K :: real
  assumes 0 < r and r < 1
      and countable I
      and ⋀i. i ∈ I ⟹ g (f i) = i
      and finite I ⟹ bij-betw f I {..< card I} and finite I ⟹ bij-betw g {..< card I} I
      and infinite I ⟹ bij-betw f I UNIV and infinite I ⟹ bij-betw g UNIV I
      and ⋀n. infinite I ⟹ f (g n) = n
      and ⋀i. i ∈ I ⟹ Metric-space (Mi i) (di i)
      and ⋀i x y. 0 ≤ di i x y
      and ⋀i x y. di i x y ≤ K
      and 0 < K
```

definition(**in** *Product-metric*) *product-dist* :: ('*i* ⇒ '*a*) ⇒ ('*i* ⇒ '*a*) ⇒ *real* **where**
product-dist ≡ (λ*x y*. *if x* ∈ (Π$_E$ *i*∈*I*. *Mi i*) ∧ *y* ∈ (Π$_E$ *i*∈*I*. *Mi i*)
then (∑ *n*. *if g n* ∈ *I then r*^*n* ∗ *di* (*g n*) (*x* (*g n*)) (*y* (*g n*)) *else 0*)
else 0)

sublocale *Product-metric* ⊆ *Metric-space* (Π$_E$ *i*∈*I*. *Mi i*) *product-dist*

In the context *Product-metric*, g is some function embedding I into $\mathbb{N}$, f is its inverse function, r specifies the coefficient of the product metric, and K is an upper bound of distances. In typical cases, we use the context *Product-metric* with $I = \mathbb{N}$ and $g = f = \mathrm{id}$, or with $g =$ *to-nat-on I* and $f =$ *from-nat-into I*. For instance, the following interpretation gives the metric on $\mathscr{C}$ defined by $d(x, y) = \sum_{n\in\mathbb{N}}(1/3)^n|x_n - y_n|$.

interpretation *Cspace*: *Product-metric* (*1/3*) (*UNIV* :: *nat set*) *id id* (λ*n*. {*0*,*1*::*real*}) (λ*n x y*. *if* (*x* = *0* ∧ *y* = *1*) ∨ (*x* = *1* ∧ *y* = *0*) *then 1 else 0*) *1*

3.2 Standard Borel Spaces

Let us see the definition of the standard Borel spaces and their properties.

Definition 3.7. A *standard Borel space* is the Borel space of a Polish space.

For instance, $\mathbb{R}$, $\overline{\mathbb{R}}$ and $\mathbb{N}$ are standard Borel spaces. The standardness is preserved by isomorphism, subspace of a measurable set, countable product, and countable coproduct.

Lemma 3.8. If M and N are measurable isomorphic and M is a standard Borel space, then N is also a standard Borel space.

Lemma 3.9. If M is a standard Borel space and $A \in \Sigma_M$, then the subspace A is a standard Borel space.

Proof Outline. Let $\mathcal{O}_M$ be a topology on M such that $(M, \mathcal{O}_M)$ is a Polish space generating (M, Σ_M). If A is a Polish space with respect to the subspace topology of $(M, \mathcal{O}_M)$, A is a standard Borel space. However, A is Polish with respect to the subspace topology if and only if A is a G_δ subset of $(M, \mathcal{O}_M)$ from Lemma 3.3. Hence, we cannot prove the lemma directly when A is not a G_δ subset of $(M, \mathcal{O}_M)$. Thus, we need construct a topology $\mathcal{O}'_M$ on M such that A is a G_δ subset of $(M, \mathcal{O}'_M)$ and $(M, \mathcal{O}'_M)$ generates M. The existence of such a topology follows from the following claim.

Claim 3.10. Let $(X, \mathcal{O}_X)$ be a Polish space and A a Borel measurable set of X. Then, there exists a finer Polish topology $\mathcal{O}'_X$ on X such that A is closed and open in $\mathcal{O}'_X$ and $\sigma[\mathcal{O}_X] = \sigma[\mathcal{O}'_X]$.

□

Lemma 3.11. Let I be a countable set. If M_i is a standard Borel space for all $i \in I$, then $\prod_{i\in I} M_i$ and $\coprod_{i\in I} M_i$ are standard Borel spaces.

The construction and formalization of coproduct spaces is found in Appendix A.1. We explain our formalization of standard Borel spaces in the next section.

3.3 Kuratowski's Theorem

Kuratowski's theorem tells us the essential structure of standard Borel spaces.

Theorem 3.12 (Kuratowski's Theorem)**.** A standard Borel space is either a countable discrete space or isomorphic to $\mathbb{R}$.

Corollary 3.13 (The Borel Isomorphism Theorem)**.** Two standard Borel spaces are isomorphic if and only if they have the same cardinality.

Corollary 3.14. If M is a non-empty standard Borel space. Then, there exist measurable functions $f : M \to \mathbb{R}$ and $g : \mathbb{R} \to M$ such that $g \circ f = \mathrm{id}_M$.

Let us first observe the Schröder–Bernstein theorem for measurable functions.

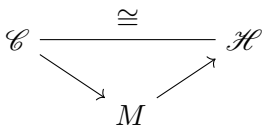
Lemma 3.15 (Schröder–Bernstein Theorem for measurable functions)**.** Let M and N be measurable spaces, $A \in \Sigma_M$, $B \in \Sigma_N$, $f : M \to B$ and $g : N \to A$ measurable isomorphisms. Then, there exist measurable isomorphisms $f' : M \to N$ and $g' : N \to M$ which extend f and g, respectively.

In general, the theorem mentions cardinality of sets without measurability assumptions as $|X| \leq |Y|$ and $|X| \geq |Y|$ implies $|X| = |Y|$. The proof is almost the same as the standard proof of the Schröder–Bernstein theorem because the isomorphisms are constructed by *countable operations*.

Proof Outline. The proof of Kuratowski's theorem is divided into three parts. If a standard Borel space M is countable, then M is obviously the discrete space because every singleton set, the closed ball with radius 0, is a measurable set. Let M be an uncountable standard Borel space.

- M is isomorphic to a Borel subset of $\mathscr{H}$ by Lemma 3.5.
- $\mathscr{C}$ is isomorphic to a Borel subset of M by Lemma 3.6.
- $\mathscr{C}$ and $\mathscr{H}$ are measurable isomorphic (Proposition 15.9 in [11][2]).

From these three facts, we conclude that any uncountable standard Borel spaces are isomorphic, especially isomorphic to $\mathbb{R}$.



□

[2]First, we prove that $[0,1]$ and $\mathscr{C}$ are isomorphic. Let $f : [0,1] \to \mathscr{C}$ be $f(x)_n = \lfloor x2^{n+1} \rfloor \mod 2$. Then, f is an isomorphic map from $[0,1]$ to a measurable set of $\mathscr{C}$. Let $g : \mathscr{C} \to [0,1]$ be $g(x) = \sum_n (1/3)^{n+1} x_n$. Then, g is an isomorphic map from $\mathscr{C}$ to a measurable set of $[0,1]$. Thus, $[0,1] \cong \mathscr{C}$ from the Schröder–Bernstein theorem. Hence, $\mathscr{C} \cong \left(\{0,1\}^{\mathbb{N}}\right)^{\mathbb{N}} \cong \mathscr{H}$. In the proof, we metrize $\mathscr{C}$ by $d(x,y) = \sum_n (1/3)^n |x_n - y_n|$.

Standard Borel Spaces in Isabelle/HOL

We define the notion of standard Borel spaces with the **locale** command.

```
locale standard-borel =
  fixes M :: 'a measure
  assumes Polish-space: ∃S. Polish-space S ∧ sets M = sets (borel-of S)
```

This definition allows a standard Borel space to be the empty space. The empty space is not suitable to define an embedding function as in Corollary 3.14. Hence, we define the context of a non-empty standard Borel space.

```
locale standard-borel-ne = standard-borel +
  assumes space-ne: space M ≠ {}
```

For a standard Borel space M, we define two measurable functions $f : M \to \mathbb{R}$ and $g : \mathbb{R} \to M$ such that $g \circ f = \mathrm{id}_M$.

```
definition to-real-on :: 'a measure ⇒ 'a ⇒ real where
to-real-on M ≡ (if uncountable (space M)
                  then (SOME f. measurable-isomorphic-map M ℝ f)
                  else (real ∘ to-nat-on (space M)))

definition from-real-into :: 'a measure ⇒ real ⇒ 'a where
from-real-into M ≡ (if uncountable (space M)
                      then the-inv-into (space M) (to-real-on M)
                      else (λr. from-nat-into (space M) (nat ⌊r⌋)))
```

If the given standard Borel space M is uncountable, then *to-real-on M* returns an isomorphic function from M to $\mathbb{R}$ and *from-real-into M* returns its inverse function. If the given standard Borel space M is countable, then *to-real-on M* and *from-real-into M* are defined using *to-nat-on* and *from-nat-into* as follows.

$$M \xrightarrow{\textit{to-nat-on (space M)}} \mathbb{N} \xrightarrow{\textit{real}} \mathbb{R} \xrightarrow{\lambda r.\ \textit{nat}\ \lfloor r \rfloor} \mathbb{N} \xrightarrow{\textit{from-nat-into (space M)}} M$$

In the context of *standard-borel*, we use abbreviations.

```
abbreviation(in standard-borel) to-real   ≡ to-real-on M
abbreviation(in standard-borel) from-real ≡ from-real-into M
```

Constants *to-real* and *from-real* have the following properties.

```
lemma(in standard-borel) to-real-measurable: to-real ∈ M →M ℝ
lemma(in standard-borel-ne) from-real-measurable: from-real ∈ ℝ →M M

lemma(in standard-borel) from-real-to-real:
  assumes x ∈ space M
  shows from-real (to-real x) = x

lemma(in standard-borel) to-real-from-real:
  assumes uncountable (space M)
  shows to-real (from-real r) = r
```

3.4 Disintegration Theorem

In this section, we apply the Kuratowski's theorem to prove the disintegration theorem. Let us see an example with the uniform distribution on the unit square $[0,1] \times [0,1]$. Such a distribution is exactly the Lebesgue measure ν on $[0,1] \times [0,1]$. Each probability of an event $A \subseteq [0,1] \times [0,1]$ is the area $\nu(A)$. We consider a *conditional probability* when $x = a$ is fixed (Fig. 3.1). What is the conditional probability κ_a for each a? A naive answer is $\kappa_a(B) = \nu(\{a\} \times B)$. However, it is obvious that $\kappa_a(B) = 0$ for all B because $\{a\} \times B$ is on the line $\{a\} \times [0,1]$ and its area is 0. Thus, this κ_a is not what we want as a conditional probability measure. From a perspective of requirements, we want the conditional probability measure κ_a to have the following properties.

- κ_a is a probability measure on $[0,1]$ for all $a \in [0,1]$.
- $\nu(A \times B) = \int_A \kappa_a(B)\nu'(\mathrm{d}a)$ for all measurable sets A and B, where ν' is the marginal measure of ν defined by[3] $\nu'(A) = \nu(A \times [0,1])$. Intuitively, $\nu(A \times B)$ is equal to the integral of "each probability of B at a" over A (Fig. 3.2).

A *correct* conditional probability measure satisfying those condition is $\kappa_a = \mu$ where μ is the Lebesgue measure on $[0,1]$, that is, the conditional probability is length.

From this example, we observed that there is a conditional probability measure for the uniform distribution on $[0,1] \times [0,1]$, but it is not the naive one. Natural questions are (1) does the conditional probability measure exist for other cases? (2) Is the conditional probability measure unique? The disintegration theorem tells us that under certain conditions, a conditional probability measure exists and it is unique in the sense of almost everywhere.

A conditional probability is represented by a probability kernel. We only provide a necessary definition here. In Section 5.1, we explain other kinds of kernels and details.

Definition 3.16. Let M and N be measurable spaces. A *probability kernel* from M to N is a function $\kappa : M \times \Sigma_N \to \overline{\mathbb{R}}_{\geq 0}$ such that:

- for each $B \in \Sigma_N$, $(\lambda x.\, \kappa_x(B))$ is measurable, and
- for each $x \in M$, $(\lambda B.\, \kappa_x(B))$ is a probability measure on N.

[3] In this case, ν' is equal to the Lebesgue measure on $[0,1]$.

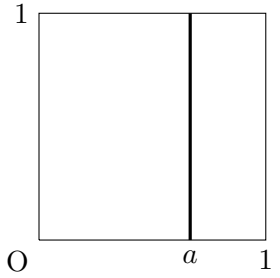


Fig. 3.1: *Conditioning* at $x = a$

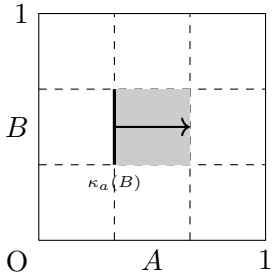


Fig. 3.2: An image of $\nu(A \times B) = \int_A \kappa_a(B)\nu'(\mathrm{d}a)$

Theorem 3.17 (Disintegration Theorem)**.** Let M be a measurable space, N a standard Borel space, ν a σ-finite measure on $M \times N$, and ν_M be the marginal measure on M defined by $\nu_M(A) = \nu(A \times N)$. Assume that ν_M is σ-finite, then there exists a probability kernel κ from M to N such that

$$\nu(A \times B) = \int_A \kappa_x(B)\nu_M(\mathrm{d}x) \quad \text{for all } A \in \Sigma_M \text{ and } B \in \Sigma_N. \tag{3.1}$$

Such a probability kernel is unique ν_N-almost everywhere.

Proof Outline. The theorem is derived from the special case when $N = \mathbb{R}$. We skip the case $N = \mathbb{R}$ in this thesis. Let us assume that the disintegration theorem holds for $N = \mathbb{R}$. From Kuratowski's theorem, we have measurable functions $f : N \to \mathbb{R}$ and $g : \mathbb{R} \to N$ such that $g \circ f = \mathrm{id}_N$. Let $\nu' = (id \times f)_*\nu$ be the push-forward measure, then ν' is a σ-finite measure on $M \times \mathbb{R}$ and ν'_M $(= \nu_M)$ is also σ-finite. Hence, there exists a probability kernel ι from M to $\mathbb{R}$ such that $\nu'(A \times B) = \int_A \iota_x(B)\nu_M(\mathrm{d}x)$ for all $A \in \Sigma_M$ and $B \in \Sigma_{\mathbb{R}}$ and ι is unique ν_M almost everywhere. Let $\kappa_x = g_*\iota_x$. Then, κ_x is a probability kernel from M to N and we have the following for $A \in \Sigma_M$ and $B \in \Sigma_N$.

$$\nu(A \times B) = \nu'(A \times f^{-1}(B)) = \int_A \iota_x(f^{-1}(B))\nu_M(\mathrm{d}x) = \int_A \kappa_x(B)\nu_M(\mathrm{d}x).$$

Next, we prove the uniqueness of κ. Let κ' be a probability kernel from M to N satisfying the equation (3.1). We define a probability kernel $\iota'_x = f_*\kappa'_X$ from M to $\mathbb{R}$. Then, $\nu'(A \times B) = \int_A \iota'_x(B)\nu_M(\mathrm{d}x)$ for all $A \in \Sigma_M$ and $B \in \Sigma_{\mathbb{R}}$. Hence, $\iota_x = \iota'_x$ holds for ν_M almost everywhere x. Thus, we have $\kappa_x = g_*\iota_x = g_*\iota'_x = \kappa'_x$ for ν_M almost everywhere x. □

We do not show Isabelle's source code in this section because it is almost a direct translation of mathematical notations.

Chapter 4

Measurable Space of Finite Measures

Measurable space on a set of measures is used in stochastic processes and semantics of probabilistic programs. In this chapter, we show the measurable space of all finite measures on a standard Borel space is also a standard Borel space using the *Lévy-Prokhorov metric*, a metric between finite measures on a metric space.

In Section 4.1, we formalize characterizations of closed sets, open sets, and topologies by limits. In Section 4.2, we define the weak convergence of measures, the topology of weak convergence, and the Lévy-Prokhorov metric. In Section 4.3, we explain Prokhorov's theorems and lemmas used in the proof of Prokhorov's theorem. In Section 4.4, we discuss the measurable space of finite measures.

Contributions

The main contributions are formalization of weak convergence, Prokhorov's theorem, and the fact that the measurable space of all finite measures on a standard Borel space is also a standard Borel space, shown by using the Lévy-Prokhorov metric. Avigad et al. [4] formalized weak convergence and a special case of Prokhorov's theorem. Their formalization of weak convergence treats only sequences of probability measures on $\mathbb{R}$. We define weak convergence for finite measures on any topological spaces. In addition, the weak convergence is defined through filters. Thanks to this generalization of weak convergence, we obtain a simple proof of the fact that the Lévy-Prokhorov metric induces the topology of weak convergence. We also formalize important theorems such as Prokhorov's theorem and the Riesz representation theorem. We finally show that the measurable space of all finite measures on a standard Borel space is also a standard Borel space. We give the proof because we could not find any proofs of the statement.

Reference

Our formalization in this chapter is mainly based on the lecture notes by Gaans [54]. The lecture notes include detailed proofs about the Lévy-Prokhorov on probability measures. We extend their definitions and proofs for finite measures. We also refer to the lecture notes by Heil [23, 24], book by Rudin [38], and book by Srivastava [46].

4.1 Characterization of Topology by Filters

There is a characterization of closed sets using convergence with respect to *nets* (Exercise A.48 [24]). We formalize the following characterization of closed sets by limit with respect to filters because nets and filters are equally expressive in terms of convergence (Section 4 [45]).

Lemma 4.1. Let X be a topological space and $C \subseteq X$. Then, the following are equivalent.

1. C is closed in X.
2. For all sets I, proper filters $\mathcal{F}$ on I, $\{x_i\}_{i \in I} \subseteq C$, and $x \in X$ such that $(x_i \longrightarrow x)\ \mathcal{F}$ in X, we have $x \in C$.

If X is first-countable, then these are also equivalent to the following.

3. For all $\{x_n\}_{n \in \mathbb{N}} \subseteq C$, $x \in X$ such that $(x_n \longrightarrow x)\ \mathcal{F}_{\text{seq}}$ in X, we have $x \in C$.

We provide a proof of Exercise A.48 in the lecture notes by Heil [24], where they use limit with respect to nets while we use limit with respect to filters.

Proof Outline. The implication that 1 implies 2 is easily shown from the definition of convergence. To show 1 from 2, it is sufficient to prove $\overline{C} \subseteq C$. Let $x \in \overline{C}$ and $\mathcal{V}_x$ the set of all open neighborhoods of x. Then, we have $C \cap V \neq \varnothing$ for all $V \in \mathcal{V}_x$ from the definition of the closure. Hence, we obtain a sequence $\{x_V\}_{V \in \mathcal{V}_x} \subseteq C$ such that $x_V \in C \cap V$ for all $V \in \mathcal{V}_x$. Let us define a proper filter $\mathcal{F}_x$ on $\mathcal{V}_x$ by

$$\mathcal{F}_x = \bigcup_{U \in \mathcal{V}_x} \{\mathcal{V} \subseteq \mathcal{V}_x \mid \{V \mid V \in\ \mathcal{V}_x \wedge V \subseteq U\} \subseteq \mathcal{V}\}.$$

Then, we have $(x_V \longrightarrow x)\ \mathcal{F}_x$ in X from the definition of $\mathcal{F}_x$ and $\{x_V\}_{V \in \mathcal{V}_x}$. Thus, $x \in C$ follows from the assumption.

The equivalence for the condition 3 is shown by using the axiom of first-countability. □

The last condition of the above equivalence has already been formalized for metric spaces. Since metric spaces are first-countable, our formalization is a relaxed version of the existing one. There is also a characterization of open sets by limit with respect to filters. The characterization is easily derived from that of closed sets.

Lemma 4.2. Let X be a topological space and $U \subseteq X$. Then, the following are equivalent.

1. U is open in X.
2. For all sets I, filters $\mathcal{F}$ on I, $\{x_i\}_{i \in I} \subseteq U$, and $x \in U$ such that $(x_i \longrightarrow x)\ \mathcal{F}$ in X. Then, we have $\forall_{\mathrm{F}}\, i \text{ in } \mathcal{F}.\ x_i \in U$.

If X is first-countable, then these are also equivalent to the following.

3. For all $\{x_n\}_{n \in \mathbb{N}} \subseteq U$, and $x \in U$ such that $(x_n \longrightarrow x)\ \mathcal{F}_{\text{seq}}$ in X. Then, we have $\forall_{\mathrm{F}}\, n \text{ in } \mathcal{F}_{\text{seq}}.\ x_n \in U$ (equivalently, $\exists N.\ \forall n \geq N.\ x_n \in U$).

From the characterization of open sets and closed sets, we obtain a condition to decide whether two topological spaces are equal using filters.

Corollary 4.3. Let $(X, \mathcal{O}_X)$ and $(X, \mathcal{O}'_X)$ be topological spaces. Then, the following are equivalent.

1. $\mathcal{O}_X = \mathcal{O}'_X$.
2. $(x_i \longrightarrow x)\ F$ in $(X, \mathcal{O}_X) \iff (x_i \longrightarrow x)\ F$ in $(X, \mathcal{O}'_X)$ for all $\{x_i\}_{i \in I}$, x, and F.

If both of $(X, \mathcal{O}_X)$ and $(X, \mathcal{O}'_X)$ are first-countable, then these are also equivalent to the following.

3. $(x_n \longrightarrow x)\ \mathcal{F}_{\text{seq}}$ in $(X, \mathcal{O}_X) \iff (x_n \longrightarrow x)\ \mathcal{F}_{\text{seq}}$ in $(X, \mathcal{O}'_X)$ for all $\{x_n\}_{n \in \mathbb{N}}$ and x.

The Characterizations in Isabelle/HOL

We start from constructing the filter $\mathcal{F}_x$ in the proof of Lemma 4.1. In the lecture notes by Heil [24], they define the nets to characterize closed sets by limit with respect to nets. A net uses a directed set as its index, while a sequence in Isabelle/HOL uses a filter as its index. Hence, we first define a function that converts directed sets to filters according to the lecture notes by Shi [45].

```
definition derived-filter :: ['i set, 'i ⇒ 'i ⇒ bool] ⇒ 'i filter where
derived-filter I op ≡ (⨅ i∈I. principal {j∈I. op i j})
```

Given a directed set $(I, \leq_I)$, *derived-filter* I $(\leq_I)$ denotes the filter $\bigcup_{i \in I} \{V \mid \{j \in I \mid i \leq_I j\} \subseteq V\}$. We obtain the filter $\mathcal{F}_x$ by applying the directed set $(\mathcal{V}_x, \supseteq)$ to *derived-filter*.

```
definition nhdsin-sets :: 'a topology ⇒ 'a ⇒ 'a set filter where
nhdsin-sets X x ≡ derived-filter {U. openin X U ∧ x ∈ U} (⊇)
```

The characterization of closed set given by Lemma 4.1 is stated as follows.

```
corollary closedin-iff-limitin-eq:
  fixes X :: 'a topology
  shows closedin X C
    ⟷ C ⊆ topspace X ∧
      (∀ xi x (F :: 'a set filter). (∀ i. xi i ∈ topspace X) ⟶ x ∈ topspace X
          ⟶ (∀F i in F. xi i ∈ C) ⟶ F ≠ ⊥ ⟶ limitin X xi x F ⟶ x ∈ C)
```

Remark 4.4. In Isabelle/HOL, we cannot quantify filters as "for any filter F" due to Isabelle/HOL's type system. For instance, when we want to show $P \longleftrightarrow (\forall F :: \square\ \textit{filter}.\ Q\ F)$, we need to specify some type $\square$ on which filters are defined. We state this lemma by quantifying filters as the type $F :: {}'a\ \textit{set filter}$ when the topology is $X :: {}'a\ \textit{topology}$ because we use the filter $\mathcal{F}_x$ on $\mathcal{V}_x$ to prove the lemma. The characterizations of open sets and topologies (Lemma 4.2 and Corollary 4.3) are also stated by quantifying filters as the type $F :: {}'a\ \textit{set filter}$ for the same reason.

4.2 Lévy-Prokhorov Metric

Historically, Lévy first introduced a metric, known as the Lévy metric, between cumulative distribution functions [32]. Later, Prokhorov defined the Lévy-Prokhorov metric between

finite measures analogous to the Lévy metric [37]. In this section, we review the notion of weak convergence and the Lévy-Prokhorov metric. At the end of this section, we discuss our formalization of the topology of weak convergence and the Lévy-Prokhorov metric. For a measurable space X, $\mathcal{M}_{\mathrm{fin}}(X)$ denotes the set of all finite measures on X. Note that X can be a metric space or a topological space since they both induce a measurable space.

4.2.1 Weak Convergence

We define the notion of weak convergence which treats finite measures on any topological spaces. The convergence in our formalization is defined using filters.

Definition 4.5 (Weak Convergence)**.** Let X be a topological space, I a set, F a filter on I, $\{\mu_i\}_{i\in I} \subseteq \mathcal{M}_{\mathrm{fin}}(X)$, and $\mu \in \mathcal{M}_{\mathrm{fin}}(X)$. We say that $\{\mu_i\}_{i\in I}$ converges weakly to μ with respect to $\mathcal{F}$, denoted by $(\mu_i \Rightarrow_{\mathrm{wc}} \mu)\ \mathcal{F}$, if $\left(\int f\mathrm{d}\mu_i \longrightarrow \int f\mathrm{d}\mu\right)\ \mathcal{F}$ for all $f \in \mathrm{C_b}(X)$, where $\mathrm{C_b}(X)$ is the set of all bounded continuous functions from X to $\mathbb{R}$.

The notion of weak convergence has several equivalent statements when X is a metric space.

Theorem 4.6 (The Portmanteau Theorem)**.** Let X be a metric space, I a set, $\mathcal{F}$ a filter on I, $\{\mu_i\}_{i\in I} \subseteq \mathcal{M}_{\mathrm{fin}}(X)$, and $\mu \in \mathcal{M}_{\mathrm{fin}}(X)$. Then, the following are equivalent.

1. $(\mu_i \Rightarrow_{\mathrm{wc}} \mu)\ \mathcal{F}$.
2. For all $f \in \mathrm{UC_b}(X)$, $\left(\int f\mathrm{d}\mu_i \longrightarrow \int f\mathrm{d}\mu\right)\ \mathcal{F}$.
3. $(\mu_i(X) \longrightarrow \mu(X))\ \mathcal{F}$ and for every closed set C, $\mathrm{Limsup}_{\mathcal{F}}\{\mu_i(C)\}_{i\in I} \leq \mu(C)$.
4. $(\mu_i(X) \longrightarrow \mu(X))\ \mathcal{F}$ and for every open set U, $\mathrm{Liminf}_{\mathcal{F}}\{\mu_i(U)\}_{i\in I} \geq \mu(U)$.
5. For every measurable set $A \in \Sigma_X$ such that $\mu\left(\partial A\right) = 0$, $(\mu_i(A) \longrightarrow \mu(A))\ \mathcal{F}$.

The set $\mathrm{UC_b}(X)$ denotes the set of all bounded uniformly continuous functions $f : X \to \mathbb{R}$.

The Portmanteau theorem is commonly stated for probability measures rather than finite measures. Notice that we require the condition $(\mu_i(X) \longrightarrow \mu(X))\ F$ in 3 and 4. This condition does not appear in the Portmanteau theorem for probability measures. In the proof for probability measures, we use $\mu_i(X) = \mu(X) = 1$. For finite measures, $\mu_i(X)$ is not equal to $\mu(X)$ in general. Hence, we use the condition $(\mu_i(X) \longrightarrow \mu(X))\ \mathcal{F}$ instead of $\mu_i(X) = \mu(X) = 1$ in order to approximate $\mu_i(X)$ to $\mu(X)$ during the proof.

4.2.2 Topology of Weak Convergence

Let X be a topological space. The topology of weak convergence on X, denoted by $\mathcal{O}_{\mathrm{WC}_X}$, is the coarsest topology on $\mathcal{M}_{\mathrm{fin}}(X)$ which makes $(\lambda\mu.\ \int f\mathrm{d}\mu) : \mathcal{M}_{\mathrm{fin}}(X) \to \mathbb{R}$ continuous for all $f \in \mathrm{C_b}(X)$. As the name suggests, convergence in the topology of weak convergence is equal to weak convergence.

Lemma 4.7. Let X be a topological space, I a set, $\mathcal{F}$ a filter on I, $\{\mu_i\}_{i\in I} \subseteq \mathcal{M}_{\mathrm{fin}}(X)$, and $\mu \in \mathcal{M}_{\mathrm{fin}}(X)$. Then,

$$(\mu_i \longrightarrow \mu)\ F \text{ in } (\mathcal{M}_{\mathrm{fin}}(X), \mathcal{O}_{\mathrm{WC}_X}) \iff (\mu_i \Rightarrow_{\mathrm{wc}} \mu)\ F.$$

4.2.3 Lévy-Prokhorov Metric

In the lecture notes by Gaans, they only treat the case when $\mathcal{M}_{\mathrm{fin}}(X)$ is the set of all probability measures on X. We generalize their definitions and proofs to the set of all finite measures.

Definition 4.8 (Lévy-Prokhorov Metric)**.** For a metric space (X, d), the Lévy-Prokhorov metric $d_{\mathcal{M}_{\mathrm{fin}}(X)}$ is a metric on $\mathcal{M}_{\mathrm{fin}}(X)$ defined by

$$d_{\mathcal{M}_{\mathrm{fin}}(X)}(\mu,\nu) = \inf\{\alpha > 0 \mid \forall A \in \Sigma_X.\, \mu(A) \le \nu(A^\alpha) + \alpha \wedge \nu(A) \le \mu(A^\alpha) + \alpha\},$$

where $A^\alpha = \bigcup_{x\in A} ball_X(x,\alpha)$.

Note that $d_{\mathcal{M}_{\mathrm{fin}}(X)}(\mu,\nu) < \infty$ because $\infty \neq \max(\mu(X),\nu(X)) \in \{\alpha > 0 \mid \forall A \in \Sigma_X.\mu(A) \le \nu(A^\alpha) + \alpha \wedge \nu(A) \le \mu(A^\alpha) + \alpha\}$. The Lévy-Prokhorov metric is also expressed using open sets, closed sets, and compact sets.

Lemma 4.9.

$$\begin{aligned} d_{\mathcal{M}_{\mathrm{fin}}(X)}(\mu,\nu) &= \inf\{\alpha > 0 \mid \forall U\text{: open. } \mu(U) \le \nu(U^\alpha) + \alpha \wedge \nu(U) \le \mu(U^\alpha) + \alpha\} \\ &= \inf\{\alpha > 0 \mid \forall C\text{: closed. } \mu(C) \le \nu(C^\alpha) + \alpha \wedge \nu(C) \le \mu(C^\alpha) + \alpha\}. \end{aligned}$$

If X is separable and complete, then

$$d_{\mathcal{M}_{\mathrm{fin}}(X)}(\mu,\nu) = \inf\{\alpha > 0 \mid \forall K\text{: compact. } \mu(K) \le \nu(K^\alpha) + \alpha \wedge \nu(K) \le \mu(K^\alpha) + \alpha\}.$$

The convergence with respect to the Lévy-Prokhorov metric is equivalent to the weak convergence when X is separable.

Theorem 4.10 (Theorem 4.1 and 4.2 [54])**.** The following hold.

1. $(\mathcal{M}_{\mathrm{fin}}(X), d_{\mathcal{M}_{\mathrm{fin}}(X)})$ is a metric space.

Let I be a set, F a filter on I, $\{\mu_i\}_{i\in I} \subseteq \mathcal{M}_{\mathrm{fin}}(X)$ and $\mu \in \mathcal{M}_{\mathrm{fin}}(X)$.

2. $(\mu_i \longrightarrow \mu)\ F$ in $(\mathcal{M}_{\mathrm{fin}}(X), \mathcal{O}_{d_{\mathcal{M}_{\mathrm{fin}}(X)}})$ implies $(\mu_i \Rightarrow_{\mathrm{wc}} \mu)\ F$.
3. If X is separable, $(\mu_i \longrightarrow \mu)\ F$ in $(\mathcal{M}_{\mathrm{fin}}(X), \mathcal{O}_{d_{\mathcal{M}_{\mathrm{fin}}(X)}})$ if and only if $(\mu_i \Rightarrow_{\mathrm{wc}} \mu)\ F$.

The proofs are similar to the one when $\mathcal{M}_{\mathrm{fin}}(X)$ is the set of all probability measures and $F = \mathcal{F}_{\mathrm{seq}}$ (i.e., the sequences are only on $\mathbb{N}$). The Lévy-Prokhorov metric metrizes the topology of weak convergence when X is separable.

Corollary 4.11. If X is separable, the Lévy-Prokhorov metric induces the topology of weak convergence, i.e., $\mathcal{O}_{\mathrm{WC}_X} = \mathcal{O}_{d_{\mathcal{M}_{\mathrm{fin}}(X)}}$.

The generalization by filters of weak convergence and Theorem 4.10 enables us to prove this lemma easily.

Proof. The metrizability is shown from the equivalence of convergences. From Lemma 4.7 and Theorem 4.10, convergences in $(\mathcal{M}_{\mathrm{fin}}(X), \mathcal{O}_{\mathrm{WC}_X})$ and $(\mathcal{M}_{\mathrm{fin}}(X), \mathcal{O}_{d_{\mathcal{M}_{\mathrm{fin}}(X)}})$ are equivalent for all filters. Hence, we have $\mathcal{O}_{\mathrm{WC}_X} = \mathcal{O}_{d_{\mathcal{M}_{\mathrm{fin}}(X)}}$ from Corollary 4.3. □

Even though Corollary 4.11 is a well-known result, only a few books include its proof. We found two books showing Corollary 4.11. In the book by Billingsley [10], they directly prove the equivalence by examining neighborhoods. In the book by Deuschel and Stroock [15], they prove the equivalence by using the equivalence of convergence with respect to the filter $\mathcal{F}_{\text{seq}}$ (i.e., sequences are defined on $\mathbb{N}$ such as $\{\mu_n\}_{n\in\mathbb{N}}$). As we stated in Corollary 4.3, their proof requires the assumption that $(\mathcal{M}_{\text{fin}}(X), \mathcal{O}_{\text{WC}_X})$ is first-countable. They use the fact that $(\mathcal{M}_{\text{fin}}(X), \mathcal{O}_{\text{WC}_X})$ is second-countable (and thus also first-countable) without providing any proof that it is second-countable. If we follow their proof, we will need additional efforts to show the first countability of $(\mathcal{M}_{\text{fin}}(X), \mathcal{O}_{\text{WC}_X})$. In our proof, we do not need the first countability because we generalized the notion of weak convergence and equivalence of convergence by filters.

Thanks to Corollary 4.11, we identify $(\mathcal{M}_{\text{fin}}(X), \mathcal{O}_{d_{\mathcal{M}_{\text{fin}}(X)}})$ with $(\mathcal{M}_{\text{fin}}(X), \mathcal{O}_{\text{WC}_X})$, when X is a separable metric space.

Proposition 4.12 (Proposition 4.4 [54])**.** If X is a separable metric space, then $\mathcal{M}_{\text{fin}}(X)$ is also a separable metric space.

The proof is similar to the one when $\mathcal{M}_{\text{fin}}(X)$ is the set of all probability measures on X. If $\{a_n\}_{n\in\mathbb{N}}$ is a dense subset of X, then

$$\bigcup_{k\in\mathbb{N}} \{r_0\delta_{a_0} + \cdots + r_k\delta_{a_k} \mid r_0, \ldots, r_k \in \mathbb{Q} \cap [0,\infty)\}$$

is a countable dense subset of $\mathcal{M}_{\text{fin}}(X)$, where δ_a denotes the Dirac measure centered at a.

Lévy-Prokhorov Metric in Isabelle/HOL

We explain our implementation of weak convergence and the Lévy-Prokhorov metric.

Weak Convergence We first define the topology of weak convergence by combining existing constants which generate topological spaces. Let f be a bounded continuous function on X and $\mathcal{O}_f$ the least topology on $\mathcal{M}_{\text{fin}}(X)$, which makes $(\lambda N.\ \int x.\ f\ x\ \partial N)$ continuous. Then, $(\mathcal{M}_{\text{fin}}(X), \mathcal{O}_f)$ is written in Isabelle/HOL as follows:

$$(\mathcal{M}_{\text{fin}}(X), \mathcal{O}_f) = \textit{pullback-topology}\ \mathcal{M}_{\text{fin}}(X)\ (\lambda N.\ \int x.\ f\ x\ \partial N)\ \mathbb{R},$$

where

$$\textit{pullback-topology} :: {}'a\ \textit{set} \Rightarrow ({}'a \Rightarrow {}'b) \Rightarrow {}'b\ \textit{topology} \Rightarrow {}'a\ \textit{topology}$$
$$\textit{pullback-topology}\ A\ f\ Y = \text{The least topology on } A \text{ which makes } f\colon A \to Y \text{ continuous.}$$

The set of all open sets $\mathcal{O}_f$ is extracted as follows:

$$\mathcal{O}_f = \textit{Collect}\ (\textit{openin}\ (\textit{pullback-topology}\ \mathcal{M}_{\text{fin}}(X)\ (\lambda N.\ \int x.\ f\ x\ \partial N)\ \mathbb{R})),$$

where

$$\textit{openin} :: {}'a\ \textit{topology} \Rightarrow {}'a\ \textit{set} \Rightarrow \textit{bool}, \quad \textit{openin}\ X\ U \iff U \text{ is an open set of } X.$$
$$\textit{Collect} :: ({}'a \Rightarrow \textit{bool}) \Rightarrow {}'a\ \textit{set}, \quad \textit{Collect}\ P = \{x.\ P\ x\}.$$

Finally, we define the topology of weak convergence $(\mathcal{M}_{\text{fin}}(X), \mathcal{O}[\bigcup_{f\in \text{C}_\text{b}(X)} \mathcal{O}_f])$.

definition *weak-conv-topology* :: *'a topology* ⇒ *'a measure topology* **where**
weak-conv-topology X ≡ *topology-generated-by*
(⋃*f*∈{*f*. *continuous-map X* ℝ *f* ∧ (∃ *B*. ∀ *x*∈*topspace X*. |*f x*| ≤ *B*)} .
Collect (*openin* (*pullback-topology* $\mathcal{M}_{\text{fin}}(X)$ (λ*N*. ∫ *x*. *f x* ∂*N*) ℝ)))

The term *continuous-map X* ℝ *f* means that f is a continuous map from X to $\mathbb{R}$ and *topology-generated-by* receives a set of sets and returns the least topology, including the received set. The topological space *weak-conv-topology X* meets the requirements of the topology of weak convergence.

lemma *continuous-map-weak-conv-topology*:
assumes *continuous-map X* ℝ *f* **and** ⋀*x*. *x* ∈ *topspace X* ⟹ |*f x*| ≤ *B*
shows *continuous-map* (*weak-conv-topology X*) ℝ (λ*N*. ∫ *x*. *f x* ∂*N*)

lemma *weak-conv-topology-minimal*:
assumes *topspace Y* = $\mathcal{M}_{\text{fin}}(X)$
and ⋀*f B*. *continuous-map X* ℝ *f* ⟹ (⋀*x*. *x* ∈ *topspace X* ⟹ |*f x*| ≤ *B*)
⟹ *continuous-map Y* ℝ (λ*N*. ∫ *x*. *f x* ∂*N*)
shows *openin* (*weak-conv-topology X*) *U* ⟹ *openin Y U*

The first lemma guarantees that *weak-conv-topology X* makes (λ*N*. ∫ *x*. *f x* ∂*N*) continuous and the second lemma states that *weak-conv-topology X* is the least topology in such topologies.

From Lemma 4.7, weak convergence and convergence in the topology of weak convergence are equivalent. Thus, we define the notion of weak convergence as an abbreviation for the convergence in the topology of weak convergence. Then, the usual definition of weak convergence (Definition 4.5) is shown as a lemma.

abbreviation *weak-conv-on* :: (*'a* ⇒ *'b measure*) ⇒ *'b measure* ⇒ *'a filter* ⇒ *'b topology* ⇒ *bool*
where *weak-conv-on Ni N F X* ≡ *limitin* (*weak-conv-topology X*) *Ni N F*

lemma *weak-conv-on-def'*:
assumes ⋀*i*. *Ni i* ∈ $\mathcal{M}_{\text{fin}}(X)$ **and** *N* ∈ $\mathcal{M}_{\text{fin}}(X)$
shows *weak-conv-on Ni N F X* ⟷
(∀ *f*. *continuous-map X* ℝ *f* ⟶ (∃ *B*. ∀ *x*∈*topspace X*. |*f x*| ≤ *B*)
⟶ ((λ*i*. ∫ *x*. *f x* ∂*Ni i*) ⟶ (∫ *x*. *f x* ∂*N*)) *F*)

The term *limitin* (*weak-conv-topology X*) *Ni N F* denotes (*Ni* ⟶ *N*) *F* in $(\mathcal{M}_{\text{fin}}(X), \mathcal{O}_{\text{WC}_X})$.

Lévy-Prokhorov Metric To formalize the Lévy-Prokhorov metric in Isabelle/HOL, we use the set-based metric space library. We introduced a new locale *Levy-Prokhorov*, which is logically equivalent to *Metric-space*.

locale *Levy-Prokhorov* = *Metric-space*

Remember that the Lévy-Prokhorov metric is defined as follows.

$$d_{\mathcal{M}_{\text{fin}}(X)}(\mu,\nu) = \inf\{\alpha > 0 \mid \forall A \in \Sigma_X.\, \mu(A) \le \nu(A^\alpha) + \alpha \wedge \nu(A) \le \mu(A^\alpha) + \alpha\},$$
$$\text{where } A^\alpha = \bigcup_{x \in A} \mathit{ball}_X(x,\alpha).$$

Hence, we define the Lévy-Prokhorov metric as follows:

```
definition(in Levy-Prokhorov) 𝒫 ≡ {N. sets N = sets (borel-of mtopology) ∧ finite-measure N}

definition(in Levy-Prokhorov) LPm :: 'a measure ⇒ 'a measure ⇒ real where
LPm N L ≡
  if N ∈ 𝒫 ∧ L ∈ 𝒫 then
     (⨅ {e. e > 0 ∧ (∀ A∈sets (borel-of mtopology).
                          measure N A ≤ measure L (⋃a∈A. mball a e) + e ∧
                          measure L A ≤ measure N (⋃a∈A. mball a e) + e)})
  else 0
```

Notice that *LPm* returns *0* when one of the arguments is not a member of $\mathcal{P}$ because the set to which we apply infimum might be empty when *LPm* receives an infinite measure. In Isabelle/HOL, the infimum operator on real numbers does not return ∞ nor any specific value when applied to the empty set; i.e., the value of $\bigsqcap \varnothing$ is unknown. This is a problem because *LPm* needs to be a non-negative function on the whole type due to the definition of *Metric-space*.

We then prove that $(\mathcal{P}, LPm)$ is a metric space.

```
sublocale Levy-Prokhorov ⊆ LPm: Metric-space 𝒫 LPm
```

The reader might wonder why we define a new locale *Levy-Prokhorov*, which is logically equivalent to *Metric-space*, rather than using *Metric-space* directly. If we try to define the Lévy-Prokhorov metric in the context of *Metric-space* without introducing a new locale, it does not work.

```
context Metric-space
begin

definition 𝒫 ≡ {N. sets N = sets (borel-of mtopology) ∧ finite-measure N}
definition LPm ≡ ...

sublocale LPm: Metric-space 𝒫 LPm

end
```

The problem is that we try to instantiate *Metric-space* inside the context of *Metric-space*. This causes Isabelle to build an infinite chain; thus, Isabelle does not terminate. This workaround is explained in the Isabelle tutorial on locales [6].

4.3 Prokhorov's Theorem

One of the important results related to the Lévy-Prokhorov metric is Prokhorov's theorem. In a typical situation in probability theory or statistics, one may want to know whether a sequence of measures has a limit or at least has a converging subsequence. Prokhorov's theorem is applied to prove the existence of a converging subsequence. The theorem is used in proofs for various important results such as the central limit theorem, Sanov's theorem, and the existence of *optimal coupling*. The central limit theorem and Sanov's theorem are key concepts in probability theory. The central limit theorem states that under appropriate conditions, the distribution of normalized sample means converges weakly to the standard

normal distribution. Sanov's theorem is an important result in the large deviation theory (e.g. Section 3.2 [15]). The theorem describes the asymptotic behavior of atypical samples and gives evidence why we use the relative entropy (Kullback-Leibler divergence) to evaluate estimated distributions. Both the central limit theorem and Sanov's theorem use Prokhorov's theorem. In transportation theory, a *coupling* is a *plan* how to move resources from supply areas to demand areas. A coupling is represented as a measure satisfying certain conditions. An optimal coupling is a coupling that minimizes the total *cost* of transporting resources. In the proof of the existence of an optimal coupling, Prokhorov's theorem is essential [55, 57].

In this section, we discuss Prokhorov's theorem and related topics.

4.3.1 Regular Measures

We define the notion of regular measures and tightness of measures. The regularity of measures gives ways to approximate a measured value $\mu(A)$ by open sets, closed sets, and compact sets. The tightness of measures is used to express a condition in Prokhorov's theorem.

Definition 4.13. Let X be a topological space. A measure μ on X is called:

1. *inner regular* if $\mu(A) = \sup\{\mu(C) \mid C \subseteq A,\ C \text{ is closed}\}$ for all measurable sets A,
2. *outer regular* if $\mu(A) = \inf\{\mu(U) \mid A \subseteq U,\ U \text{ is open}\}$ for all measurable sets A, and
3. *regular* if μ is inner regular and outer regular.

Proposition 4.14. Let X be a metrizable space. Then, any finite measure on X is regular.

Remark 4.15. This definition of inner regularity by Gaans is different from the standard definition. In general, a measure μ on X is called inner regular if

1'. $\mu(A) = \sup\{\mu(K) \mid K \subseteq A,\ K \text{ is compact}\}$ for all measurable sets A.

This definition is stronger than the condition 1 in Definition 4.13, when every compact set is closed (e.g. when X is metrizable). As we will see soon, Proposition 4.14 still holds even if we use the condition 1' as inner regularity when X is a Polish space (Corollary 4.19).

Proposition 4.14 has already been included in the standard Isabelle/HOL's library. They assume that X is a Polish space and use the condition 1' as the definition of inner regular. Their formalization is restricted to measures on the Borel space of topological spaces on type classes; thus, they treat only when X is the universal set such as $\mathbb{R}$. We formalize the general result when X is an arbitrary metrizable space or a Polish space.

Next, we define tightness.

Definition 4.16 (Tightness)**.** Let X be a topological space and $\Gamma \subseteq \mathcal{M}_{\text{fin}}(X)$. We call Γ *tight* if for every $\varepsilon > 0$, there exists a compact set K of X such that $\mu(X - K) \leq \varepsilon$ for all $\mu \in \Gamma$. A measure μ on X is tight if $\{\mu\}$ is tight.

The existing definition of tightness in Isabelle/HOL's library is restricted to when Γ is a sequence on $\mathbb{N}$ of probability measures on $\mathbb{R}$.

Lemma 4.17. If X is metrizable and μ is a tight measure on X, then $\mu(A) = \sup\{\mu(K) \mid K \subseteq A,\ K \text{ is compact}\}$ for all measurable sets A.

Theorem 4.18. If X is a Polish space, then any finite measure on X is tight.

Corollary 4.19. If X is a Polish space and μ is a finite measure on X, then $\mu(A) = \sup\{\mu(K) \mid K \subseteq A,\ K \text{ is compact}\}$ for all measurable sets A.

4.3.2 Prokhorov's Theorem

We formalize Prokhorov's theorem. Let $\mathcal{M}_{\mathrm{fin}\leq r}(X) = \mathcal{M}_{\mathrm{fin}}(X) \cap \{\mu \mid \mu(X) \leq r\}$ for $r \geq 0$.

Theorem 4.20 (Prokhorov's Theorem)**.** Let X be a Polish space and $\Gamma \subseteq \mathcal{M}_{\mathrm{fin}\leq r}(X)$ for some $r \geq 0$. Then, the following are equivalent.

1. Γ is relatively compact.
2. Γ is tight.

Remark 4.21. Actually, the assumption $\Gamma \subseteq \mathcal{M}_{\mathrm{fin}\leq r}(X)$ is relaxed to $\Gamma \subseteq \mathcal{M}_{\mathrm{fin}}(X)$ in the proof that 1 implies 2. The completeness assumption is not required in the proof that 2 implies 1.

The following corollary is applied to show the existence of a converging subsequence.

Corollary 4.22. Let X be a separable metrizable space and $\{\mu_n\}_{n\in\mathbb{N}} \subseteq \mathcal{M}_{\mathrm{fin}\leq r}(X)$ for some $r \geq 0$. If $\{\mu_n\}_{n\in\mathbb{N}}$ is tight, then there exists a subsequence $\{\mu_{n_k}\}_{k\in\mathbb{N}}$ and $\mu \in \mathcal{M}_{\mathrm{fin}\leq r}(X)$ such that $(\mu_{n_k} \Rightarrow_{\mathrm{wc}} \mu)$ $\mathcal{F}_{\mathrm{seq}}$.

Avigad et al. formalized the above corollary when $\{\mu_n\}_{n\in\mathbb{N}}$ is a sequence of probability measures on $\mathbb{R}$ and applied it to prove the central limit theorem [4]. In the case of probability measures on $\mathbb{R}$, there is a simpler proof using Helly's selection theorem. In the general case, we need to prove in another way because the proof using Helly's selection theorem uses cumulative distribution function; i.e., X needs to be $\mathbb{R}$.

The proof that 1 implies 2 in Prokhorov's theorem is more straightforward. The proof that 2 implies 1 requires more effort to prove for us. We do not discuss the details of the proof. Instead, we explain a key lemma for the proof that 2 implies 1.

Lemma 4.23. If X is a compact metric space, then $\mathcal{M}_{\mathrm{fin}\leq r}$ is compact.

Once this lemma is proved, the direction that 2 implies 1 of Prokhorov's theorem is derived using the isomorphism between X and a subset of $\mathscr{H}$ (Lemma 3.5).

Proof of Lemma 4.23. The idea of the proof is to make a homeomorphism between $\mathcal{M}_{\mathrm{fin}\leq r}$ and a compact space. Let Φ be

$$\Phi = \left(\mathbb{R}^{\mathrm{C}(X)}\right) \cap \{\varphi \mid \varphi \text{ is a positive linear functional} \wedge \varphi(1) \leq r\}. \tag{4.1}$$

Remember that an element φ of $\mathbb{R}^{\mathrm{C}(X)}$ is a function $\varphi : \mathrm{C}(X) \to \mathbb{R}$. We denote $\varphi(f)$ by φ_f. Then, the linearity of $\varphi \in \Phi$ means that for all $\alpha, \beta \in \mathbb{R}$ and $f, g \in \mathrm{C}(X)$, $\varphi_{\alpha f+\beta g} = \alpha\varphi_f + \beta\varphi_g$. The positiveness means that for all $f \in \mathrm{C}(X)$ such that $f \geq 0$, $\varphi_f \geq 0$.

We assume that Φ is equipped with the subspace topology of the product topology $\mathbb{R}^{\mathrm{C}(X)}$ (subspace topology of the *weak* topology*). We define the function T from $\mathcal{M}_{\mathrm{fin}\leq r}$ to Φ by $T(\mu)_f = \int f \mathrm{d}\mu$. It is easy to check that $T(\mu) \in \Phi$, and T is a sequential homeomorphic map. For instance, the linearity of the integral implies the linearity of $T(\mu)$. The function T is bijective by the Riesz representation theorem (Corollary 4.31). As Gaans stated, Φ is

metrizable[1]. Thus, T is a homeomorphism[2]. Furthermore, Φ is compact by the special case of Alaoglu's theorem (Theorem 4.28). Hence, $\mathcal{M}_{\text{fin}\leq r}$ is compact. □

Remark 4.24. In the lecture notes, Gaans stated that the sequential compactness of a closed subset of Φ follows from its compactness. This statement is true because Φ is metrizable. However, they did not mention that in their proof.

Prokhorov's theorem is applied to prove the completeness of the Lévy-Prokhorov metric.

Corollary 4.25. If X is separable and complete, then $(\mathcal{M}_{\text{fin}}(X), d_{\mathcal{M}_{\text{fin}}(X)})$ is complete.

When we prove the existence of a limit of a Cauchy sequence $\{\mu_n\}_{n\in\mathbb{N}} \subseteq \mathcal{M}_{\text{fin}}(X)$, we use Prokhorov's theorem as $\Gamma = \{\mu_n\}_{n\in\mathbb{N}}$. Hence, we need to show that $\{\mu_n\}_{n\in\mathbb{N}} \subseteq \Gamma_r$ for some $r \geq 0$. This follows from the fact that $\{\mu_n\}_{n\in\mathbb{N}}$ is a Cauchy sequence.

As a consequence of Corollary 4.11, Proposition 4.12, and Corollary 4.25, we have the following.

Corollary 4.26. If X is a Polish space, then so is $\mathcal{M}_{\text{fin}}(X)$.

4.3.3 Alaoglu's Theorem

Alaoglu's theorem (sometimes called the Banach-Alaoglu theorem) is an important result in functional analysis. The theorem states that the closed unit ball of the dual space of a normed vector space is compact. Let Y be a vector space over $\mathbb{R}$ and Y^* the dual space of Y. The *weak* topology* is a topology on Y^*, which is the coarsest topology that makes every $(\lambda f.\, f(y)) : Y^* \to \mathbb{R}$ continuous. The original statement of Alaoglu's theorem is the following.

Theorem 4.27 (Alaoglu's Theorem)**.** Let Y be a normed vector space and $B^* = \{\varphi \in Y^* \mid \|\varphi\| \leq r\}$. Then, B^* is compact in Y^* with respect to the weak* topology.

We do not prove the above form of the theorem due to the lack of a set-based vector space library in Isabelle/HOL. Instead, we prove a special case of Alaoglu's theorem for our use.

Theorem 4.28. If a topological space X is compact, then Φ defined by (4.1) in the proof of Lemma 4.23 is compact.

Remark 4.29. While the Alaoglu's theorem says that $\{\varphi \in C(X)^* \mid \|\varphi\| \leq r\}$ is compact, Theorem 4.28 states that $\Phi = \{\varphi \in C(X)^* \mid \|\varphi\| \leq r \wedge \varphi \text{ is positive}\}$ is compact. Note that $\|\varphi\| = \varphi(1)$ when $\varphi \in C(X)^*$ is positive.

Proof Outline. We formalize the theorem following the proof in the lecture notes by Heil [23]. The proof is simple. We first observe that $\prod_{f\in \mathrm{C}(X)} [-r\|f\|, r\|f\|]$ is compact in $\mathbb{R}^{\mathrm{C}(X)}$ by Tychonoff's theorem. Note that every $f \in \mathrm{C}(X)$ is bounded because X is compact. We then show that $\Phi \subseteq \prod_{f\in \mathrm{C}(X)}[-r\|f\|, r\|f\|]$ and Φ is closed. The fact that Φ is closed is shown by the characterization of closed sets by limit (Lemma 4.1). □

[1]Since X is compact, $\mathrm{C}(X)$ along with the topology of uniform convergence is separable (Theorem 2.4.3 [46]). Let $\{g_n\}_{n\in\mathbb{N}}$ be a dense subset of $\mathrm{C}(X)$. Then, the metric on Φ is, for instance, given by

$$d(\varphi, \psi) = \sum_{n=0}^{\infty} \frac{1}{2^{n+1}} \min(1, |\varphi(g_n) - \psi(g_n)|).$$

[2]A function f from a first-countable space is continuous if and only if it is sequentially continuous.

4.3.4 Riesz Representation Theorem

The Riesz representation theorem (sometimes called the Riesz-Markov representation theorem or Riesz-Markov-Kakutani representation theorem) states that a real-valued (or complex-valued) positive linear functional is represented by the Lebesgue integration with respect to a unique measure. We prove the Riesz representation theorem following the book by Rudin [38].

Theorem 4.30 (The Riesz representation theorem)**.** Let X be a locally compact Hausdorff space and φ a real-valued positive linear functional on $C_{\mathrm{c}}(X)$, where $C_{\mathrm{c}}(X)$ is the set of all continuous functions on X whose closed support is compact. Then, there exists a σ-algebra $\mathcal{N}$ in X and a unique measure μ on $(X, \mathcal{N})$ such that:

- $\varphi(f) = \int f \mathrm{d}\mu$ for all $f \in C_{\mathrm{c}}(X)$,
- $\Sigma_X \subseteq \mathcal{N}$,
- $\mu(K) < \infty$ for all compact sets K,
- $\mu(A) = \inf\{\mu(U) \mid A \subseteq U,\ U \text{ is open}\}$ for all $A \in \mathcal{N}$,
- $\mu(A) = \sup\{\mu(K) \mid K \subseteq A,\ K \text{ is compact}\}$ for all open sets A and for all $A \in \mathcal{N}$ such that $\mu(A) < \infty$, and
- μ is a complete measure, i.e., if $E \in \mathcal{N}$, $A \subseteq E$, and $\mu(E) = 0$, then $A \in \mathcal{N}$.

In the book, the proof of the Riesz representation theorem is divided into ten steps and uses two lemmas. Their proofs consist of around nine pages, whereas we spent more than 2,100 lines for their proofs. The proof requires Urysohn's lemma on locally compact Hausdorff spaces. Although Isabelle/HOL's library has several forms of Urysohn's lemmas and lemmas related to locally compact spaces, the library does not include Urysohn's lemma on locally compact Hausdorff spaces. Hence, we formalized the lemma by ourselves.

We use the following corollary in the proof of Prokhorov's theorem.

Corollary 4.31. Let X be a compact metric space and φ be a real-valued positive linear functional on $\mathrm{C}(X)$. Then, there exists a unique measure μ on X such that for all $f \in \mathrm{C}(X)$,

$$\varphi(f) = \int f \mathrm{d}\mu.$$

Prokhorov's Theorem in Isabelle/HOL

As we discussed, the proof of Lemma 4.23 relies on results from vector space theory such as Alaoglu's theorem and the Riesz representation theorem. Although these theorems need to be stated in set-based vector spaces in Isabelle/HOL for our use, most of Isabelle/HOL's vector space library is based on type classes. The set-based vector space library by Lee [31] includes only basic definitions. Thiemann and Yamada also formalized a set-based vector space [51]. However, their work treats only finite-dimensional spaces. Since we are interested in the Lévy-Prokhorov metric rather than vector space theory, we leave the development of the set-based vector space library for future work. Thus, we formalize positive linear functionals used in proofs and their properties without mentioning vector spaces.

definition *has-compact-support-on* :: ('*a* ⇒ '*b* :: *monoid-add*) ⇒ '*a topology* ⇒ *bool*
(**infix** *has'-compact'-support'-on 60*) **where**
f has-compact-support-on X ⟷ *compactin X* (*X closure-of support-on* (*topspace X*) *f*)

definition *positive-linear-functional-on-CX* :: '*a topology* ⇒ (('*a* ⇒ '*b* :: {*ring*, *order*, *topological-space*}) ⇒ '*b*) ⇒ *bool*
where *positive-linear-functional-on-CX X φ* ≡
(∀*f*. *continuous-map X euclidean f* ⟶ *f has-compact-support-on X*
⟶ (∀*x*∈*topspace X*. *f x* ≥ *0*) ⟶ *φ* (λ*x*∈*topspace X*. *f x*) ≥ *0*) ∧
(∀*f a*. *continuous-map X euclidean f* ⟶ *f has-compact-support-on X*
⟶ *φ* (λ*x*∈*topspace X*. *a* ∗ *f x*) = *a* ∗ *φ* (λ*x*∈*topspace X*. *f x*)) ∧
(∀*f g*. *continuous-map X euclidean f* ⟶ *f has-compact-support-on X*
⟶ *continuous-map X euclidean g* ⟶ *g has-compact-support-on X*
⟶ *φ* (λ*x*∈*topspace X*. *f x* + *g x*) = *φ* (λ*x*∈*topspace X*. *f x*) + *φ* (λ*x*∈*topspace X*. *g x*))

The special case of Alaoglu's theorem (Theorem 4.28), the Riesz representation theorem (Theorem 4.30), and Prokhorov's theorem (Theorem 4.20) are stated as follows.

theorem *Alaoglu-theorem-real-functional*:
fixes *X* :: '*a topology* **and** *r* :: *real*
defines *prod-space* ≡ $\mathbb{R}^{C(X)}$
defines *B* ≡ {*φ*∈*topspace prod-space*. *φ* (λ*x*∈*topspace X*. *1*) ≤ *r* ∧ *positive-linear-functional-on-CX X φ*}
assumes *compact*: *compact-space X* **and** *ne*: *topspace X* ≠ {}
shows *compactin prod-space B*

theorem *Riesz-representation-real-complete*:
fixes *X* :: '*a topology* **and** *φ* :: ('*a* ⇒ *real*) ⇒ *real*
assumes *locally-compact-space X* **and** *Hausdorff-space X*
and *positive-linear-functional-on-CX X φ*
shows ∃*M*. ∃!*N*. *sets N* = *M* ∧ *subalgebra N* (*borel-of X*)
∧ (∀*A*∈*sets N*. *emeasure N A* = (∏ *C*∈{*C*. *openin X C* ∧ *A* ⊆ *C*}. *emeasure N C*))
∧ (∀*A*. *openin X A*
⟶ *emeasure N A* = (⨆ *K*∈{*K*. *compactin X K* ∧ *K* ⊆ *A*}. *emeasure N K*))
∧ (∀*A*∈*sets N*. *emeasure N A* < ∞
⟶ *emeasure N A* = (⨆ *K*∈{*K*. *compactin X K* ∧ *K* ⊆ *A*}. *emeasure N K*))
∧ (∀*K*. *compactin X K* ⟶ *emeasure N K* < ∞)
∧ (∀*f*. *continuous-map X* $\mathbb{R}$ *f* ⟶ *f has-compact-support-on X*
⟶ *φ* (λ*x*∈*topspace X*. *f x*) = (∫ *x*. *f x* ∂*N*))
∧ (∀*f*. *continuous-map X* $\mathbb{R}$ *f* ⟶ *f has-compact-support-on X* ⟶ *integrable N f*)
∧ *complete-measure N*

lemma *Prokhorov*:
assumes *Polish-space X*
and Γ ⊆ {*N*. *N* (*space N*) ≤ *ennreal r* ∧ *sets N* = *sets* (*borel-of X*)}
shows *tight-on-set X* Γ
⟷ *compactin* (*weak-conv-topology X*) (*weak-conv-topology X closure-of* Γ)

4.4 Space of Finite Measures

In this section, we discuss the measurable space of all finite measures. In stochastic processes, measures are usually indexed by time or states. A stochastic process is interpreted as a measurable function from its index set to the space of measures. In the semantics of probabilistic programs, the Giry monad [20] G (or sub-Giry monad) gives a standard semantics of probabilistic programs where $G(M)$ is the measurable space of all probability measures on M defined independently from metric or topology.

We will show that this type of measurable space of all finite measures is generated from the topology of weak convergence when the underlying topological space is a Polish space.

Definition 4.32. Let M be a measurable space. The space of finite measures on M is denoted by $(\mathcal{M}_{\text{fin}}(M), \Sigma_{\mathcal{M}_{\text{fin}}(M)})$, where $\Sigma_{\mathcal{M}_{\text{fin}}(M)}$ is the least σ-algebra that makes $(\lambda\mu.\, \mu(A))$ measurable for all $A \in \Sigma_M$.

Note that this definition does not use any metric or topology. In Isabelle/HOL's library, the space of all sub-probability measures $\mathcal{M}_{\text{sprob}}(M)$ and the space of all probability measures $\mathcal{M}_{\text{prob}}(M)$ are already formalized by Eberl et al. [18] (*subprob-algebra* M and *prob-algebra* M, respectively). We have formalized the space of all finite measures in the same way as *subprob-algebra*. Subsequently, we have shown that $\mathcal{M}_{\text{sprob}}(M)$ and $\mathcal{M}_{\text{prob}}(M)$ are subspaces of $\mathcal{M}_{\text{fin}}(M)$.

The following lemma follows immediately from the Portmanteau theorem[3].

Lemma 4.33 (Corollary 17.21 [29])**.** For an open set $U \subseteq X$, $(\lambda\mu.\mu(U)) : (\mathcal{M}_{\text{fin}}(X), \mathcal{O}_{d_{\mathcal{M}_{\text{fin}}(X)}}) \to \mathbb{R}$ is lower semi-continuous. For a closed set $C \subseteq X$, $(\lambda\mu.\, \mu(C)) : (\mathcal{M}_{\text{fin}}(X), \mathcal{O}_{d_{\mathcal{M}_{\text{fin}}(X)}}) \to \mathbb{R}$ is upper semi-continuous.

Corollary 4.34. $\Sigma_{\mathcal{M}_{\text{fin}}(X)} \subseteq \Sigma_{d_{\mathcal{M}_{\text{fin}}(X)}}$.

Proof. From the definition of $\Sigma_{\mathcal{M}_{\text{fin}}(X)}$, it is sufficient to show that for all $A \in \Sigma_X$, $(\lambda\mu.\, \mu(A))$ is a measurable function from $(\mathcal{M}_{\text{fin}}(X), \Sigma_{d_{\mathcal{M}_{\text{fin}}(X)}})$ to $\mathbb{R}$. It is easy to check the measurability because by Lemma 4.33, $(\lambda\mu.\, \mu(U)) : (\mathcal{M}_{\text{fin}}(X), d_{\mathcal{M}_{\text{fin}}(X)}) \to \mathbb{R}$ is lower semi-continuous for all open sets $U \subseteq X$, hence measurable. □

The inverse inclusion holds when X is separable and complete.

Theorem 4.35. If a metric space X is separable and complete, then $\Sigma_{\mathcal{M}_{\text{fin}}(X)} = \Sigma_{d_{\mathcal{M}_{\text{fin}}(X)}}$.

Corollary 4.36. If X is a Polish space, then $\Sigma_{\mathcal{M}_{\text{fin}}(X)} = \Sigma_{(\mathcal{M}_{\text{fin}}(X), \mathcal{O}_{\text{WC}_X})}$.

We constructed the proof of Theorem 4.35 by ourselves because we could not find any proof for the statement. We provide an informal proof here.

Proof of Theorem 4.35. Since $\Sigma_{d_{\mathcal{M}_{\text{fin}}(X)}}$ is generated from closed balls, it is sufficient to prove that every closed ball is a member of $\Sigma_{\mathcal{M}_{\text{fin}}(X)}$. Let μ be a finite measure on X and $\varepsilon \geq 0$. Our goal is to show that $cBall_{\mathcal{M}_{\text{fin}}(X)}(\mu, \varepsilon) \in \Sigma_{\mathcal{M}_{\text{fin}}(X)}$. Let $\mathcal{O}_b$ be a countable base of X and $\mathcal{O}_{b\text{fU}}$ the set of all finite unions of elements of $\mathcal{O}_b$. Then, $\mathcal{O}_{b\text{fU}}$ is also countable.

[3]Remember that for a first-countable space X,

- $f : X \to \mathbb{R}$ is lower semi-continuous iff $(x_n \longrightarrow x)\ \mathcal{F}_{\text{seq}}$ in X implies $f(x) \leq \text{Liminf}_{\mathcal{F}_{\text{seq}}}\{f(x_n)\}_{n\in\mathbb{N}}$.
- $f : X \to \mathbb{R}$ is upper semi-continuous iff $(x_n \longrightarrow x)\ \mathcal{F}_{\text{seq}}$ in X implies $f(x) \geq \text{Limsup}_{\mathcal{F}_{\text{seq}}}\{f(x_n)\}_{n\in\mathbb{N}}$.

Claim 4.37.

$$cBall_{\mathcal{M}_{\mathrm{fin}}(X)}(\mu,\varepsilon) = \bigcap_{U\in\mathcal{O}_{bfU}} \left(\bigcap_{n\in\mathbb{N}} (\lambda\nu.\,\nu(U))^{-1}\left(-\infty, \mu\left(U^{\left(\varepsilon+\frac{1}{1+n}\right)}\right) + \varepsilon + \frac{1}{1+n}\right] \cap \left(\lambda\nu.\,\nu\left(U^{\left(\varepsilon+\frac{1}{1+n}\right)}\right)\right)^{-1}\left[\mu(U) - \left(\varepsilon+\frac{1}{1+n}\right), \infty\right)\right) \tag{4.2}$$

If the above claim is shown, $cBall(\mu,\varepsilon) \in \Sigma_{\mathcal{M}_{\mathrm{fin}}(X)}$ follows from the definition of $\Sigma_{\mathcal{M}_{\mathrm{fin}}(X)}$.

The inclusion $\subseteq$ in Equation (4.2) is directly proven by unfolding the definition of the Lévy-Prokhorov metric. Hence, we show $\supseteq$ of (4.2). Let us assume that ν is a member of the right hand side of (4.2). Then, for all $U \in \mathcal{O}_{bfU}$ and $n \in \mathbb{N}$, we have

$$\nu(U) \leq \mu\left(U^{\left(\varepsilon+\frac{1}{1+n}\right)}\right) + \varepsilon + \frac{1}{1+n}, \quad \mu(U) \leq \nu\left(U^{\left(\varepsilon+\frac{1}{1+n}\right)}\right) + \varepsilon + \frac{1}{1+n}. \tag{4.3}$$

We show $\nu \in cBall_{\mathcal{M}_{\mathrm{fin}}(X)}(\mu,\varepsilon)$ by proving that $d_{\mathcal{M}_{\mathrm{fin}}(X)}(\mu,\nu) < \varepsilon'$ for all $\varepsilon' > \varepsilon$. Let $\varepsilon' > \varepsilon$, then there exists $n \in \mathbb{N}$ such that $\varepsilon + \frac{1}{1+n} < \varepsilon'$. For an open set $A \subseteq X$, we have

$$\begin{aligned}
\mu(A) &= \sup\{\mu(K) \mid K \subseteq A,\ K \text{ is compact}\} \ \text{(Corollary 4.19)} \\
&\leq \sup\{\mu(U) \mid U \subseteq A,\ U \in \mathcal{O}_{bfU}\} \\
&\leq \sup\left\{\nu\left(U^{\left(\varepsilon+\frac{1}{1+n}\right)}\right) + \varepsilon + \frac{1}{1+n} \mid U \subseteq A,\ U \in \mathcal{O}_{bfU}\right\} \ \text{(by (4.3))} \\
&\leq \nu\left(A^{\left(\varepsilon+\frac{1}{1+n}\right)}\right) + \varepsilon + \frac{1}{1+n}.
\end{aligned} \tag{4.4}$$

The inequality (4.4) above is shown as follows: Since $\mathcal{O}_b$ is a base of X, there exists $\mathcal{O}' \subseteq \mathcal{O}_b$ such that $A = \bigcup_{U\in\mathcal{O}'} U$. If $K \subseteq A$ is compact, there exists a finite subset $\mathcal{O}'_{\mathrm{fin}} \subseteq \mathcal{O}'$ such that $K \subseteq \bigcup_{U\in\mathcal{O}'_{\mathrm{fin}}} U$. By the definition of $\mathcal{O}_{bfU}$, we have $\bigcup_{U\in\mathcal{O}'_{\mathrm{fin}}} U \in \mathcal{O}_{bfU}$. Thus, (4.4) holds.

Similarly, we have $\nu(A) \leq \mu\left(A^{\left(\varepsilon+\frac{1}{1+n}\right)}\right) + \varepsilon + \frac{1}{1+n}$ for all open sets $A \subseteq X$. Hence,

$$\begin{aligned}
d_{\mathcal{M}_{\mathrm{fin}}(X)}(\mu,\nu) &= \inf\{\alpha > 0 \mid \forall A\text{: open. } \mu(A) \leq \nu(A^\alpha) + \alpha \wedge \nu(A) \leq \mu(A^\alpha) + \alpha\} \\
&\leq \varepsilon + \frac{1}{1+n} < \varepsilon'.
\end{aligned}$$

□

Corollary 4.36 is applied to prove that the space of finite measures is a standard Borel space, which is a measurable space generated from a Polish space. As a consequence of Corollary 4.26 and Corollary 4.36, we obtain the following.

Corollary 4.38. If M is a standard Borel space, then so is $\mathcal{M}_{\mathrm{fin}}(M)$.

Corollary 4.39. If M is a standard Borel space, then $\mathcal{M}_{\mathrm{sprob}}(M)$ and $\mathcal{M}_{\mathrm{prob}}(M)$ are also standard Borel spaces.

Chapter 5

Quasi-Borel Spaces

The theory of quasi-Borel spaces was introduced by Heunen et al. [25] in order to give a denotational semantics for higher-order probabilistic programs. The theory is a suitable model for the semantics of higher-order probabilistic programs because of the following reasons.

- Function spaces with desired properties always exist. In other words, the category of quasi-Borel spaces **QBS** is Cartesian closed. The function spaces are necessary in the semantics of higher-order programs. The existence of function spaces is the reason why we use quasi-Borel spaces instead of measure theory. Because measurable spaces do not have function spaces in general.
- *Measures* are defined on quasi-Borel spaces. Especially, any s-finite measures on a standard Borel space is represented as a *measure* on the corresponding quasi-Borel spaces. In addition, *measurability* and *integral* are equal to those in measure theory when we consider only standard Borel spaces.
- There is the s-finite measure monad on quasi-Borel spaces. The monad gives a semantics of probabilistic programs with conditioning.

In this chapter, we first formalize s-finite measures and s-finite kernels, which are the basis of a semantics of first-order probabilistic programs. Then, we construct quasi-Borel spaces and the s-finite measure monad using standard Borel spaces and s-finite kernels. We also discuss a proof automation of our quasi-Borel spaces library.

Contributions

This is the first formalization of the theory of quasi-Borel spaces, to the best of our knowledge. We construct quasi-Borel spaces and the s-finite measure monad on them. Although the construction of the s-finite measure monad is non-trivial, no paper includes its details. In addition, the details of the definition vary among previous studies [44, 52, 56]. We recover the omitted details during formalization. We define the basic spaces, e.g., product spaces and function spaces, and construct the list spaces using the isomorphism $List(X) \cong \coprod_{n=0}^{\infty} \prod_{0 \leq k < n} X$. The probability monad is also included in our formalization, which is obtained by taking the subspace of the s-finite measure monad. For usability, we implement a proof automation for quasi-Borel spaces in ML. In the usual definition, the function spaces are defined through morphisms (structure-preserving functions) between quasi-Borel spaces. In our definition, we

first construct the function spaces, then define morphisms. This design works better with our proof automation.

Reference

The definition of s-finite kernels is based on the work by Staton [47]. We refer to the paper by Heunen et al. [25] for the definition and standard properties of quasi-Borel spaces and the lecture notes by Yang [56] for the s-finite measure monad on quasi-Borel spaces.

5.1 s-Finite Measures and s-Finite Kernels

We formalize the s-finite measures and s-finite kernels which are used for a semantic model of first-order probabilistic programs with soft conditioning.

5.1.1 s-Finite Measures

First, we define the s-finite measure.

Definition 5.1. A measure is called an *s-finite measure* if it is represented as a countable sum of finite measures.

From the definition, all σ-finite measures, such as the Lebesgue measure, are also s-finite measures. Note that there are s-finite measures which are not σ-finite. For instance, the measure μ on $\{0\}$ defined by $\mu(\{0\}) = \infty$ is s-finite since $\mu = \sum_{n=0}^{\infty} \delta_0$, but not σ-finite.

One of the important theorems is a restricted Fubini-Tonelli theorem.

Theorem 5.2 (Restricted Fubini-Tonelli Theorem)**.** Let M and N be measurable spaces, and μ and ν s-finite measures on M and N, respectively.

- If $f : M \times N \to \overline{\mathbb{R}}_{\geq 0}$ is measurable, then
$$\int \int f(x,y)\mu(\mathrm{d}x)\nu(\mathrm{d}y) = \int \int f(x,y)\nu(\mathrm{d}y)\mu(\mathrm{d}x)$$
- If $f : M \times N \to \mathbb{R}$ is measurable and $\int \int |f(x,y)|\mu(\mathrm{d}x)\nu(\mathrm{d}y) < \infty$, (or equivalently, $\int \int |f(x,y)|\nu(\mathrm{d}y)\mu(\mathrm{d}x) < \infty$) then the above equation holds for f.

Note that we do not have the equation for the (binary) product measure $\mu \otimes \nu$ because product measures of s-finite measures are not determined uniquely in general. However, it is possible to define a product measure of s-finite measures and show the Fubini-Tonelli theorem. We will explain this point in the implementation part.

5.1.2 s-Finite Kernels

Next, we define s-finite kernels and related notions. Intuitively, s-finite kernels are generalization of probabilistic processes that return s-finite measures. Although the class of kernels is not closed under composition in general, the class of s-finite kernels is closed.

Definition 5.3. Let M and N be measurable spaces and $\kappa : M \times \Sigma_N \to \overline{\mathbb{R}}_{\geq 0}$. The function κ is called a *measure kernel* from M to N, denoted by $\kappa : M \rightsquigarrow N$, if

- for all $x \in M$, κ_x is a measure on N, and
- for all $A \in \Sigma_N$, $(\lambda x.\, \kappa_x(A))$ is measurable.

Furthermore, a measure kernel $\kappa : M \rightsquigarrow N$ is called

- a *probability kernel* if κ_x is a probability measure for every $x \in M$,
- a *sub-probability kernel* if κ_x is a sub-probability measure for every $x \in M$,
- a *finite kernel* if there exists $r < \infty$ such that $\kappa_x(N) < r$ for all $x \in M$, and
- an *s-finite kernel* if there exists a countable family of finite kernels $\{\kappa(n)\}_{n\in\mathbb{N}}$ from M to N such that $\kappa_x = \sum_{n=0}^{\infty} \kappa(n)_x$ for all $x \in M$.

Notice that the bound of a finite kernel and the choice of sequence of an s-finite kernel are uniform across all arguments. The notions of probability kernel and sub-probability kernel are expressed as measurable functions using the Giry monad.

Lemma 5.4. Let M and N be measurable spaces and $\kappa : M \times \Sigma_N \to \overline{\mathbb{R}}_{\geq 0}$. Then,

- κ is a probability kernel if and only if $(\lambda x\, A.\, \kappa_x(A))$ is a measurable function from M to $G(N)$.
- κ is a sub-probability kernel if and only if $(\lambda x\, A.\, \kappa_x(A))$ is a measurable function from M to $G_{\mathrm{sub}}(N)$.

We define the binary operator receiving a measure and a kernel.

Definition 5.5. Let M and N be measurable spaces, μ a measure on M and κ a measure kernel from M to N. We define a binary operator $\gg\!=_{\mathrm{k}}$. We write $\gg\!=_{\mathrm{k}}$ in infix notation and $\mu \gg\!=_{\mathrm{k}} \kappa$ is the measure on N defined by

$$(\mu \gg\!=_{\mathrm{k}} \kappa)\,(A) = \int \kappa_x(A)\mu(\mathrm{d}x).$$

The operator $\gg\!=_{\mathrm{k}}$ is an extension of the bind operator of the Giry monad. The Dirac δ forms a unit of $\gg\!=_{\mathrm{k}}$.

Lemma 5.6. Let M and N be measurable space. Then, the following holds.

- $\delta : M \rightsquigarrow M$ is a probability kernel.
- If μ is a measure on M, then $\mu \gg\!=_{\mathrm{k}} \delta = \mu$.
- If $x \in M$ and $\kappa : M \rightsquigarrow N$ is a measure kernel, then $\delta_x \gg\!=_{\mathrm{k}} \kappa = \kappa_x$.

The operator $\gg\!=_{\mathrm{k}}$ has compositionality, associativity, and commutativity for s-finite measures and s-finite kernels. These properties are important for constructing the s-finite measure monad on quasi-Borel spaces.

Lemma 5.7. Let M, N and L be measurable spaces.

- (Compositionality) If $\kappa : M \rightsquigarrow N$ and $\kappa' : M \times N \rightsquigarrow L$ are s-finite kernels, then $(\lambda(x, A).\ (\kappa_x \gg\!=_{\mathrm{k}} (\lambda y.\ \kappa'_{(x,y)}))(A))$ is an s-finite kernel from M to L.
- (Associativity) If μ is a measure on M, $\kappa : M \rightsquigarrow N$ and $\kappa' : N \rightsquigarrow L$ are s-finite kernels, then $\mu \gg\!=_{\mathrm{k}} (\lambda x.\ \kappa_x \gg\!=_{\mathrm{k}} \kappa') = (\mu \gg\!=_{\mathrm{k}} \kappa) \gg\!=_{\mathrm{k}} \kappa'$.
- (Commutativity) If μ is an s-finite measure on M, ν is an s-finite measure on N, and $\kappa : M \times N \rightsquigarrow L$ is a s-finite kernel, then $\mu \gg\!=_{\mathrm{k}} (\lambda x. \nu \gg\!=_{\mathrm{k}} (\lambda y. \kappa_{(x,y)})) = \nu \gg\!=_{\mathrm{k}} (\lambda y. \mu \gg\!=_{\mathrm{k}} (\lambda x.\ \kappa_{(x,y)}))$.

s-Finite Measures and s-Finite Kernels in Isabelle/HOL

We define s-finite measures with the **locale** command same as other classes of measures.

```
locale s-finite-measure =
  fixes M :: 'a measure
  assumes ∃ Mi :: nat ⇒ 'a measure.
                (∀ i. sets (Mi i) = sets M) ∧ (∀ i. finite-measure (Mi i))
                ∧ (∀ A∈sets M. M A = (∑ i. Mi i A))
```

As we mentioned, all σ-finite measures are s-finite measures.

```
sublocale sigma-finite-measure ⊆ s-finite-measure
```

In measure theory, the product measure is usually defined as the unique measure satisfying $(M \bigotimes_M N)\ (A \times B) = M\ A * N\ B$, while Isabelle/HOL's library defines the product measure as $(M \bigotimes_M N)\ A = (\int x.\ (\int y.\ \mathit{indicator}\ A\ (x,y)\ \partial N)\ \partial M)$. Using Isabelle/HOL's definition, we can prove the Fubini-Tonelli theorem in almost similar ways as the proofs for σ-finite measures. For instance, we have the following theorems.

```
lemma nn-integral-fst:
  assumes s-finite-measure M2
  assumes f ∈ M1 ⨂M M2 →M ℝ≥0
  shows (∫+ x. (∫+ y. f (x, y) ∂M2) ∂M1) = (∫+ z. f z ∂(M1 ⨂M M2)
```

```
lemma nn-integral-snd:
  assumes s-finite-measure M1 and s-finite-measure M2
     and f ∈ M1 ⨂M M2 →M ℝ≥0
   shows (∫+ y. (∫+ x. f (x, y) ∂M1) ∂M2) = (∫+ z. f z ∂(M1 ⨂M M2)
```

Next, we define measure kernels.

```
locale measure-kernel =
  fixes M :: 'a measure
   and N :: 'b measure
   and κ :: 'a ⇒ 'b measure
  assumes ⋀x. x ∈ space M ⟹ sets (κ x) = sets N
      and ⋀B. B ∈ sets N⟹(λx. κ x B) ∈ M →M ℝ≥0
      and space M ≠ ∅ ⟹ space N ≠ ∅
```

The third assumption *space* $M \neq \emptyset \Longrightarrow$ *space* $N \neq \emptyset$ in *measure-kernel* is required in order to define the operator $\gg=_k$ in a convenient way, later. We formalize finite kernels, sub-probability kernels, probability kernels, and s-finite kernels as sublocales of measure kernels. Sub-probability kernels and probability kernels are expressed as measurable functions using the Giry monad (Lemma 5.4).

```
locale finite-kernel = measure-kernel +
  assumes ∃ r<∞. ∀ x∈ space M. κ x (space N) < r

locale subprob-kernel = measure-kernel +
  assumes ⋀x. x ∈ space M ⟹ subprob-space (κ x)

locale prob-kernel = measure-kernel +
  assumes ⋀x. x ∈ space M ⟹ prob-space (κ x)

locale s-finite-kernel = measure-kernel +
  assumes ∃ ki. (∀ i. finite-kernel M N (ki i) ∧
                (∀ x∈space M. ∀ A∈sets N. κ x A = (∑ i. ki i x A)))

lemma subprob-kernel M N κ ⟷ κ ∈ M →M subprob-algebra N
lemma prob-kernel M N κ ⟷ κ ∈ M →M prob-algebra N
```

We define the operator $M \gg=_k \kappa$.

```
definition bind-kernel :: 'a measure ⇒ ('a ⇒ 'b measure) ⇒ 'b measure (infixl ≫=k 54) where
bind-kernel M κ =
   (if space M = {} then count-space {}
    else
     let Y = κ (SOME x. x ∈ space M) in
     measure-of (space Y) (sets Y) (λB. ∫+x. (κ x B) ∂M))
```

The measure $M \gg=_k \kappa$ satisfies the following properties for a measure M and *measure-kernel* $M\ N\ \kappa$ when M is not an empty space.

$$sets\ (M \gg=_k \kappa) = sets\ N, \quad (M \gg=_k \kappa)\ B = (\int x.\ (\kappa\ x\ B)\ \partial M)$$

If M is the measure on an empty space, we cannot obtain the measurable structure of N from M and κ (recall the definition of *measure-kernel*). Hence, $M \gg=_k \kappa$ is set to return the discrete empty space as a *default value*. Due to this definition, we need the assumption *space* $M \neq \emptyset \Longrightarrow$ *space* $N \neq \emptyset$ in *measure-kernel*. Without this assumption, we will get stuck to prove compositionality of s-finite kernels later.

The *bind* operator, which has been already defined in the Isabelle/HOL's library, satisfies the same equations as the above equation for $\gg=_k$ when κ is a sub-probability kernel.

```
lemma bind-kernel-bind:
  assumes κ ∈ M →M subprob-algebra N
  shows M ≫=k κ = M ≫= κ
```

Unfortunately, *bind* is defined through the *join* operator of the Giry monad, and thus we do not have the above equations for general measure kernels. Hence, we need to introduce the operator $\gg=_k$ and prove lemmas similar to ones of *bind*.

5.2 Quasi-Borel Spaces

In this section, we formalize basic structures of quasi-Borel spaces.

5.2.1 Quasi-Borel Spaces

In the standard probability theory, we consider a measurable space (Ω, Σ_Ω), where Ω is called a sample space and Σ_Ω is a set of random events. We observe random events through a measurable function called a *random variable*. Thus, we first axiomatize measurable spaces and then the notion of random variables comes later. In contrast, in the theory of quasi-Borel spaces, we first axiomatize random variables where the sample space is restricted to $\mathbb{R}$.

Definition 5.8 (Quasi-Borel Spaces)**.** A *quasi-Borel space* is a pair of a set X and a set $M_X \subseteq \mathbb{R} \to X$ satisfying the following.

- If $\alpha \in M_X$ and $f : \mathbb{R} \to \mathbb{R}$ is measurable, then $\alpha \circ f \in M_X$.
- If α is a constant map, then $\alpha \in M_X$.
- If $\{\alpha_i\}_{i\in\mathbb{N}} \subseteq M_X$ and $P : \mathbb{R} \to \mathbb{N}$ is measurable, then $(\lambda r.\alpha_{P(r)}(r)) \in M_X$.

Intuitively, M_X is the set of random variables over the sample space $\mathbb{R}$. We sometimes write X for a quasi-Borel space (X, M_X) if the structure is obvious from the context. As an example, $\mathbb{R}$ is the quasi-Borel space $(\mathbb{R}, M_\mathbb{R})$ where $M_\mathbb{R}$ is the set of measurable functions from $\mathbb{R}$ to $\mathbb{R}$.

As an analogy of measurable functions, we define the structure-preserving functions between quasi-Borel spaces.

Definition 5.9. A function $f : X \to Y$ is called a *morphism* from (X, M_X) to (Y, M_Y) if $f \circ \alpha \in M_Y$ for all $\alpha \in M_X$.

Quasi-Borel spaces and morphisms between them form the category **QBS**. It has products, countable coproducts, and function spaces, where the function space $X \Rightarrow_Q Y$ is the set $\mathbf{QBS}(X, Y)$ of morphisms from X to Y (thus it is Cartesian closed).

Lemma 5.10. Products, coproducts, and functions spaces of quasi-Borel spaces have the following structures.

$$M_{\prod_{i\in I} X_i} = \{\alpha : \mathbb{R} \to \textstyle\prod_{i\in I} X_i \mid \forall i \in I.\, \pi_i \circ \alpha \in M_i\},$$
$$M_{\coprod_{i\in I} X_i} = \{\lambda r.(f(r), \alpha_{f(r)}(r)) \mid f : \mathbb{R} \to I \text{ is measurable}, \forall i \in \mathrm{image}(f).\, \alpha_i \in M_{X_i}\},$$
$$M_{X\Rightarrow_Q Y} = \{\alpha : \mathbb{R} \to Y^X \mid \mathrm{uncurry}(\alpha) \in \mathbb{R} \times X \Rightarrow_Q Y)\}.$$

Remark 5.11. Note that the evaluation function $\mathrm{ev}\colon \mathbb{R}^\mathbb{R} \times \mathbb{R} \to \mathbb{R}$, $\mathrm{ev}(f, x) = f(x)$ is a morphism. It is easily shown by unfolding the definitions because the product spaces and function spaces are constructed in a simple way.[1]

[1]On the other hand, ev cannot be measurable for an arbitrary σ-algebra on $\mathbb{R}^\mathbb{R}$ [3]. The difficulty comes from the fact that the structure of the product measurable spaces is not simple, unlike quasi-Borel spaces. The structure of product spaces is $\Sigma_{X\times Y} = \sigma[\{A \times B \mid A \in \Sigma_X \wedge B \in \Sigma_Y\}]$, not $\{A \times B \mid A \in \Sigma_X \wedge B \in \Sigma_Y\}$. The operators σ generate a σ-algebra by taking complements and countable unions, and that makes it impossible to construct a σ-algebra on $\mathbb{R}^\mathbb{R}$ making ev measurable.

Quasi-Borel Spaces in Isabelle/HOL

We first introduce a predicate that ensures that a given pair forms a quasi-Borel space.

```
definition qbs-closed1 :: (real ⇒ 'a) set ⇒ bool
 where qbs-closed1 Mx ≡ (∀ a ∈ Mx. ∀ f ∈ real-borel →M real-borel. a ∘ f ∈ Mx)

definition qbs-closed2 :: ['a set, (real ⇒ 'a) set] ⇒ bool
 where qbs-closed2 X Mx ≡ (∀ x ∈ X. (λr. x) ∈ Mx)

definition qbs-closed3 :: (real ⇒ 'a) set ⇒ bool
 where qbs-closed3 Mx ≡ (∀ P::real ⇒ nat. ∀ Fi::nat ⇒ real ⇒ 'a.
        (P ∈ ℝ →M ℕ) ⟶ (∀ i. Fi i ∈ Mx) ⟶ (λr. Fi (P r) r) ∈ Mx)

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

Then, we define the type of quasi-Borel spaces with the **typedef** command.

```
typedef 'a quasi-borel = {(X::'a set, Mx ). is-quasi-borel X Mx}
```

We extract components of quasi-Borel spaces by the following projections.

```
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)
```

We sometimes write X instead of *qbs-space* X using coercion.

In mathematical definition, we first define morphisms of quasi-Borel spaces, then construct function spaces. In our implementation, we first construct function spaces, then define the set of morphisms as an abbreviation.

```
definition exp-qbs :: ['a quasi-borel, 'b quasi-borel] ⇒ ('a ⇒ 'b) quasi-borel (infixr ⇒Q 61) where
X ⇒Q Y ≡ Abs-quasi-borel
                ({f. ∀ α ∈ qbs-Mx X. f ∘ α ∈ qbs-Mx Y},
                 {g. ∀ α∈ ℝ →M ℝ. ∀ β∈ qbs-Mx X. (λr. g (α r) (β r)) ∈ qbs-Mx Y})

abbreviation qbs-morphism :: ['a quasi-borel, 'b quasi-borel] ⇒ ('a ⇒ 'b) set (infixr →Q 60)
 where X →Q Y ≡ qbs-space (X ⇒Q Y)
```

Since we construct function spaces before defining morphisms, the structures of function spaces are written with the definition of morphisms unfolded. We chose this style of definition for the proof automation presented in Section 5.4.

Besides function spaces, our formalization includes binary products, binary coproducts, products, and countable coproducts. We denote a binary product space by $X \bigotimes_Q Y$, a binary coproduct space by $X \bigoplus_Q Y$, a product space by $\Pi_Q\ i{\in}I.\ X\ i$, and a coproduct space by $\amalg_Q\ i{\in}I.\ X\ i$, respectively. For a product space and a coproduct space , every $X\ i$ has to be a quasi-Borel space over the same type due to Isabelle's type system.

5.2.2 List Spaces

List is one of the most important data structure in programming languages. We define the space of lists to use lists in probabilistic programs. The space of lists is constructed using the following isomorphism.

$$List(X) \cong \coprod_{n=0}^{\infty} \prod_{0 \le k < n} X = \{(n, (x_1, \ldots, x_n)) \mid n \in \mathbb{N},\ x_1, \ldots, x_n \in X\}.$$

A pair $(n, (x_1, \ldots, x_n))$ corresponds to $[x_1, \ldots, x_n]$. We also prove that standard constants on lists are morphisms. For instance, the primitive recursive operator *rec_list* is the following morphism.

$$rec\_list \in Y \Rightarrow_Q (X \Rightarrow_Q List[X] \Rightarrow_Q Y \Rightarrow_Q Y) \Rightarrow_Q List[X] \Rightarrow_Q Y. \tag{5.1}$$

In the proof, we first use the product and coproduct style definition where we can use properties of coproduct and product. Then, we convert the statement to the list style one using the isomorphic map.

(5.1)
$\Longleftrightarrow (\lambda l\ y\ f.\ rec\_list\ y\ f\ l) \in List[X] \Rightarrow_Q Y \Rightarrow_Q (X \Rightarrow_Q List[X] \Rightarrow_Q Y \Rightarrow_Q Y) \Rightarrow_Q Y$
$\Longleftrightarrow \forall\, n \in \mathbb{N}, \big(\lambda l'\ y\ f.\ rec\_list\ y\ f\ (n, l')\big)$
$\quad \in \prod_{0 \le k < n} X \Rightarrow_Q Y \Rightarrow_Q (X \Rightarrow_Q List[X] \Rightarrow_Q Y \Rightarrow_Q Y) \Rightarrow_Q Y$

The last equivalence is derived from the universal property of coproducts of quasi-Borel spaces[2].

List Spaces in Isabelle/HOL

In Isabelle/HOL, we define the space of lists as follows.

definition *list-qbs* :: *'a quasi-borel* ⇒ *'a list quasi-borel* **where**
list-qbs X ≡ *map-qbs to-list* (Π$_Q$ *n*∈(*UNIV*::*nat set*). Π$_Q$ *i*∈{..<*n*}. *X*)

The constant *map-qbs*::(*'a* ⇒ *'b*) ⇒ *'a quasi-borel* ⇒ *'b quasi-borel* generates the following quasi-Borel space.

$$qbs\text{-}space\ (map\text{-}qbs\ f\ X) = f\ `\ (qbs\text{-}space\ X),$$
$$qbs\text{-}Mx\ (map\text{-}qbs\ f\ X) = \{f \circ \alpha \mid \alpha.\ \alpha \in qbs\text{-}Mx\ X\}.$$

The function *to-list* maps $(n, (x_1, \ldots, x_n))$ to $[x_1, \ldots, x_n]$. Note that every Π$_Q$ *i*∈{..<*n*}. *X* has quasi-Borel spaces over the same type (*nat* ⇒ *'a*) *quasi-borel*. The following lemmas are used to prove that standard list operators are morphisms from the coproduct space's one.

lemma *map-qbs-morphism-f*: *f* ∈ *X* →$_Q$ *map-qbs f X*

lemma *map-qbs-morphism-inverse-f*:
assumes ⋀*x*. *x* ∈ *qbs-space X* ⟹ *g* (*f x*) = *x*
shows *g* ∈ *map-qbs f X* →$_Q$ *X*

[2]Let I be a countable set and X_i $(i \in I)$, Y quasi-Borel spaces, then $f : \coprod_{i \in I} X_i \to Y$ is a morphism if and only if $(\lambda x.\ f(i, x)) : X_i \to Y$ is a morphism for every $i \in I$.

5.3 Connection between Measurable Spaces and Quasi-Borel Spaces

There are conversions (called *functors* in category theory) between measurable spaces and quasi-Borel spaces. Using the conversions, we can easily derive from theorems in measure theory that basic functions, such as $+$ and $-$, are morphisms. The conversions $L : \mathbf{QBS} \to \mathbf{Meas}$ and $R : \mathbf{Meas} \to \mathbf{QBS}$ return the following structures.

$$\Sigma_{L(X)} = \{A \mid \forall \alpha \in M_X.\ \alpha^{-1}(U) \in \Sigma_{\mathbb{R}}\}, \quad M_{R(N)} = \mathbf{Meas}(\mathbb{R}, N).$$

The conversions are functors because we have $\mathbf{QBS}(X, Y) \subseteq \mathbf{Meas}(L(X), L(Y))$ and $\mathbf{Meas}(M, N) \subseteq \mathbf{QBS}(R(M), R(N))$. We use a measurable space M as a quasi-Borel space $R(M)$.

The functors L and R have the following properties.

Lemma 5.12.
1. $\mathbf{Meas}(L(X), M) = \mathbf{QBS}(X, R(M))$. Thus, (L, R) forms an adjunction between **Meas** and **QBS**.
2. $L(R(M)) = M$ if M is a standard Borel space.
3. R preserves products and countable coproducts.

The connection in Isabelle/HOL

We define the functors L and R with the following structures.

$$\begin{aligned} L &:: 'a\ \textit{quasi-borel} \Rightarrow 'a\ \textit{measure} \\ \textit{space}\ (L\ X) &= \textit{qbs-space}\ X \\ \textit{sets}\ (L\ X) &= \{U \cap \textit{qbs-space}\ X \mid U. \forall \alpha \in \textit{qbs-Mx}\ X.\ \alpha -` U \in \textit{sets}\ \mathbb{R}\} \\ R &:: 'a\ \textit{measure} \Rightarrow 'a\ \textit{quasi-borel} \\ \textit{qbs-space}\ (R\ M) &= \textit{space}\ M \\ \textit{qbs-Mx}\ (R\ M) &= \mathbb{R} \to_M M \end{aligned}$$

We define abbreviations which denote Borel spaces and discrete spaces as quasi-Borel spaces.

```
abbreviation qbs-borel (borel_Q) where borel_Q ≡ R borel
abbreviation qbs-count-space (count'-space_Q) where qbs-count-space I ≡ R (count-space I)
```

The following lemma is used to prove that basic operators are morphisms.

```
lemma
  assumes uncurry f ∈ M ⨂_M N →_M L
  shows f ∈ R M ⇒_Q R N ⇒_Q R L
```

For instance, we obtain $(+) \in \mathbb{R} \Rightarrow_Q \mathbb{R} \Rightarrow_Q \mathbb{R}$ from $(\lambda(x,y).\ x + y) \in \mathbb{R} \bigotimes_M \mathbb{R} \to_M \mathbb{R}$.

5.4 Proof Automation

The Isabelle/HOL's measure theory library provides the automated *measurability prover*. In the context of measure theory, one often needs to show measurability: $A \in \mathit{sets}\ M$ or $f \in M \to_M N$. In pen-and-paper mathematics, measurability proofs are often omitted since they are trivial, while one needs to show measurability each time in the formal proof. The measurability prover automates such proofs of measurability and greatly reduces the cost of proofs. Similar to measure theory, we often need to prove that some function is a morphism, $f \in X \to_Q Y$, in the context of quasi-Borel theory. We have implemented an automated *qbs prover*. Unlike measurable spaces, quasi-Borel spaces have function spaces, hence our qbs prover is similar to type checking of a simply-typed functional programming language.

We construct the qbs prover which tries to prove $x \in \mathit{qbs\text{-}space}\ X$ automatically. The qbs prover can also be used to solve morphism statements $f \in X \to_Q Y$ and $\alpha \in \mathit{qbs\text{-}Mx}\ X$ because we have $X \to_Q Y = \mathit{qbs\text{-}space}\ (X \Rightarrow_Q Y)$ and $\mathit{qbs\text{-}Mx}\ X = \mathbb{R} \to_Q X$.

We regard $(\lambda x.\ e) \in X \Rightarrow_Q Y$ as the typing judgment $x : X \vdash e : Y$, and $e \in \mathit{qbs\text{-}space}\ X$ as $\vdash e : X$. Then, solving $x \in \mathit{qbs\text{-}space}\ X$ is equivalent to solving the corresponding typing judgment. The qbs prover tries to solve typing judgments with the following method:

Algorithm We prepare two sets of introduction rules: Rule_1 and Rule_2. Then, repeat the following steps.

- Try to apply a rule in Rule_1.
- If none of the rules in Rule_1 is applied, then try to apply a rule in Rule_2.

Rule_1 and Rule_2 consist of (at least) the following inference rules.

- Rule_1

$$\frac{}{x : X \vdash x : X}\ \mathrm{ID} \qquad \frac{\vdash e : Y}{x : X \vdash e : Y}\ \mathrm{CONST}\ (x \text{ does not occur free in } e)$$

  After $e \in \mathit{qbs\text{-}space}\ X$ is proved, it may be added as an axiom of Rule_1.

$$\frac{}{\vdash e : X}\ \mathrm{AXIOMS}$$

- Rule_2

$$\frac{\vdash f : X \Rightarrow_Q Y \qquad \vdash x : X}{\vdash f\ x : Y}\ \mathrm{APP}_1 \qquad \frac{x : X \vdash e_1 : Y \Rightarrow_Q Z \qquad x : X \vdash e_2 : Y}{x : X \vdash e_1\ e_2 : Z}\ \mathrm{APP}_2$$

$$\frac{z : X \bigotimes_Q Y \vdash f[\mathit{fst}\ z/x,\ \mathit{snd}\ z/y] : Z}{x : X \vdash (\lambda y.\ f) : Y \Rightarrow_Q Z}\ \mathrm{CURRY}$$

For CURRY, we need to have $\mathit{fst} \in X \bigotimes_Q Y \Rightarrow_Q X$ and $\mathit{snd} \in X \bigotimes_Q Y \Rightarrow_Q Y$ as axioms of Rule_1. There are mainly two reasons why we divide the rules. First, the rule CONST might overlap with APP_2 or CURRY. Because the rule CONST should be applied first, we add CONST to Rule_1. The other reason is to prevent terms from being split in certain

situations. We sometimes add rules for composition of terms, for example *emeasure* M $A \in \overline{\mathbb{R}}_{\geq 0}$, to Rule_1. If we apply a rule in Rule_2 first, then the composed term will be split by the rule APP_1 or APP_2, that is not what we want the prover to do.

The following code is an example usage of the qbs prover.

```
lemma
  assumes [qbs]: f ∈ ℝ ⇒Q ℝ
  shows (λx. 1 + f x) ∈ ℝ ⇒Q ℝ
  by qbs
```

In the above code, we add $f \in \mathbb{R} \Rightarrow_Q \mathbb{R}$ to the axioms of Rule_1 using the attribute $[qbs]$. Rule_1 is configured by our library so that the axioms contain $r \in \mathbb{R}$ and $+$. Then, we call the qbs prover by the tactic *qbs*, which immediately solves the goal.

However, it cannot handle assumptions on typing of lambda abstraction well. It fails for the following example.

```
lemma
  assumes [qbs]: (λx. f x c) ∈ X ⇒Q Y
  shows (λx z. f x c) ∈ X ⇒Q Z ⇒Q Y
```

Implementation Note We have implemented the qbs prover using raw ML code. There are some points to be noted.

- The following theorem corresponds to the rule APP_2 in Isabelle/HOL.

  ```
  lemma
    assumes f ∈ X ⇒Q Y ⇒Q Z and g ∈ X ⇒Q Y
    shows (λx. f x (g x)) ∈ X ⇒Q Z
  ```

  When applying the rule APP_2, we need to instantiate f and g in the lemma so that higher-order unification achieves an intended unification.

- When applying the rule CURRY, we should check by pattern matching that the goal is a lambda abstraction. Otherwise, it may overlap with APP_2 by eta-expanding e_1 e_2 when the term has a function type.

We expect that this *typing* algorithm works in a similar situation where we want to restrict function spaces and constants in Isabelle/HOL. In our situation, function spaces are restricted to the set of morphisms.

5.5 The s-Finite Measure Monad

The s-finite measure monad on **QBS** is an extension of the probability monad. The s-finite measure monad is required to denote semantics of probabilistic programs with soft conditioning, where we need to treat measures possibly infinite.

The s-finite measure monad on quasi-Borel spaces was introduced by Ścibior et al. [44] as the σ-finite measure monad. Then, it was reformulated as a submonad of the continuation monad $[0,\infty]^{[0,\infty]^{(-)}}$ by Vákár et al. [52]. The details of the definition vary among these

previous studies[3], and we could not find detailed proofs of monad laws and commutativity in any of them. We thus recover the detailed proofs first, and then we formalize them. We choose the definition given in Yang's lecture slide [56], because it is suitable for formalization in Isabelle/HOL. Its definition is quite similar to the probability monad introduced by Heunen et al. [25]. The probability monad is derived from the monad laws and the commutativity of the Giry monad, while the s-finite measure monad is derived from the properties of s-finite kernels and $\gg\!=_{\mathrm{k}}$.

5.5.1 Measures on Quasi-Borel Spaces

First, we define measures on quasi-Borel spaces to treat infinite measures such as the Lebesgue measure. Intuitively, a measure is a pair consisting of an s-finite measure μ on $\mathbb{R}$ and a random variable $\alpha \in M_X$. We also introduce the equivalence relation $\sim$ of measures on quasi-Borel spaces defined by relating pairs with equal push-forward measures.

Definition 5.13 (Measures on Quasi-Borel Spaces)**.** A *measure* on quasi-Borel space X is an equivalence class $[\alpha, \mu]_{\sim_X}$ where $\alpha \in M_X$ and μ is an s-finite measure on $\mathbb{R}$. The equivalence relation is defined by[4] $(\alpha, \mu) \sim_X (\beta, \nu) \iff \alpha_*\mu = \beta_*\nu$.

We call a measure on a quasi-Borel space a *qbs-measure* in order to distinguish it from measures in measure theory.

Any qbs-measures is converted to s-finite measures by $l_X([\alpha, \mu]_{\sim_X}) = \alpha_*\mu$. The function l_X is injective by the definition of qbs-measures. Furthermore, l_X is bijective if X is a standard Borel space. The function l_X is a measure kernel from $R(\mathscr{M}(X))$ to $R(X)$ and $l_X(p)$ is an s-finite measure on $R(X)$ for every qbs-measure p on X.

5.5.2 Integral

Integral with qbs-measure is defined through the Lebesgue integral. Let $f : X \to \mathbb{R}$ be a morphism and p a qbs-measure on X. Then, the integral of f with respect to p is defined as[5] $\int f \mathrm{d}p \stackrel{\mathrm{def}}{=} \int f \mathrm{d}l_X(p)$. The notions of integrability and almost everywhere are also defined in the same way.

Integral with respect to an s-finite measure on a standard Borel space is represented as an integral in quasi-Borel theory. Let μ be an s-finite measure on a standard Borel space N. Then, we have a pair of measurable functions $N \xrightarrow{f} \mathbb{R} \xrightarrow{g} N$ such that $g \circ f = \mathrm{id}_N$ from Kuratowski's theorem. Since the push-forward measure $g_*\mu$ is s-finite and $f \in M_N$, $[f, g_*\mu]_{\sim_N}$ is a qbs-measure on N. Hence, we have $\int h \mathrm{d}[f, g_*\mu]_{\sim_N} = \int h \mathrm{d}l_N([f, g_*\mu]_{\sim_N}) = \int h \mathrm{d}(f_*g_*\mu) = \int h \mathrm{d}\mu$ for measurable functions $h : M \to \mathbb{R}$.

5.5.3 Density Measure

In measure theory, the density measure $\mu_{\int f}$ is defined by $\mu_{\int f}(A) = \int_A f \mathrm{d}\mu$, where μ is a measure on a measurable space M, $f : M \to \overline{\mathbb{R}}_{\geq 0}$ a measurable function, and $A \in \Sigma_M$.

[3] Thanks to Kuratowski's theorem and the fact that s-finite measures can be rewritten as push-forward of σ-finite measures, those definitions are essentially equivalent.

[4] The measure $\alpha_*\mu$ is a measure on $L(X)$ because $\alpha \in M_X = \mathbf{QBS}(\mathbb{R}, X) \subseteq \mathbf{Meas}(\mathbb{R}, L(X))$.

[5] Since $f \in \mathbf{QBS}(X, \mathbb{R}) \subseteq \mathbf{Meas}(L(X), \mathbb{R})$, f is measurable.

Similarly, we define the density qbs-measure. Let $p = [\alpha, \mu]_{\sim_X}$ be a qbs-measure on X and $f : X \to \overline{\mathbb{R}}_{\geq 0}$ a morphism. The density qbs-measure $p_{\int f}$ is defined by $p_{\int f} = [\alpha, \mu_{\int (f \circ \alpha)}]_{\sim_X}$. The density qbs-measure is well-defined because $\alpha_* \mu_{\int (f \circ \alpha)} = (\alpha_* \mu)_{\int f}$. As density measures, we have $\int g \mathrm{d}p_{\int f} = \int f g \mathrm{d}p$ for every morphism $g : X \to \mathbb{R}$.

The *normalizer* of qbs-measure is defined by $normalize(p) = p_{\int (1/l_X(p)(X))}$.

5.5.4 The s-Finite Measure Monad

Next, we construct the s-finite measure monad.

Lemma 5.14. The quasi-Borel spaces of qbs-measures on X has the following structure.

$$\begin{aligned}\mathscr{M}(X) &= \{s.\ s \text{ is a qbs-measure on } X\},\\ M_{\mathscr{M}(X)} &= \{(\lambda r.\ [\alpha, \kappa(r)]_{\sim_X}) \mid \alpha \in M_X \wedge \kappa : \mathbb{R} \rightsquigarrow \mathbb{R} \text{ is an s-finite kernel}\}.\end{aligned}$$

Notice that we use the s-finite kernel in the definition of $M_{\mathscr{M}(X)}$. We reconstruct the proof because there is no literature including the proof.

Proof. We show the third condition in Definition 5.8 as the other conditions are easy to check. Let $\{\beta_i\}_{i \in \mathbb{N}} \subseteq M_{\mathscr{M}(X)}$ and $P : \mathbb{R} \to \mathbb{N}$ be measurable. Then, there exists $\{\alpha_i\}_{i \in \mathbb{N}} \subseteq M_X$ and s-finite kernels $\{\kappa_i : \mathbb{R} \rightsquigarrow \mathbb{R}\}_{i \in \mathbb{N}}$ such that $\beta_i = \lambda r.\ [\alpha_i, \kappa_i(r)]_{\sim_X}$. Our goal is to show $\lambda r.\ [\alpha_{P(r)}, \kappa_{P(r)}(r)]_{\sim_X} \in M_{\mathscr{M}(X)}$.

Since $\mathbb{N} \times \mathbb{R}$ is a standard Borel space, we have measurable functions $\mathbb{N} \times \mathbb{R} \xrightarrow{a} \mathbb{R} \xrightarrow{b} \mathbb{N} \times \mathbb{R}$ such that $b \circ a = \mathrm{id}_{\mathbb{N} \times \mathbb{R}}$ from Kuratowski's theorem. Let $\alpha' = \lambda(i, r).\ \alpha_i(r)$. Then, we have the following equation for $r \in \mathbb{R}$.

$$\begin{aligned}{\alpha_{P(r)}}_* \kappa_{P(r)}(r) &= (\alpha' \circ (\lambda l.\ (P(r), l)))_* \kappa_{P(r)}(r)\\ &= \alpha'_*((\lambda l.\ (P(r), l))_* \kappa_{P(r)}(r))\\ &= \alpha'_*((b \circ a)_* (\lambda l.\ (P(r), l))_* \kappa_{P(r)}(r))\\ &= (\alpha' \circ b)_* (a_* (\lambda l.\ (P(r), l))_* \kappa_{P(r)}(r))\end{aligned}$$

It is easy to check $\alpha' \circ b \in M_X$ and $\lambda r.\ a_*(\lambda l.\ (P(r), l))_* \kappa_{P(r)}(r) : \mathbb{R} \rightsquigarrow \mathbb{R}$ is an s-finite kernel. Hence, $(\lambda r.\ [\alpha_{P(r)}, g_{P(r)}(r)]_{\sim_X}) = (\lambda r.\ [\alpha' \circ b, a_*(\lambda l.\ (P(r), l))_* \kappa_{P(r)}(r)]_{\sim_X}) \in M_{\mathscr{M}(X)}$. □

Next, we introduce the return (unit) and bind operators.

Definition 5.15. The return (unit) operator is defined by $\eta_X(x) = [\lambda r.\ x, \nu]_{\sim_X}$ where ν is an arbitrary probability measure on $\mathbb{R}$.

The operator η is well defined because $(\lambda r.\ x)_* \nu = \delta_x$ for any probability measures ν on $\mathbb{R}$.

Definition 5.16. The bind operator is defined by $[\alpha, \mu]_{\sim_X} \gg\!= f = [\beta, \mu \gg\!=_{\mathrm{k}} \kappa]_{\sim_Y}$ where $\alpha \in M_X$, $f : X \to \mathscr{M}(Y)$ is a morphism, $\beta \in M_Y$ and $\kappa : \mathbb{R} \rightsquigarrow \mathbb{R}$ is an s-finite kernel such that $f \circ \alpha = \lambda r.\ [\beta, \kappa(r)]_{\sim_Y}$.

Lemma 5.17. The bind operator is well defined.

Proof. Let $(\alpha, \mu) \sim_X (\alpha', \mu')$, $f : X \to \mathscr{M}(X)$ be an morphism, $f \circ \alpha = (\lambda r.\, [\beta, \kappa(r)]_{\sim_Y})$, and $f \circ \alpha' = (\lambda r.\, [\beta', \kappa'(r)]_{\sim_Y})$. Then,

$$\begin{aligned}
\beta_*(\mu \gg\!=_{\mathrm{k}} \kappa) &= \mu \gg\!=_{\mathrm{k}} (\lambda r.\, \beta_*(\kappa(r))) \\
&= \mu \gg\!=_{\mathrm{k}} (\lambda r.\, l_Y([\beta, \kappa(r)]_{\sim_Y})) \\
&= \mu \gg\!=_{\mathrm{k}} (\lambda r.\, l_Y((f \circ \alpha)(r))) \\
&= \alpha_*\mu \gg\!=_{\mathrm{k}} (\lambda x.\, l_Y(f(x))) \\
&= \alpha'_*\mu' \gg\!=_{\mathrm{k}} (\lambda x.\, l_Y(f(x))) \\
&= \mu' \gg\!=_{\mathrm{k}} \big(\lambda r.\, l_Y((f \circ \alpha')(r))\big) \\
&= \mu' \gg\!=_{\mathrm{k}} \big(\lambda r.\, l_Y([\beta', \kappa'(r)]_{\sim_Y})\big) \\
&= \mu' \gg\!=_{\mathrm{k}} (\lambda r.\, \beta'_*(\kappa'(r))) \\
&= \beta'_*(\mu' \gg\!=_{\mathrm{k}} \kappa').
\end{aligned}$$

□

The injection l preserves return and bind operators.

$$l_X(\eta_X(x)) = \delta_x, \qquad l_Y(p \gg\!= f) = l_X(p) \gg\!=_{\mathrm{k}} l_Y \circ f. \tag{5.2}$$

Theorem 5.18. The triple $(\mathscr{M}, \eta, \gg\!=)$ forms a commutative strong monad on **QBS**.

Proof. Monad laws and commutativity of $\mathscr{M}$ follow from the properties of s-finite kernels (Lemma 5.7), the equations 5.2, and the injectivity of l. For instance, the associativity is shown as follows.

$$\begin{aligned}
[\alpha, \mu]_{\sim_X} \gg\!= f_1 \gg\!= f_2 &= [\gamma, \mu \gg\!=_{\mathrm{k}} \kappa_1 \gg\!=_{\mathrm{k}} \kappa_2]_{\sim_Z} \\
&= [\gamma, \mu \gg\!=_{\mathrm{k}} (\lambda r.\, \kappa_1(r) \gg\!=_{\mathrm{k}} \kappa_2)]_{\sim_Z} \\
&= [\alpha, \mu]_{\sim_X} \gg\!= (\lambda x.\, f_1(x) \gg\!= f_2).
\end{aligned}$$

In the above equation, γ, κ_1, and κ_2 satisfy $f \circ \alpha = (\lambda r.[\beta, \kappa_1]_{\sim_Y})$ and $f_2 \circ \beta = (\lambda r.[\gamma, \kappa_2(r)]_{\sim_Z}$.

The strength $\mathrm{st}_{X,Y} : X \times \mathscr{M}(Y) \to \mathscr{M}(X \times Y)$ is defined by $\mathrm{st}_{X,Y}(x, [\beta, \mu]_{\sim_Y}) = [(\lambda r.\, (x, \beta(r))), \mu]_{\sim_{X \times Y}}$. We show that $\mathrm{st}_{X,Y}$ is a morphism. Other axioms of strength are easily proved.

Since $\mathbb{R} \times \mathbb{R}$ is a standard Borel space, we have measurable functions $\mathbb{R} \times \mathbb{R} \xrightarrow{a} \mathbb{R} \xrightarrow{b} \mathbb{R} \times \mathbb{R}$ such that $b \circ a = \mathrm{id}_{\mathbb{R} \times \mathbb{R}}$ from Kuratowski's theorem. Let $\gamma \in M_{X \times \mathscr{M}(Y)}$, then we have $\alpha \in M_X$, $\beta \in M_Y$, and an s-finite kernel $\kappa : \mathbb{R} \rightsquigarrow \mathbb{R}$ such that $\gamma = (\lambda r.\, (\alpha(r), [\beta, \kappa(r)]_{\sim_Y}))$. We need to show $\mathrm{st}_{X,Y} \circ \gamma = (\lambda r.\, [\lambda l.\, (\alpha(r), \beta(l)), \kappa(r)]_{\sim_{X \times Y}}) \in M_{\mathscr{M}(X \times Y)}$. Let us denote $\lambda(r_1, r_2).\, (\alpha(r_1), \beta(r_2))$ by $\alpha \times \beta$. We have the following equation for $r \in \mathbb{R}$.

$$\begin{aligned}
(\lambda l.\, (\alpha(r), \beta(l)))_*\kappa(r) &= ((\alpha \times \beta) \circ (\lambda l.\, (r, l)))_*\kappa(r) \\
&= (\alpha \times \beta)_*(\lambda l.\, (r, l))_*\kappa(r) \\
&= (\alpha \times \beta)_*(\delta_r \times \kappa(r)) \\
&= (\alpha \times \beta)_*(b \circ a)_*(\delta_r \times \kappa(r)) \\
&= ((\alpha \times \beta) \circ b)_*(a_*(\delta_r \times \kappa(r))).
\end{aligned}$$

It is easy to check that $(\alpha \times \beta) \circ b \in M_{X \times Y}$ and $(\lambda r.\, a_*(\delta_r \times \kappa(r))) : \mathbb{R} \rightsquigarrow \mathbb{R}$ is an s-finite kernel. Thus, $\mathrm{st}_{X,Y} \circ \gamma = (\lambda r.\, [(\alpha \times \beta) \circ b, a_*(\delta_r \times \kappa(r))]_{\sim}) \in M_{\mathscr{M}(X \times Y)}$. □

As a corollary, we obtain that the bind operator is a morphism.

Corollary 5.19. The bind operator is a morphism, that is, $(\lambda p\ f.\ p \gg\!= f) \in \mathscr{M}(X) \Rightarrow_Q (X \Rightarrow_Q \mathscr{M}(Y)) \Rightarrow_Q \mathscr{M}(Y)$.

Proof. We first observe that for a morphism $f : X \to Y$, the functorial action of $\mathscr{M}$ is defined by $\mathscr{M}(f)([\alpha, \mu]_{\sim_X}) = [f \circ \alpha, \mu]_{\sim_Y}$. The join operator $\mathrm{join}_X : \mathscr{M}(\mathscr{M}(X)) \to \mathscr{M}(X)$ is defined by $\mathrm{join}_X(p) = p \gg\!= \mathrm{id}$. It is easy to check that both of $\mathscr{M}(f)$ and join_X are morphisms. Then, we have

$$(\lambda(f, p).\ p \gg\!= f) = \mathrm{join}_X \circ \mathscr{M}(ev) \circ \mathrm{st}_{X \Rightarrow_Q \mathscr{M}(Y), \mathscr{M}(X)} \tag{5.3}$$

where ev is the evaluation function. The right hand side of the equation 5.3 is a composition of morphisms, thus a morphism. Hence, $\gg\!=$ is a morphism. □

5.5.5 The Probability Monad

The probability monad $\mathscr{P}$ on **QBS** introduced by Heunen et.al. [25] is obtained by taking a subspace of $\mathscr{M}$. For a quasi-Borel space X and a set $A \subseteq X$, the subspace has the structure $M_A = \{\alpha \in M_X \mid \forall r.\ \alpha(r) \in A\}$.

Lemma 5.20. Let $\mathscr{P}(X) = \{p \in \mathscr{M}(X).\ l_X(p) \text{ is a probability space}\}$. Then, the subspace $\mathscr{P}(X)$ has the same structure as the probability monad on **QBS**. That is,

$$M_{\mathscr{P}(X)} = \{(\lambda r.\ [\alpha, g(r)]_{\sim_X}) \mid \alpha \in M_X \wedge g \in \mathbf{Meas}(\mathbb{R}, G(\mathbb{R}))\}.$$

The triple $(\mathscr{P}, \eta, \gg\!=,)$ also forms a commutative strong monad on **QBS**.

The measurable function $l_X : L(\mathscr{P}(X)) \to G(L(X))$ forms a *monad opfunctor* from the probability monad $\mathscr{P}$ on **QBS** to the Giry monad G on **Meas** (Proposition 22 [25]) because it satisfies the following equations.

$$l_X(\eta_X(x)) = \delta_x, \qquad l_Y(p \gg\!= f) = l_X(p) \gg\!=_G l_Y \circ f. \tag{5.4}$$

The s-Finite Measure Monad in Isabelle/HOL

We define the type of qbs-measure. We first define the partial equivalence relation $\sim$.

```
type-synonym 'a qbs-s-finite-t = 'a quasi-borel * (real ⇒ 'a) * real measure

definition qbs-s-finite-eq :: ['a qbs-s-finite-t, 'a qbs-s-finite-t] ⇒ bool where
  qbs-s-finite-eq p1 p2 ≡
   (let (X, α, μ) = p1;
        (Y, β, ν) = p2 in
    qbs-s-finite X α μ ∧ qbs-s-finite Y β ν ∧ X = Y ∧
      distr μ (qbs-to-measure X) α = distr ν (qbs-to-measure Y) β)
```

Notice that we define the relation $\sim$ on triple (X, α, μ) rather than (α, μ) because X cannot be inferred from α in a simple type system. In the definition of *qbs-s-finite-eq*, the predicate *qbs-s-finite X α μ* means that $\alpha \in$ *qbs-Mx X* and μ is an s-finite measure on $\mathbb{R}$. The type of qbs-measure is defined using the **quotient-type** command.

```
quotient-type 'a qbs-measure = 'a qbs-s-finite-t / partial: qbs-s-finite-eq
  morphisms rep-qbs-measure qbs-measure
```

We explicitly give names for abstraction/representation functions as *qbs-measure*/*rep-qbs-measure*, respectively with the keyword **morphisms**. We abbreviate *qbs-measure X α μ* as $[\![X, \alpha, \mu]\!]_{\mathsf{sfin}}$. Let us define the function which extracts the quasi-Borel space on which the measure is defined, and l.

```
lift-definition qbs-space-of :: 'a qbs-measure ⇒ 'a quasi-borel
is fst

lift-definition qbs-l :: 'a qbs-measure ⇒ 'a measure
is λp. distr (snd (snd p)) (qbs-to-measure (fst p)) (fst (snd p))
```

We define two kinds of integrals following two integrals in the measure theory library: non-negative integral and Bochner integral.

```
lift-definition qbs-nn-integral :: ['a qbs-measure, 'a ⇒ ennreal] ⇒ ennreal
is λ(X,α,μ) f.(∫⁺x. f x ∂distr μ (qbs-to-measure X) α)

lift-definition qbs-integral :: ['a qbs-measure, 'a ⇒ ('b :: {banach,second-countable-topology})] ⇒
'b
is λp f. if f ∈ fst p →Q qbs-borel then (∫ x. f (fst (snd p) x) ∂ (snd (snd p))) else 0
```

Although we define the integrals using **lift-definition**, they are essentially same as $\int f \mathrm{d}p \stackrel{\mathrm{def}}{=} \int f \mathrm{d}l_X(p)$ which we used in their definition.

```
lemma qbs-nn-integral s f = integral^N (qbs-l s) f
lemma qbs-integral s f = integral^L (qbs-l s) f
```

The density qbs-measure is defined as follows.

```
lift-definition density-qbs :: ['a qbs-measure, 'a ⇒ ennreal] ⇒ 'a qbs-measure
is λ(X,α,μ) f. if f ∈ X →Q ℝ̄≥0
               then (X, α, density μ (f ∘ α))
               else (X, SOME a. a ∈ qbs-Mx X, null-measure borel)
```

The density qbs-measure *density-qbs* returns the null measure when the argument is not a morphism. The normalizer is defined with *density-qbs*.

```
definition normalize-qbs :: 'a qbs-measure ⇒ 'a qbs-measure where
normalize-qbs s ≡ (let X = qbs-space-of s;
                       r = qbs-l s (qbs-space X) in
                   if r ≠ 0 ∧ r ≠ ∞
                   then density-qbs s (λx. 1 / r)
                   else qbs-null-measure X)
```

If the received measure is null-measure or infinite measure, *normalize-qbs* returns the null measure.

The inverse function of *qbs-l* is defined as follows.

```
definition qbs-l-inverse :: 'a measure ⇒ 'a qbs-measure where
qbs-l-inverse M ≡ [![measure-to-qbs M, from-real-into M, distr M ℝ (to-real-on M)]!]sfin
```

For instance, the Lebesgue measure is represented as a qbs-measure.

definition $lborel_Q \equiv$ *qbs-l-inverse lborel*
lemma *qbs-l* $lborel_Q =$ *lborel*
corollary $(\int_Q x.\ f\ x\ \partial lborel_Q) = (\int x.\ f\ x\ \partial lborel)$

Next, we define the s-finite measure monad. The space of qbs-measure, the unit operator, and the bind operator are defined as follows.

definition *monadM-qbs* :: $'a$ *quasi-borel* $\Rightarrow$ $'a$ *qbs-measure quasi-borel* **where**
monadM-qbs $X \equiv$ *Abs-quasi-borel*
$(\{s.\ qbs\text{-}space\text{-}of\ s = X\},$
$\{\lambda r.\ [\![X,\ \alpha,\ k\ r]\!]_{\mathsf{sfin}}\ |\alpha\ k.\ \alpha \in qbs\text{-}Mx\ X \wedge s\text{-}finite\text{-}kernel\ \mathbb{R}\ \mathbb{R}\ k\})$

definition *return-qbs* :: $'a$ *quasi-borel* $\Rightarrow$ $'a \Rightarrow$ $'a$ *qbs-measure* $(return_Q)$ **where**
$return_Q\ X\ x \equiv [\![X,\ \lambda r.\ x,\ SOME\ \mu.\ real\text{-}distribution\ \mu]\!]_{\mathsf{sfin}}$

definition *bind-qbs* :: $['a$ *qbs-measure*, $'a \Rightarrow$ $'b$ *qbs-measure*$] \Rightarrow$ $'b$ *qbs-measure* **where**
bind-qbs $s\ f \equiv$ (*let*
$(X,\ \alpha,\ \mu) =$ *rep-qbs-measure* s;
$Y =$ *qbs-space-of* $(f\ (\alpha\ undefined))$;
$(\beta,\ k) = (SOME\ (\beta,\ k).\ f \circ \alpha = (\lambda r.\ [\![Y,\ \beta,\ k\ r]\!]_{\mathsf{sfin}}) \wedge \beta \in qbs\text{-}Mx\ Y \wedge s\text{-}finite\text{-}kernel\ \mathbb{R}\ \mathbb{R}\ k)$
in $[\![Y,\ \beta,\ \mu \gg\!=_k\ k]\!]_{\mathsf{sfin}}$)

The predicate *real-distribution* μ means that μ is a probability measure on $\mathbb{R}$. We write *bind-qbs* $s\ f$ for $s \gg\!= f$.

The probability monad $\mathscr{P}$ is defined by taking subspace.

definition *monadP-qbs* $X \equiv$ *sub-qbs* $(monadM\text{-}qbs\ X)$ $\{s.\ prob\text{-}space\ (qbs\text{-}l\ s)\}$

In the end of this section, we observe that many constants related to qbs-measures are morphisms.

lemma *qbs-almost-everywhere* $\in$ *monadM-qbs* $X \Rightarrow_Q (X \Rightarrow_Q \mathbb{B}) \Rightarrow_Q \mathbb{B}$

lemma *qbs-nn-integral* $\in$ *monadM-qbs* $X \Rightarrow_Q (X \Rightarrow_Q \overline{\mathbb{R}}_{\geq 0}) \Rightarrow_Q \overline{\mathbb{R}}_{\geq 0}$

lemma *qbs-integral* $\in$ *monadM-qbs* $X \Rightarrow_Q (X \Rightarrow_Q qbs\text{-}borel)$
$\Rightarrow_Q$ $(qbs\text{-}borel :: ('b :: \{second\text{-}countable\text{-}topology, banach\})\ quasi\text{-}borel)$

lemma *density-qbs* $\in$ *monadM-qbs* $X \Rightarrow_Q (X \Rightarrow_Q \overline{\mathbb{R}}_{\geq 0}) \Rightarrow_Q$ *monadM-qbs* X

lemma *normalize-qbs* $\in$ *monadM-qbs* $X \Rightarrow_Q$ *monadM-qbs* X

Chapter 6

Applications

In this chapter, we apply our quasi-Borel space library to program verification of probabilistic programs. We also demonstrate a formalization of differential privacy using quasi-Borel spaces.

Contributions

We verify several examples from previous studies. We first show four examples of probabilistic programs: "two dice", "what time is it?", "Monte Carlo approximation", and "Gaussian mean learning". The first two examples are *toy* problems, while the latter two are more practical. The example of the Monte Carlo approximation states that the average of samples converges in probability to the expected value. This property is a variant of the weak law of large numbers. Gaussian mean learning algorithm infers the unknown mean of a Gaussian distribution with a data sampled from the distribution. We formalize two properties of the algorithm: convergence and stability under change of priors. We also apply quasi-Borel spaces to formalize differential privacy and show a simple example. The definitions and properties of differential privacy using quasi-Borel spaces are easily derived from the ones using measurable spaces.

One of the benefits of using quasi-Borel spaces as the semantics of probabilistic programs is that we can treat higher-order programs. Another benefit is that we can use Isabelle/HOL's terms as probabilistic programs. Most of Isabelle/HOL's constants are defined as curried forms, e.g., the type of $(+)$ is $'a \Rightarrow 'a \Rightarrow 'a$, not $'a \times 'a \Rightarrow 'a$. We can directly use Isabelle/HOL terms as probabilistic programs without uncurrying terms, e.g., $(+) \in \mathbb{R} \Rightarrow_Q \mathbb{R} \Rightarrow_Q \mathbb{R}$, because quasi-Borel spaces have function spaces.

Reference

Examples in Section 6.2.4, and Section 6.2.5, and the semantics of the higher-order probabilistic programs are based on the work by Sato et al. [41]. Other examples are taken from the work by Staton [48, Section 2.2] and Sampson [40, Section 2.3]. We refer to the work by Dwork et al. [16], Barthe and Olmedo [7], and Dwork and Roth [17] for the definition, properties, and example of differential privacy. Our formalization of differential privacy is based on the work by Sato and Minamide [43], where they formalized differential privacy using measure theory.

```
data {
  int N;
  real X[N];
  real sigma;
}

parameters {
  real mu;
}

model {
  for (n in 1:N) {
    X[n] ~ normal(mu,sigma);
  }
}
```

Fig. 6.1: Stan program `ex.stan`.

```
N = 100000
sd = 1
samples <- rnorm(N, mean=1, sd=sd)

data <-
  list(N=N, X=samples, sigma = sd)
fit <- stan(file='ex.stan', data=data)
ms <- rstan::extract(fit)
```

```
> ms$mu[1]
[1] 1.005344
```

Fig. 6.2: Above: R program executing the Stan program. Below: a sample from the posterior distribution

6.1 Probabilistic Programming Languages

Before we formalize probabilistic programs in Isabelle/HOL, let us review what is a probabilistic programming language. Probabilistic programming languages are usually domain-specific languages to describe probabilistic models. Users input a generative probabilistic model and data. Then, the implementation of the language infers unknown parameters of the model automatically. In a typical situation of Bayesian data modeling[1], one repeats the following tasks until obtaining *good* results.

1. Construct a probabilistic model.
2. Implement an inference algorithm according to the model.
3. Estimate parameters using the algorithm.

The step 2 requires more effort to consider and write complex inference algorithms every time they design a probabilistic model. Probabilistic programming languages reduce the cost of modeling by automating the step 2. As a simple example, let us consider the situation that we try to guess the mean of a Gaussian (normal) distribution with a known deviation. The probabilistic model is written as follows.

$$X_n \sim \mathrm{Gauss}(\mu, \sigma) \quad (n = 1, \ldots, N) \tag{6.1}$$

The inference task is done with the probabilistic programming language Stan [13]. Stan has interfaces with popular data analysis programming languages, e.g., R and Python. We run Stan using R in this example. Fig. 6.1 is the Stan program denoting model (6.1) and Fig. 6.2 is an R program that executes the Stan program. In this example, we prepare a dataset sampled from the Gauss distribution with the mean 1 and the standard deviation 1. The function `stan` runs an inference algorithm automatically, then generates a sample sequence obtained from the posterior distribution. As we can observe, probabilistic programming languages

[1] In Baysian data modeling, we assume that parameters are distributed under some distributions. We try to guess the conditional distribution of the parameters after obtained dataset. The posterior distribution is calculated according to the Bayes' rule: posterior $\propto$ prior $\times$ likelihood.

hide complex details of inference algorithms and enable users to concentrate on designing probabilistic models rather than implementing inference algorithms by themselves.

Remark 6.1. We omit details about how the inference tasks are done in this thesis because we focus on the denotational semantics of probabilistic programs rather than the implementation of probabilistic programming languages. Roughly speaking, probabilistic programming languages first compute an unnormalized density function of the posterior distribution according to the model. In a complex model, we can directly specify the density function by writing likelihood functions. Then, probabilistic programming languages run an inference algorithm to obtain a sample sequence from the posterior distribution. Typical inference algorithms are Markov chain Monte Carlo (MCMC) methods. MCMC methods try to construct a Markov chain which quickly reaches its stationary distribution equal to the posterior distribution.

Remark 6.2. The word "probabilistic programming languages" means the programming languages for Bayesian modeling, introduced in this section. The word "probabilistic programs" refers to randomized programs and/or programs written in probabilistic programming languages. We also use the word "probabilistic programs with conditioning" for programs written in probabilistic programming languages.

6.2 Higher-Order Probabilistic Programs

Let us implement a programming language supporting higher-order functions, sampling from distributions, and conditioning with quasi-Borel spaces and the s-finite measure monad. We discuss four examples in this section.

6.2.1 The Language

We use Isabelle/HOL terms as programs. The language design is inspired by `HPProg` introduced by Sato et al. [41]. The language is a higher-order functional programming language based on simply-typed lambda calculus along with the monadic operators for distributions and an operator for conditioning (inference tasks). We first briefly review the type system and semantics of `HPProg`. Types are defined inductively as follows.

$$T ::= \mathsf{nat} \mid \mathsf{bool} \mid \mathsf{real} \mid \mathsf{preal} \mid \mathsf{list}[T] \mid T \times T \mid T \Rightarrow T \mid M[T].$$

The type preal denotes the type of $\overline{\mathbb{R}}_{\geq 0}$ and $M[T]$ denotes the type of distributions (measures) on T. In the semantics, types are interpreted as quasi-Borel spaces.

$$[\![\mathsf{nat}]\!] = \mathbb{N}, \quad [\![\mathsf{bool}]\!] = \mathbb{B}, \quad [\![\mathsf{real}]\!] = \mathbb{R}, \quad [\![\mathsf{preal}]\!] = \overline{\mathbb{R}}_{\geq 0}, \quad [\![\mathsf{list}[T]]\!] = \mathit{list\text{-}qbs}\ [\![T]\!],$$
$$[\![T_1 \times T_2]\!] = [\![T_1]\!] \bigotimes_Q [\![T_2]\!], \quad [\![T_1 \Rightarrow T_2]\!] = [\![T_1]\!] \Rightarrow_Q [\![T_2]\!], \quad [\![M[T]]\!] = \mathit{monadM\text{-}qbs}\ [\![T]\!].$$

A typing judgment $\Gamma \vdash t : T$ is interpreted as "$[\![t]\!]$ is a morphism from $[\![\Gamma]\!]$ to $[\![T]\!]$". A typing judgment $\vdash t : T$ is interpreted as "$[\![t]\!] \in [\![T]\!]$".

According to this semantics, an Isabelle/HOL term is interpreted as a probabilistic program. We say that an Isabelle/HOL term t is a program of type T if $t \in \mathit{qbs\text{-}space}\ T$. Many

standard constants in Isabelle/HOL are programs.

$$(+) \in \mathbb{R} \Rightarrow_Q \mathbb{R} \Rightarrow_Q \mathbb{R}\ , \qquad (-) \in \mathbb{R} \Rightarrow_Q \mathbb{R} \Rightarrow_Q \mathbb{R}\ , \qquad (*) \in \mathbb{R} \Rightarrow_Q \mathbb{R} \Rightarrow_Q \mathbb{R}$$

$$[] \in \textit{list-qbs}\ X\ , \qquad \textit{Cons} \in X \Rightarrow_Q \textit{list-qbs}\ X \Rightarrow_Q \textit{list-qbs}\ X$$

$$\textit{rec-list} \in Y \Rightarrow_Q (X \Rightarrow_Q \textit{list-qbs}\ X \Rightarrow_Q Y \Rightarrow_Q Y) \Rightarrow_Q \textit{list-qbs}\ X \Rightarrow_Q Y$$

Operators for distributions are also programs.

$$\textit{return}_Q\ X \in X \Rightarrow_Q \textit{monadM-qbs}\ X$$

$$(\gg\!=) \in \textit{monadM-qbs}\ X \Rightarrow_Q (X \Rightarrow_Q \textit{monadM-qbs}\ Y) \Rightarrow_Q \textit{monadM-qbs}\ Y$$

$$(\bigotimes_{Qmes}) \in \textit{monadM-qbs}\ X \Rightarrow_Q \textit{monadM-qbs}\ Y \Rightarrow_Q \textit{monadM-qbs}\ (X \bigotimes_Q Y)$$

$$\textit{Uniform} \in \mathbb{R} \Rightarrow_Q \mathbb{R} \Rightarrow_Q \textit{monadM-qbs}\ \mathbb{R}\ , \qquad \textit{Gauss} \in \mathbb{R} \Rightarrow_Q \mathbb{R} \Rightarrow_Q \textit{monadM-qbs}\ \mathbb{R}$$

The program $(\bigotimes_{Qmes})$ is defined for $p \in \textit{monadM-qbs}\ X$ and $q \in \textit{monadM-qbs}\ Y$ by

$$p \bigotimes_{Qmes} q = p \gg\!= (\lambda x.\ q \gg\!= (\lambda y.\ \textit{return}_Q\ (X \bigotimes_Q Y)\ (x,y)))$$

which denotes their product distribution[2]. The program *Uniform a b* denotes the continuous uniform distribution between a and b. The program *Gauss* μ σ denotes the Gaussian distribution with the average μ and the standard deviation σ.

Our implementation uses Isabelle/HOL terms directly. This approach is similar to CryptHOL by Basin et al. [8, 33], where they have embedded functional probabilistic programs for discrete distributions in order to verify cryptographic algorithms. The benefit is that it is much more readable and easier to work with terms when writing programs and reasoning about programs. Our qbs prover presented in Section 5.4 almost automates type checking. As we will demonstrate in later sections, program verification can be done directly in Isabelle/HOL.

The *query* Command

The language `HPProg` supports conditioning with the *query* command. The *query* command works as a *subroutine* that describes a conditional distribution. The *query* has the following type:

$$\textit{query} \in \textit{monadM-qbs}\ X \Rightarrow_Q (X \Rightarrow_Q \overline{\mathbb{R}}_{\geq 0}) \Rightarrow_Q \textit{monadM-qbs}\ X\ .$$

For a prior distribution s and a likelihood f, *query* s f returns the posterior distribution. The *query* command is defined through two operators: *density-qbs* (`scale` in `HPProg`) and *normalize-qbs*.

definition $\textit{query} \equiv (\lambda s\ f.\ \textit{normalize-qbs}\ (\textit{density-qbs}\ s\ f))$

[2]Because the s-finite measure monad is commutative, we have

$$p \bigotimes_{Qmes} q = q \gg\!= (\lambda y.\ p \gg\!= (\lambda x.\ \textit{return}_Q\ (X \bigotimes_Q Y)\ (x,y))).$$

As we observed at the end of Section 5.5, *density-qbs* and *normalize-qbs* have the following types and property.

$$\mathit{density}_Q \in \mathit{monadM\text{-}qbs}\ X \Rightarrow_Q (X \Rightarrow_Q \overline{\mathbb{R}}_{\geq 0}) \Rightarrow_Q \mathit{monadM\text{-}qbs}\ X$$

$$\mathit{normalize}_Q \in \mathit{monadM\text{-}qbs}\ X \Rightarrow_Q \mathit{monadM\text{-}qbs}\ X$$

$$(\int_Q x.\ g\ x\ \partial(\mathit{density}_Q\ s\ f)) = (\int_Q x.\ f\ x * g\ x\ \partial s)$$

The operator *density-qbs* returns the *unnormalized conditional distribution*. It takes a qbs-measure s and a non-negative function f and rescales s with the density function f. The operator *normalize-qbs* normalizes a qbs-measure s on X. If *qbs-l* $s\ X = 0$ or ∞, then *normalize-qbs* s returns the null-measure on X.

As a simple example, the Gauss distribution is expressed with its density function and $\mathit{lborel}_Q \in \mathit{monadM\text{-}qbs}\ \mathbb{R}$.

$$\mathit{Gauss}\ \mu\ \sigma = \mathit{query}\ \mathit{lborel}_Q\ (\mathit{normal\text{-}density}\ \mu\ \sigma)$$

Remark 6.3. Our language does not support inputs of probabilistic models as in Stan (Fig. 6.1). We need to specify a prior and a likelihood function explicitly.

The *condition* Command

We introduce the *condition* command, which produces a conditional distribution with a predicate. The *condition* command has the following type and is defined using the *query* command and the indicator function as follows.

definition *condition s P* ≡ *query s* (λ*x*. *if P x then 1 else 0*)
lemma *condition* ∈ *monadM-qbs X* ⇒$_Q$ (*X* ⇒$_Q$ 𝔹) ⇒$_Q$ *monadM-qbs X*

6.2.2 Example: Two Dice

Let us start with a simple example by Sampson [40, Section 2.3]. This example uses two language features: sampling and conditioning. We consider the following problem.

- We roll two dice.
- We observe at least one die is 4.
- What is the sum of the two dice?

We describe the distribution of the sum of the two dice as follows.

```
definition two-dice :: nat qbs-measure where
two-dice ≡ do {
             let die1 = die;
             let die2 = die;
             let twodice = die1 ⊗_Qmes die2;
             (x,y) ← condition twodice (λ(x,y). x = 4 ∨ y = 4);
             return_Q ℕ (x + y)
           }
```

Here, *die* ∈ *monadM-qbs* ℕ denotes the distribution of rolling a fair die. The program picks a sample from the conditional distribution, then returns the sum of the dice. The program *two-dice* has the following type.

```
lemma two-dice ∈ monadM-qbs ℕ
  by(simp add: two-dice-def)
```

We show the probabilities where the program takes each possible value.

```
lemma
 𝒫(x in two-dice. x = 5)  = 2 / 11   𝒫(x in two-dice. x = 6)  = 2 / 11
 𝒫(x in two-dice. x = 7)  = 2 / 11   𝒫(x in two-dice. x = 8)  = 1 / 11
 𝒫(x in two-dice. x = 9)  = 2 / 11   𝒫(x in two-dice. x = 10) = 2 / 11
```

6.2.3 Example: What time is it?

We formalize the example from Staton [48, Section 2.2]. This example uses two language features: higher-order functions and conditioning. Let us consider the following situation.

- We want to know what time it is.
- We know the rate of bikes per hour, which depends on time.
- We observed a 1 minute gap between two bikes.
- What time is it?

This situation is described as the following model.

$$T \sim \mathrm{Uniform}(0, 24),$$
$$X \sim \mathrm{Exponential}(f(T)).$$

The variable T denotes the time and X corresponds to the observed gap ($X = 1$ minute in this case). The value $f(T)$ is the rate of bikes per hour at time T. Thus, the gap between two bikes at time T follows the exponential distribution with the parameter $f(T)$.

We write this situation as the program *whattime*.

```
definition whattime :: (real ⇒ real) ⇒ real qbs-measure where
whattime ≡ (λf. do {
                  let T = Uniform 0 24 in
                  query T (λt. let r = f t in
                               exponential-density r (1 / 60))
                })
```

The program *whattime* receives a function f which determines the rate of bikes per hour. Then, the program returns the posterior after observing a 1 minute (1/60 hour) gap between two bikes. The likelihood is calculated using the density function *exponential-density* of the exponential distribution. The term *whattime* is a program with the following type.

```
lemma whattime ∈ (ℝ ⇒Q ℝ) ⇒Q monadM-qbs ℝ
  by(simp add: whattime-def)
```

As explained by Staton, the posterior is calculated as follows.

```
lemma
  assumes f ∈ ℝ ⇒_Q ℝ and U ∈ sets ℝ and ⋀t. f t ≥ 0
  defines N ≡ (∫ t∈{0<..<24}. (f t * exp (− 1/ 60 * f t)) ∂lborel)
  defines N' ≡ (∫⁺ t∈{0<..<24}. (f t * exp (− 1/ 60 * f t)) ∂lborel)
  assumes N' ≠ 0 and N' ≠ ∞
 shows 𝒫(t in whattime f. t ∈ U) = (∫ t∈{0<..<24}∩U. (f t * exp (− 1/ 60 * f t)) ∂lborel) / N
```

Note that N has the type reals, and N' has the type extended non-negative reals.

6.2.4 Example: Monte Carlo Approximation

The third example is the Monte Carlo approximation presented in the work by Sato et al. [41]. The Monte Carlo approximation tries to infer the expected value $\mu = \mathbb{E}_{x\sim d}[h(x)]$ from n samples. For a sample sequence $[x_0, \ldots, x_{n-1}]$, the approximation value is $\hat{\mu_n} = \sum_{i<n} \frac{1}{n} h(x_i)$. The algorithm *montecarlo* receives a distribution d, a morphism h, and a natural number n, then returns the approximation value.

```
primrec montecarlo :: 'a qbs-measure ⇒ ('a ⇒ real) ⇒ nat ⇒ real qbs-measure where
montecarlo - - 0       = return_Q ℝ 0 |
montecarlo d h (Suc n) = do {
                          m ← montecarlo d h n;
                          x ← d;
                          return_Q ℝ ((h x + m * real n) / (real (Suc n)))
                         }
```

We can prove that *montecarlo* is a program of the following type just by unfolding the definition[3] thanks to our qbs prover (Section 5.4).

```
lemma montecarlo ∈ monadM-qbs X ⇒_Q (X ⇒_Q ℝ) ⇒_Q ℕ ⇒_Q monadM-qbs ℝ
  by(simp add: montecarlo-def)
```

According to the weak law of large numbers, $\hat{\mu_n}$ converges in probability to the true expected value μ if $\mathbb{E}_{x\sim d}[(h(x) - \mu)^2] < \infty$. More concretely, $\lim_{n\to\infty} \Pr[|\hat{\mu}_n - \mu| \geq \epsilon] = 0$ holds for all $\epsilon > 0$. The statement is proved in two steps. First, we show that both the expected value and the variance of *montecarlo d h n* are finite.

```
lemma
  assumes d ∈ monadP-qbs X and h ∈ X ⇒_Q ℝ
     and qbs-integrable d h and qbs-integrable d (λx. h x * h x)
  shows qbs-integrable (montecarlo d h n) (λx. x)
    and qbs-integrable (montecarlo d h n) (λx. x * x)
```

The integrability assumptions ensure that the expectation and the variance of h with respect to d exist as finite values.

Next, we show the main statement.

[3] Internally, the **primrec** command defines a primitive recursive function using recursors such as *rec-nat* and *rec-list*.

```
lemma
  assumes d ∈ monadP-qbs X and h ∈ X ⇒Q ℝ
      and qbs-integrable d h and qbs-integrable d (λx. h x * h x)
      and e > 0 and n > 0
      and (∫Q x. h x ∂d) = μ and (∫Q x. (h x − μ)² ∂d) = σ²
    shows 𝒫(y in montecarlo d h n. |y − μ| ≥ e) ≤ σ² / (real n * e²)
```

Notice that the statement gives an upper bound of the probability for each $n > 0$. The probability converges to 0 as $n \to \infty$.

6.2.5 Example: Gaussian Mean Learning

As the final example, let us formalize the example from Sato et al. [41, Section 8.2]. We implement the Gaussian Mean Learning algorithm and prove two properties: convergence and stability under change of priors. In a common situation in statistical modeling or machine learning, we try to infer unknown parameters from a sample list. For instance, let us consider the following situation.

- We want to know the mean of a Gaussian distribution with a known standard deviation.
- We have a sample sequence from the Gaussian distribution.
- What is the posterior of the mean?

This situation is described as the following model.

$$X_i \sim \text{Gauss}(\mu, \sigma), \quad i = 1, \ldots, n$$

The following algorithm does Bayesian learning of the mean of a Gaussian distribution with a known standard deviation σ from a sample list.

```
primrec GaussLearn' :: [real, real qbs-measure, real list] ⇒ real qbs-measure where
  GaussLearn' - p [] = p
| GaussLearn' σ p (y#ls) = query (GaussLearn' σ p ls) (normal-density y σ)
```

The term *normal-density* $y\ \sigma$ is the density function of the Gaussian distribution *Gauss* $y\ \sigma$ with mean y.

The program *GaussLearn'* receives a standard deviation σ, a prior p and a sample list L. In each iteration, the program picks a sample from L, then updates the prior. Our qbs prover can show that *GaussLearn'* is a program because *GaussLearn'* is a primitive recursive function.

```
lemma GaussLearn' ∈ ℝ ⇒Q monadM-qbs ℝ ⇒Q list-qbs ℝ ⇒Q monadM-qbs ℝ
  by (simp add: GaussLearn'-def)
```

From now on, we fix $\sigma > 0$ and abbreviate *GaussLearn'* σ as *GaussLearn*.

The first property, convergence, is described as follows.

```
lemma
  assumes ξ > 0 and n = length L
  shows GaussLearn (Gauss δ ξ) L =
        Gauss ((Total L * ξ² + δ * σ²) / (n * ξ² + σ²)) (sqrt ((ξ² * σ²) / (n * ξ² + σ²)))
```

Here, the program *Total* sums up all elements of a list. The above statement says that if the prior of the mean is *Gauss* δ ξ, then the posterior is also a Gaussian distribution. Furthermore, its mean and standard deviation are close to the average of the samples and *0*, respectively, when n is sufficiently large.

Next, let us see the second property, stability under change of priors. We show that if we run *GaussLearn* from two different priors and give a large sample list whose average is bounded, then the resulting posteriors will be close. We measure the difference between distributions by the Kullback-Leibler (KL) divergence. The KL divergence is provided as *KL-divergence* in the standard Isabelle/HOL library. If p and q are probability distributions on $\mathbb{R}$ which have positive density functions f and g, respectively, then we have the following well-known form of KL divergence.

```
KL-divergence (exp 1) p q = (∫ x. g x * ln (g x / f x) ∂lborel)
```

The second property is stated as follows.

```
lemma GaussLearn-KL-divergence:
  fixes a b c d ε K :: real
  assumes ε > 0 and b > 0 and d > 0
  shows ∃ N. ∀ L. length L > N ⟶ |Total L / length L| < K ⟶
              KL-divergence (exp 1) (GaussLearn (Gauss a b) L) (GaussLearn (Gauss c d) L) < ε
```

Intuitively, the above property says that if we run *GaussLearn* with two different Gauss distributions, then we can make the distance of posteriors as close as we want with a large sample list whose average is bounded.

6.3 Differential Privacy

Differential privacy is a mathematical definition to express the privacy of databases. In general, differential privacy is discussed with measurable spaces. Sato and Katsumata showed that quasi-Borel spaces are also used to define differential privacy [42]. In this section, we formalize differential privacy using quasi-Borel spaces.

6.3.1 Definition of Differential Privacy

The notion of privacy is defined as follows. Let M be a randomized algorithm which receives a dataset (e.g., a list or a finite vector). The algorithm M is *private* if for any *adjacent* datasets D and D', then $M(D)$ and $M(D')$ behave similarly. Since $M(D)$ and $M(D')$ are interpreted as probability measures in the semantics, we evaluate the similarity of them using these measures.

Definition 6.4 (Differential Privacy [16])**.** Let X and Y be measurable spaces, R a symmetric relation on X, and $M : X \to G(Y)$ measurable function. Then, M is (ε, δ)-*differentially private (DP) with respect to* R if for all $(D, D') \in R$ and $A \in \Sigma_Y$,

$$M(D)(A) \leq e^{\varepsilon} M(D')(A) + \delta.$$

The differential privacy is also defined through the divergence[4] Δ^{ε}.

[4]A divergences is a kind of metrics. However, it might not be a metric in the mathematical sense. For instance, the KL divergence does not satisfy the symmetry and triangular-inequality.

Definition 6.5 (Divergence Δ^ε [7])**.** Let X be a measurable space, $\varepsilon \geq 0$, and $\mu, \nu \in G(X)$. Then, the divergence Δ^ε is defined as follows.

$$\Delta^\varepsilon_X(\mu, \nu) \stackrel{\text{def}}{=} \sup\{\mu(A) - e^\varepsilon \nu(A) \mid A \in \Sigma_X\}.$$

Lemma 6.6.

$$\Delta^\varepsilon_X(\mu, \nu) \leq \delta \iff \forall A \in \Sigma_X,\ \mu(A) - e^\varepsilon \nu(A) \leq \delta.$$

Corollary 6.7.

$$M \text{ is } (\varepsilon, \delta)\text{-DP w.r.t. } R \iff \forall (D, D') \in R,\ \Delta^\varepsilon_Y(M(D), M(D')) \leq \delta.$$

The differential privacy and the divergence Δ^ε have several basic properties. They enable us to prove the privacy of large algorithms from its small components. We only show the properties of the divergence Δ^ε.

Lemma 6.8. Let $\mu, \nu \in G(X)$ and $f, g : X \to G(Y)$ measurable functions. Then, the following hold.

- (Non-negativity) $0 \leq \Delta^\varepsilon_X(\mu, \nu)$.
- (Reflexivity) $\Delta^0_X(\mu, \mu) = 0$.
- (Anti-monotonicity) If $\varepsilon \leq \varepsilon'$, then $\Delta^\varepsilon_X(\mu, \nu) \geq \Delta^{\varepsilon'}_X(\mu, \nu)$.
- (Composability) If $\Delta^\varepsilon_X(\mu, \nu) \leq \delta$ and $\Delta^{\varepsilon'}_Y(f(x), g(x)) \leq \delta'$ for all $x \in X$, then $\Delta^{\varepsilon+\varepsilon'}_Y(\mu \gg\!= f, \nu \gg\!= g) \leq \delta + \delta'$.

6.3.2 Differential Privacy using Quasi-Borel Spaces

The differential privacy is also discussed with quasi-Borel spaces and the probability monad $\mathscr{P}$ by using the measurable function $l_X : L(\mathscr{P}(X)) \to G(L(X))$.

Definition 6.9. (Differential Privacy in **QBS**) Let X and Y be quasi-Borel spaces, R a symmetric relation on X, and $M : X \to \mathscr{P}(Y)$ a morphism. Then, M is $(\varepsilon, \delta)_{\mathbf{QBS}}$-differentially private (DP) with respect to R if $l_Y \circ M$ is (ε, δ)-DP with respect to R.

Definition 6.10. Let X be a quasi-Borel space, $\varepsilon \geq 0$, and $p,\ q \in \mathscr{P}(X)$. Then, the divergence ${}^{\mathrm{Q}}\Delta$ is defined by ${}^{\mathrm{Q}}\Delta^\varepsilon_X(p, q) \stackrel{\text{def}}{=} \Delta^\varepsilon_{L(X)}(l_X(p), l_X(q))$.

Remark 6.11. If $M : X \to \mathscr{P}(Y)$ is a morphism, then $l_Y \circ M$ is measurable since $M \in \mathbf{QBS}(X, \mathscr{P}(Y)) \subseteq \mathbf{Meas}(L(X).L(\mathscr{P}(Y)))$ and l_Y is also measurable. Thus, Definition 6.9 is well-defined.

Remark 6.12. When we use only standard Borel spaces, quasi-Borel spaces can treat usual differential privacy because $l_Y^{-1} \circ M : R(X) \to \mathscr{P}(R(Y))$ is a morphism for a non-empty standard Borel space Y and a measurable function $M : X \to G(Y)$.

The properties of differential privacy and the divergence Δ^ε still hold in a quasi-Borel setting thanks to l being a monad opfunctor (equation (5.4)). Most of them are easily obtained from ones in measure theory. For instance, the composability of ${}^{\mathrm{Q}}\Delta$ is derived using the following equation.

$$\begin{aligned} {}^{\mathrm{Q}}\Delta^{\varepsilon+\varepsilon'}_Y(p \gg\!= f, q \gg\!= g) &= \Delta^{\varepsilon+\varepsilon'}_{L(Y)}(l_Y(p \gg\!= f), l_Y(q \gg\!= g)) \\ &= \Delta^{\varepsilon+\varepsilon'}_{L(Y)}(l_X(p) \gg\!=_G l_Y \circ f, l_X(q) \gg\!=_G l_Y \circ g) \end{aligned}$$

6.3.3 Implementation in Isabelle/HOL

We formalize the differential privacy using quasi-Borel spaces. We built our formalization on the work by Sato and Minamide [43]. They define the constants denoting Δ^ε and differential privacy as follows.

DP-divergence	::	*'a measure* ⇒ *'a measure* ⇒ *real* ⇒ *ereal*
DP-divergence μ ν ε	=	$\Delta^\varepsilon(\mu, \nu)$
differential-privacy	::	(*'a* ⇒ *'b measure*) ⇒ *'a rel* ⇒ *real* ⇒ *real* ⇒ *bool*
differential-privacy M R ε δ	⟺	M is (ε, δ)-DP w.r.t. R

We define the divergence ${}^{\mathrm{Q}}\Delta$ and differential privacy through those constants.

```
definition DP-qbs-divergence :: 'a qbs-measure ⇒ 'a qbs-measure ⇒ real ⇒ ereal
  (DP'-divergence_Q) where
  DP-divergence_Q p q ε ≡ DP-divergence (qbs-l p) (qbs-l q) ε

definition DP-qbs :: ('a ⇒ 'b qbs-measure) ⇒ 'a rel ⇒ real ⇒ real ⇒ bool
  (differential'-privacy_Q) where
  differential-privacy_Q M ≡ differential-privacy (λx. qbs-l (M x))
```

6.3.4 Example: Naive Random Noisy Max

As an example, we prove the differential privacy of the naive random noisy max algorithm. This algorithm is a simplified version of the Report Noisy Max algorithm, where argmax is replaced by max, presented by Dwork and Roth[17]. The algorithm works for $\varepsilon > 0$ as follows.

- Input a dataset $[x_0, \ldots, x_n]$ as a list of real numbers.
- Add noise by Laplace distribution to each elements. More precisely, take an independent sample sequence $x'_i \leftarrow \mathrm{Laplace}(1/\varepsilon, x_i)$.
- Return the maximum value of $[x'_0, \ldots, x'_n]$.

In Isabelle/HOL, this algorithm is written as a primitive recursive function.

```
primrec qbs-NaiveRNM :: real ⇒ real list ⇒ real qbs-measure where
  qbs-NaiveRNM ε [] = return_Q ℝ 0 |
  qbs-NaiveRNM ε (x # xs) =
  (case xs of
    Nil ⇒ Lap-mechanism_Q (1 / ε) x |
    y#ys ⇒ do {
                  x1 ← Lap-mechanism_Q (1 / ε) x;
                  x2 ← qbs-NaiveRNM ε xs;
                  return_Q ℝ (max x1 x2)
                })
```

When the dataset is empty, the function returns the *default* value. The function *qbs-DP-NaiveRNM* has the following type. This is automatically proved by the qbs prover.

```
lemma qbs-NaiveRNM ∈ ℝ_Q ⇒_Q list-qbs ℝ ⇒_Q monadP-qbs ℝ_Q
```

The program *qbs-DP-NaiveRNM* satisfies the following differential privacy.

definition *adj-naive-RNM* :: *real* ⇒ (*real list* × *real list*) *set* **where**
adj-naive-RNM r ≡ {(*xs*,*ys*). *length xs* = *length ys* ∧ (∑ *i*<*length xs*. |*nth xs i* − *nth ys i*|) ≤ *r*}

theorem *qbs-DP-NaiveRNM*:
assumes ε > *0*
shows *differential-privacy*$_Q$ (*qbs-NaiveRNM* ε) (*adj-naive-RNM r*) (*r* ∗ ε) *0*

The proof is almost the same as the measurable one.

The benefit of using quasi-Borel spaces in the differential privacy is that we can treat higher-order programs. Although this example is a first-order algorithm, one sometimes wants to use higher-order functions, e.g., abstracting *weight* functions. Quasi-Borel spaces are convenient in such situations. Another benefit is that it is easy to check morphismness of functions. Proving measurability of recursive functions on lists is sometimes tedious. Primitive recursive functions on lists are defined using the term *rec-list* :: $'a$ ⇒ ($'b$ ⇒ $'b$ *list* ⇒ $'a$ ⇒ $'a$) ⇒ $'b$ *list* ⇒ $'a$, which is a higher-order term whose arguments include a function. Hence, showing measurability of programs containing *rec-list* is not so straightforward because measurable spaces do not have function spaces in general. In the theory of quasi-Borel spaces, the morphismness of programs including *rec-list* is shown in the same way as other programs. Our qbs prover also automates solving morphismness.

Chapter 7

Conclusion

We have formalized standard Borel spaces, the Lévy-Prokhorov metric, and quasi-Borel spaces in Isabelle/HOL. Formalization of standard Borel spaces and the Lévy-Prokhorov metric includes important results which are often used in applied probability theory, e.g., Kuratowski's theorem, Prokhorov's theorem, and the Riesz representation theorem. We also formally constructed the notion of quasi-Borel spaces and the s-finite measure monad. Then, we applied them to verify several probabilistic program examples and differential privacy. When working with our quasi-Borel space library, our qbs prover reduces the cost of proofs showing that terms are members of quasi-Borel spaces.

Efforts

Our formalization consists of around 36,400 lines of code (LOC), including comments and blank lines[1]. The line number is measured by the `wc -l` command.

AFP entry	LOC
Differential Privacy using Quasi-Borel Spaces	430
Coproduct Measure	1,761
The Lévy-Prokhorov Metric	6,598
The Riesz Representation Theorem	4,410
Disintegration Theorem	2,853
S-Finite Measure Monad on Quasi-Borel Spaces	11,896
Standard Borel Spaces	8,451
Total	**36,399**

Future works

Algorithms terminating in probability 1 Probabilistic programs sometimes treat *almost-sure terminating* algorithms, i.e., algorithms halt in probability 1. An approach to give semantics to almost-sure terminating algorithms is using the least fixed-point as in the standard semantics of `While` programs. Basin et al. implemented discrete probabilistic programs [8] which can treat almost-sure termination in Isabelle/HOL by showing that the set of sub-probability mass functions forms a complete partial order (cpo). The semantics of almost-sure

[1]The entry "Quasi-Borel Spaces" is omitted from the list because "S-Finite Measure Monad on Quasi-Borel Spaces" contains most of reconstructed contents of it.

terminating programs with quasi-Borel spaces is studied by Vákár et al. [52] and Geoffroy [19], where they use *ω-quasi-Borel spaces* which are quasi-Borel spaces with ωcpo structures. In order to use these semantics in Isabelle/HOL, we need to implement ω-quasi-Borel spaces and prove their properties, such as the existence of the least fixed-point.

Advanced theories There are still many topics which have not been formalized. For instance, transportation theory and large deviation theory could be interesting future works because important theorems in their theory depend on theorems we have formalized in this thesis, e.g., Prokhorov's theorem. There are also other applications of quasi-Borel spaces. For instance, Sabok et al. applied quasi-Borel spaces to the semantics of the ν-calculus [39], a calculus for name generation. They showed that the semantics based on quasi-Borel spaces is fully abstract at first-order, i.e., two first order programs are observationally equivalent if and only if their interpretations are the same.

Application to implementation of probabilistic programs In this thesis, we only mention denotational semantics of probabilistic programs. Besides denotational semantics, verification of operational semantics and implementation of probabilistic programs are important future works. The execution of probabilistic programming languages normally consists of two parts. In the first part, the *compiler* transforms the density function of the posterior distribution. Then, in the second part, an MCMC algorithm generates samples from the posterior distribution. Some previous works formalized the first step. Tassarotti and Tristan verified a compiler for a subset of the Stan probabilistic programming language using Coq [49]. In Isabelle/HOL, Eberl et al. formalized a density compiler [18], which is developed by Bhat et al. [9]. We expect that quasi-Borel spaces would benefit verifying the implementation of higher-order probabilistic programming languages.

Appendix A

Appendix

A.1 Coproduct Measures

The category of measurable spaces **Meas** has coproducts. We explain its construction and measures on coproduct spaces.

Definition A.1. Let I be a set and $\{M_i\}_{i\in I}$ measurable spaces. The coproduct measurable space $\coprod_{i\in I} M_i = \{(i,x) \mid i \in I \wedge x \in M_i\}$ consists of the following σ-algebra.

$$\Sigma_{\coprod_{i\in I} M_i} = \{A \mid \forall i \in I.\, A_i \in \Sigma_{M_i}\}, \quad \text{where } A_i = \{x \mid (i,x) \in A\}.$$

It is easy to check that $\left(\coprod_{i\in I} M_i, \Sigma_{\coprod_{i\in I} M_i}\right)$ forms a measurable space.

Lemma A.2. Let I be a set, $\{M_i\}_{i\in I}$ measurable spaces, and N a measurable space. A function $f : \coprod_{i\in I} M_i \to N$ is measurable if and only if $(\lambda x.\, f(i,x)) : M_i \to N$ is measurable for all $i \in I$.

The countable coproduct space of standard Borel spaces is also a standard Borel space.

Lemma A.3. Let I be a countable set and $\{M_i\}_{i\in I}$ a family of standard Borel spaces. Then, $\coprod_{i\in I} M_i$ is a standard Borel space.

Proof. Without loss of generality, we assume that $I = \mathbb{N}$. From Kuratowski's theorem (Theorem 3.12) and Lemma 3.9, we have a family of Borel measurable sets $\{A_n\}_{n\in\mathbb{N}}$ such that $A_n \subseteq [n, n+1)$ and $M_n \cong A_n$ for all $n \in \mathbb{N}$. Then, $\coprod_{i\in I} M_i \cong \bigcup_{n\in\mathbb{N}} A_n$ and $\bigcup_{n\in\mathbb{N}} A_n$ is a standard Borel space. Thus, $\coprod_{i\in I} M_i$ is a standard Borel space by Lemma 3.8. □

There is the *natural* measures on coproduct spaces.

Lemma A.4. Let I be a set, $\{M_i\}_{i\in I}$ measurable spaces, and μ_i a measure on M_i for all $i \in I$. Then, there is the unique measure $\coprod_{i\in I} \mu_i$ on $\coprod_{i\in I} M_i$ such that

$$\left(\coprod_{i\in I} \mu_i\right)(A) = \sum_{i\in I} \mu_i(A_i) \quad \text{for } A \in \Sigma_{\coprod_{i\in I} M_i} \tag{A.1}$$

Notice that the sum in the equation (A.1) is defined on an arbitrary set[1]. The coproduct measure satisfies axioms of measure because the exchange of order of sum holds for sequences of extended non-negative values. The uniqueness is obvious from its definition.

Similar to the measure, the integral is also decomposed into the sum of integrals.

Lemma A.5. Let I be a set, $\{M_i\}_{i\in I}$ measurable spaces, μ_i a measure on M_i for all $i \in I$.

- If $f : \coprod_{i\in I} M_i \to \overline{\mathbb{R}}_{\geq 0}$ is a non-negative measurable function, then
$$\int f \mathrm{d}\left(\coprod_{i\in I} \mu_i\right) = \sum_{i\in I} \int f(i,x)\mu_i(\mathrm{d}x).$$
- If $f : \coprod_{i\in I} M_i \to \mathbb{R}$ is measurable and integrable with respect to $\coprod_{i\in I} \mu_i$, then the above equation holds.

Coproduct Measures in Isabelle/HOL

We define the coproduct measure in Isabelle/HOL. Same as the product measure, binary coproduct measures and general coproduct measures are defined separately.

```
definition copair-measure :: ['a measure, 'b measure] ⇒ ('a + 'b) measure (infixr ⨁M 65) where
M ⨁M N = measure-of (space M <+> space N)
                    ({Inl ‘ A |A. A ∈ sets M} ∪ {Inr ‘ A|A. A ∈ sets N})
                    (λA. emeasure M (Inl −‘ A) + emeasure N (Inr −‘ A))
```

```
definition coPiM :: ['i set, 'i ⇒ 'a measure] ⇒ ('i × 'a) measure where
coPiM I Mi ≡ measure-of
             (SIGMA i:I. space (Mi i))
             {A. A⊆(SIGMA i:I. space (Mi i)) ∧ (∀ i∈I. Pair i −‘ A ∈ sets (Mi i))}
             (λA. (∑∞i∈I. emeasure (Mi i) (Pair i −‘ A)))
```

The term *measure-of* M $\mathcal{A}$ μ denotes the measure μ on $(M, \sigma[\mathcal{A}])$.

Since we formalize both of the coproduct quasi-Borel spaces and the coproduct measurable spaces, we are ready to prove the statement 3 in Lemma 5.12.

```
lemma r-preserve-copair: R (M ⨁M N) = R M ⨁Q R N
```

```
lemma r-preserve-coproduct:
  assumes countable I
  shows R (coPiM I M) = (ΠQ i∈I. R (M i))
```

In the proof of Lemma A.4, we use the following lemmas to show the change of orders of the infinite sum.

```
lemma infsum-Sigma:
  fixes A :: 'a set and B :: 'a ⇒ 'b set
    and f :: 'a × 'b ⇒ 'c::{comm-monoid-add, t2-space, uniform-space}
  assumes uniformly-continuous-on UNIV (λ(x::'c,y). x+y)
  assumes f summable-on (Sigma A B)
  assumes ⋀x. x∈A ⟹ (λy. f (x, y)) summable-on (B x)
  shows infsum f (Sigma A B) = infsum (λx. infsum (λy. f (x, y)) (B x)) A
```

[1]The sum of $\{x_i\}_{i\in I}$ is defined as the limit of $(\lambda A.\ \sum_{i\in A} x_i)$ with respect to a filter on 2^I. The details are found in the book by Conway (Definition 4.11 [14]) or Isabelle/HOL's library `HOL/Analysis/Infinite_Sum.thy`.

corollary *compact-uniformly-continuous*:
fixes f :: $'a$ :: *metric-space* ⇒ $'b$ :: *metric-space*
assumes *continuous-on* S f **and** *compact* S
shows *uniformly-continuous-on* S f

Hence, we need to interpret *ennreal* as a metric space if we try to show the change of orders by using the above lemmas. However, there is no natural metric on $\overline{\mathbb{R}}_{\geq 0}$, although $\overline{\mathbb{R}}_{\geq 0}$ is a Polish space (thus, a metrizable space). In order to avoid the instantiation of *metric-space* by *ennreal* with an unnatural metric, we define a copy of *ennreal*. Then, we prove the change of order for the copied type.

typedef *ennreal'* = *UNIV* :: *ennreal set*

instantiation *ennreal'* :: *metric-space*
begin
...
end

lemma *infsum-swap-ennreal'*:
fixes f :: - ⇒ - ⇒ *ennreal'*
shows *infsum* (λx. *infsum* (λy. f x y) B) A = *infsum* (λy. *infsum* (λx. f x y) A) B

Finally, we obtain the change of order for *ennreal* by transferring the above theorem with the following lemma.

lemma *infsum* f A = *Rep-ennreal'* (*infsum* ((λx. *Abs-ennreal'* (f x))) A)

Note that the *infsum* on the left-hand side is on *ennreal* while the *infsum* on the right-hand side is on *ennreal'*.

Bibliography

[1] R. Affeldt and C. Cohen. Measure construction by extension in dependent type theory with application to integration. *Journal of Automated Reasoning*, 67(3):28:1–28:27, 2023.

[2] R. Affeldt, C. Cohen, and A. Saito. Semantics of probabilistic programs using s-finite kernels in Coq. In *Proceedings of the 12th ACM SIGPLAN International Conference on Certified Programs and Proofs*, CPP 2023, page 3–16, New York, NY, USA, 2023. Association for Computing Machinery.

[3] R. J. Aumann. Borel structures for function spaces. *Illinois Journal of Mathematics*, 5(4):614 – 630, 1961.

[4] J. Avigad, J. Hölzl, and L. Serafin. A formally verified proof of the central limit theorem. *Journal of Automated Reasoning*, 59(4):389–423, 2017.

[5] F. Baccelli, B. Blaszczyszyn, and M. Karray. *Random Measures, Point Processes, and Stochastic Geometry*. Inria, Jan. 2020.

[6] C. Ballarin. Tutorial to locales and locale interpretation. *Contribuciones científicas en honor de Mirian Andrés Gómez*, pages 123–140, 2010.

[7] G. Barthe and F. Olmedo. Beyond differential privacy: Composition theorems and relational logic for f-divergences between probabilistic programs. In F. V. Fomin, R. Freivalds, M. Kwiatkowska, and D. Peleg, editors, *Automata, Languages, and Programming*, pages 49–60, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[8] D. A. Basin, A. Lochbihler, and S. R. Sefidgar, Sefidgar. CryptHOL: Game-based proofs in higher-order logic. *Journal of Cryptology*, 33:494–566, 2020.

[9] S. Bhat, J. Borgström, A. D. Gordon, and C. Russo. Deriving probability density functions from probabilistic functional programs. In N. Piterman and S. A. Smolka, editors, *Tools and Algorithms for the Construction and Analysis of Systems*, pages 508–522, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[10] P. Billingsley. *Convergence of Probability Measures*. Wiley series in probability and mathematical statistics, Tracts on probability and statistics. Wiley, New York, United States, 1968.

[11] M. Biskup. Lecture note of math245b in UCLA. https://web.archive.org/web/20210506130459/https://www.math.ucla.edu/~biskup/245b.1.20w/, 2020. Accessed: January 17. 2023.

[12] S. Boldo, F. Clément, F. Faissole, V. MARTIN, and M. Mayero. A Coq formalization of Lebesgue integration of nonnegative functions. Research Report RR-9401, Inria, France, 2021.

[13] B. Carpenter, A. Gelman, M. D. Hoffman, D. Lee, B. Goodrich, M. Betancourt, M. Brubaker, J. Guo, P. Li, and A. Riddell. Stan: A probabilistic programming language. *Journal of Statistical Software*, 76(1):1–32, 2017.

[14] J. Conway. *A Course in Functional Analysis*. Graduate Texts in Mathematics. Springer New York, 2013.

[15] J.-D. Deuschel and D. W. Stroock. *Large Deviations*, volume 137 of *Pure and Applied Mathematics*. Elsevier Science, 1989.

[16] C. Dwork, K. Kenthapadi, F. McSherry, I. Mironov, and M. Naor. Our data, ourselves: Privacy via distributed noise generation. In S. Vaudenay, editor, *Advances in Cryptology - EUROCRYPT 2006*, pages 486–503, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[17] C. Dwork and A. Roth. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3–4):211–407, 2014.

[18] M. Eberl, J. Hölzl, and T. Nipkow. A verified compiler for probability density functions. In *European Symposium on Programming (ESOP 2015)*, volume 9032 of *LNCS*, pages 80–104. Springer, 2015.

[19] G. Geoffroy. Extensional denotational semantics of higher-order probabilistic programs, beyond the discrete case, 2021.

[20] M. Giry. A categorical approach to probability theory. In B. Banaschewski, editor, *Categorical Aspects of Topology and Analysis*, pages 68–85, Berlin, Heidelberg, 1982. Springer Berlin Heidelberg.

[21] S. Gouëzel. `Basic.lean`. https://github.com/leanprover-community/mathlib4/blob/master/Mathlib/MeasureTheory/Constructions/Polish/Basic.lean. Accessed September 29th 2024.

[22] F. Haftmann. Haskell-style type classes with Isabelle/Isar. https://isabelle.in.tum.de/doc/classes.pdf.

[23] C. E. Heil. Alaoglu's theorem. https://heil.math.gatech.edu/6338/summer08/section9f.pdf, 2008. Lecture notes on MATH 6338 (Real Analysis II) at Gerogia Insisute of Technology, Accessed: January 5, 2024.

[24] C. E. Heil. Nets, directed sets, and convergence. https://heil.math.gatech.edu/6338/summer08/section9b.pdf, 2008. Lecture notes on MATH 6338 (Real Analysis II) at Gerogia Insisute of Technology, Accessed: January 5, 2024.

[25] 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 2017. IEEE Press, 2017.

[26] J. Hölzl and A. Heller. Three chapters of measure theory in Isabelle/HOL. In M. van Eekelen, H. Geuvers, J. Schmaltz, and F. Wiedijk, editors, *Interactive Theorem Proving*, ITP 2011, pages 135–151. Springer Berlin Heidelberg, 2011.

[27] J. Hölzl, F. Immler, and B. Huffman. Type classes and filters for mathematical analysis in Isabelle/HOL. In *Proceedings of the 4th International Conference on Interactive Theorem Proving*, ITP 2013, page 279–294, Berlin, Heidelberg, 2013. Springer-Verlag.

[28] B. Huffman and O. Kunčar. Lifting and transfer: A modular design for quotients in Isabelle/HOL. In G. Gonthier and M. Norrish, editors, *Certified Programs and Proofs*, CPP 2013, pages 131–146. Springer International Publishing, 2013.

[29] A. S. Kechris. *Classical Descriptive Set Theory*. Graduate Texts in Mathematics. Springer New York, 1995.

[30] K. Kytölä. `LevyProkhorovMetric.lean`. https://github.com/leanprover-community/mathlib4/blob/master/Mathlib/MeasureTheory/Measure/LevyProkhorovMetric.lean. Accessed May 27th 2024.

[31] H. Lee. Vector spaces. *Archive of Formal Proofs*, August 2014. https://isa-afp.org/entries/VectorSpace.html, Formal proof development.

[32] P. Lévy. *Thèorie de l'addition des variables alèatoires*. Gauthier-Villars, Paris, 1937.

[33] A. Lochbihler. Probabilistic functions and cryptographic oracles in higher order logic. In P. Thiemann, editor, *Programming Languages and Systems*, pages 503–531, Berlin, Heidelberg, 2016. Springer Berlin Heidelberg.

[34] M. Maggesi. A formalization of metric spaces in HOL Light. *Journal of Automated Reasoning*, 60:237–254, 2018.

[35] T. Nipkow, M. Wenzel, and L. C. Paulson. *Isabelle/HOL: A Proof Assistant for Higher-Order Logic*. Springer-Verlag, Berlin, Heidelberg, 2002.

[36] L. C. Paulson. Porting the HOL Light metric space library. https://lawrencecpaulson.github.io/2023/07/12/Metric_spaces.html. Accessed: December 31. 2023.

[37] Y. V. Prokhorov. Convergence of random processes and limit theorems in probability theory. *Theory of Probability and Its Applications*, 1(2):157–214, 1956.

[38] W. Rudin. *Real and Complex Analysis, 3rd Ed.* McGraw-Hill, Inc., USA, 1987.

[39] M. Sabok, S. Staton, D. Stein, and M. Wolman. Probabilistic programming semantics for name generation. *Proc. ACM Program. Lang.*, 5(POPL), Jan. 2021.

[40] A. Sampson. Probabilistic programming. http://adriansampson.net/doc/ppl.html. Accessed: January 25. 2023.

[41] T. Sato, A. Aguirre, G. Barthe, M. Gaboardi, D. Garg, and J. Hsu. Formal verification of higher-order probabilistic programs: reasoning about approximation, convergence, bayesian inference, and optimization. *Proceedings of the ACM on Programming Languages*, 3(POPL):1–30, 2019.

[42] T. Sato and S. Katsumata. Divergences on monads for relational program logics. *Mathematical Structures in Computer Science*, 33(4–5):427–485, 2023.

[43] T. Sato and Y. Minamide. Formalization of differential privacy in Isabelle/HOL. In *Proceedings of the 14th ACM SIGPLAN International Conference on Certified Programs and Proofs*, CPP '25, page 67–82, New York, NY, USA, 2025. Association for Computing Machinery.

[44] A. Ścibior, O. Kammar, M. Vákár, S. Staton, H. Yang, Y. Cai, K. Ostermann, S. K. Moss, C. Heunen, and Z. Ghahramani. Denotational validation of higher-order bayesian inference. *Proc. ACM Program. Lang.*, 2(POPL), 2017.

[45] M. Shi. Nets and filters. https://www.uvm.edu/~smillere/TProjects/MShi20s.pdf, 2020. Accessed November 17th 2023.

[46] S. M. Srivastava. *A Course on Borel Sets*. Springer, 1998.

[47] S. Staton. Commutative semantics for probabilistic programming. In H. Yang, editor, *Programming Languages and Systems*, pages 855–879, Berlin, Heidelberg, 2017. Springer Berlin Heidelberg.

[48] S. Staton. *Foundations of Probabilistic Programming*, chapter Probabilistic Programs as Measures, page 43–74. Cambridge University Press, 2020.

[49] J. Tassarotti and J.-B. Tristan. Verified density compilation for a probabilistic programming language. *Proc. ACM Program. Lang.*, 7(PLDI), June 2023.

[50] The mathlib Community. The Lean mathematical library. In *Proceedings of the 9th ACM SIGPLAN International Conference on Certified Programs and Proofs*, CPP 2020, page 367–381. Association for Computing Machinery, 2020.

[51] R. Thiemann and A. Yamada. Matrices, jordan normal forms, and spectral radius theory. *Archive of Formal Proofs*, August 2015. https://isa-afp.org/entries/Jordan_Normal_Form.html, Formal proof development.

[52] M. Vákár, O. Kammar, and S. Staton. A domain theory for statistical probabilistic programming. *Proc. ACM Program. Lang.*, 3(POPL), Jan 2019.

[53] F. van Doorn. Measure theory. Lean Together 2021, https://leanprover-community.github.io/lt2021/slides/floris-measure.pdf. Accesed September 30th 2024.

[54] O. van Gaans. Probability measures on metric spaces. https://www.math.leidenuniv.nl/~vangaans/jancol1.pdf. Accessed: February 29. 2024.

[55] C. Villani. *Optimal Transport: Old and New*. Grundlehren der mathematischen Wissenschaften. Springer Berlin Heidelberg, 2008.

[56] H. Yang. Semantics of higher-order probabilistic programs with continuous distributions. https://www.cs.ox.ac.uk/people/hongseok.yang/talk/ProbProg17-semantics.pdf. Accessed: February 8. 2023.

[57] S. Zhang. Existence and application of optimal markovian coupling with respect to non-negative lower semi-continuous functions. *Acta Mathematica Sinica*, 16(2):261–270, 2000.