石井 大海（いしい ひろみ）

基本情報

DeepFlow 株式会社
E-Mail: konn.jinro@gmail.com
Web Site: https://konn-san.com
GitHub: https://github.com/konn
Zenn: https://zenn.dev/konn

教育・学位

博士（理学）	2019 年 3 月．筑波大学数理物質科学研究科数学専攻 博士後期課程・修了
	博士論文： Bidirectional Interplay between Mathematics and Computer Science: Safety and Extensibility in Computer Algebra and Haskell
修士（理学）	2016 年 3 月．筑波大学数理物質科学研究科 博士前期課程・修了
	修士論文： On Regularity Properties of Sets of Reals and Inaccessible Cardinals
学士（理学）	2014 年 3 月．早稲田大学基幹理工学部数学科・卒

職歴・採用歴

2019 年 4 月-現在　DeepFlow 株式会社 研究開発部.

Haskell による大規模並列数値計算ソルバー Elkurage の設計・開発に従事．当該ソルバーはスレッド並列や MPI 通信などを活かした Haskell による高性能プログラミングの技術を用い，数億セルの並列計算を実現．高い抽象性と拡張性・効率性を両立するための Haskell 内への埋め込みドメイン特化型言語の設計や，型システムの設計等を主に担当する．また，研究・産業用途のシミュレーションワークフロー・最適化ツールを，顧客との綿密なコミュニケーションを通じて設計・開発.

主な実績は以下の通り：

- 大規模数値計算ソルバ用の依存型つき Haskell 製 DSL の設計・実装
  - 個別の方程式系や，ソルバのプラグイン機構などを適切な依存型を用いて代数的に抽象化・実装
  - 任意カインドでラベル付け可能な拡張可能レコードを実装し，型制約の自動解消を行う型検査器プラグインも併せて開発
  - 詳細については，Haskell Day 2019 での発表資料 [J-1] やプレプリント [J-2] を参照.
- メッシュ生成ライブラリ Gmsh への Haskell バインディングを，Higher-Kinded Datatypes の技法などを用いて実装
- 産業・学術用シミュレーション GitOps ワークフローツールの開発
  - 単一の設定ファイルから必要な設定・入力をパラメトリックに生成
  - クラスタにインストール済のジョブマネージャまたはスレッド並列・Cloud Haskell などを用いた独自実装のクエジューラを通じて並列で多数のジョブを同時実行
  - ParaView の Python スクリプティング機能による自動可視化

- 各ワークフローのジョブ間の複雑な依存関係は Shake の規則により記述
- 実行結果のレポートや比較表を HTML などとして HTML や PDF で出力，R2 やローカルストレージ上に保存し Servant サーバよりサーブ

2019 年 4 月-**2020 年 3 月**　統計数理研究所 外来研究員

2017 年 4 月-**2019 年 3 月**　日本学術振興会特別研究員 DC2

研究課題：『実数の集合の性質の集合論的解明と工学的応用』

2014 年-**2017 年**　筑波大学数学類『計算機演習』ティーチング・アシスタント

2014 年 4 月　Google Summer of Code 2014 採択

題目：『Haskell による効率的な Gröbner 基底計算とそのための疎行列対角化アルゴリズムの実装』

2014 年 4 月-**2017 年 3 月**　筑波大学数学専攻計算機管理アルバイト（www-admin）

2010 年 10 月-**2014 年 3 月**　株式会社 Preferred Infrastructure アルバイト

2010 年 8 月-**9 月**　株式会社 Preferred Infrastructure インターン生

職歴欄参考文献

[J-1] Hiromi Ishii, **大規模数値計算を支える Haskell——Pragmatic Haskell in large-scale numerical computation**, Talk at Haskell Day 2019, Tokyo, Japan., 2019, URL: https://speakerdeck.com/konn/da-gui-mo-shu-zhi-ji-suan-wozhi-eru-haskell-nil-nil-pragmatic-haskell-in-large-scale-numerical-computation-nil-nil.

[J-2] _____, **Functional Pearl: Witness Me – constructive arguments must be guided with concrete witness**, preprint, 2021, DOI: 10.48550/arXiv.2103.11751, URL: https://arxiv.org/abs/2103.11751.

賞罰・特記事項

- 日本学術振興会特別研究員 DC2,（2017 年〜2019 年）
- 第十四回茗渓会賞（2016 年）
- 2013 年度早稲田大学基幹理工学部長賞最優秀賞（第一回），基幹理工学部卒業生総代

研究業績

査読付き会議論文

August 2021, **Automatic Differentiation with Higher Infinitesimals, or Computational Smooth Infinitesimal Analysis in Weil Algebra**. Computer Algebra in Scientific Computing 2021, Sochi, Russia.

September 2018, **A Purely Functional Computer Algebra System Embedded in Haskell**. Computer Algebra in Scientific Computing 2018, Lille, France.

September 2015. Oleg Kiselyov and Hiromi ISHII, **Freer Monads, More Extensible effects**. Haskell Symposium 2015, Vancouber, Canada.

学会発表

August 2021, **Automatic Differentiation with Higher Infinitesimals, or Computational Smooth Infinitesimal Analysis in Weil Algebra**. Computer Algebra in Scientific Computing 2021, Sochi, Russia, 査読あり.
September 2018, **A Purely Functional Computer Algebra System Embedded in Haskell**. Computer Algebra in Scientific Computing 2018, Lille, France, 査読あり.
March 2016, **Freer Monads, More Extensible Effects**. Programming and Programming Language Workshop (PPL) 2016, Okayama-prefecture, Japan, 査読あり.
November 2017, **Reflection Principle and construction of saturated ideals on $\mathcal{P}_{\omega_1}\lambda$**. Workshop on Iterated Forcing Theory and Cardinal Invariants, Kyoto-prefecture, Japan, 査読なし.

論文

特に、Oleg Kiselyov 氏との共著論文 “Freer Monads, More Extensible Effects” [5] は現代のイフェクトシステムの隆盛につながるもので、特筆に値する。

[1] ———, **On regularity properties of set of reals and inaccessible cardinals**, MA thesis, Tsukuba University, 2016, URL: http://hdl.handle.net/2241/00135591.

[2] ———, **A purely functional computer algebra system embedded in Haskell**, Computer Algebra in Scientific Computing (Lille, France), ed. by Vladimir P. Gerdt, Wolfram Koepf, and Werner M. Seiler, vol. 11077, Lecture Notes in Computer Science, Springer, Cham, 2018, pp. 288–303, ISBN: 978-3-319-99638-7, DOI: 10.1007/978-3-319-99639-4_20, arXiv: 1807.01456.

[3] ———, **Bidirectional interplay between mathematics and computer science: safety and extensibility in computer algebra and haskell**, PhD thesis, University of Tsukuba, 2019, DOI: 10.15068/00156548.

[4] ———, **Automatic differentiation with higher infinitesimals, or computational smooth infinitesimal analysis in weil algebra**, Computer Algebra in Scientific Computing (Sochi, Russia), ed. by François Boulier et al., vol. 11077, Lecture Notes in Computer Science, Springer, Cham, 2021, pp. 174–191, ISBN: 978-3-030-85164-4, DOI: 10.1007/978-3-030-85165-1_11, arXiv: 2106.14153.

[5] Oleg Kiselyov and Hiromi Ishii, **Freer monads, more extensible effects**, Proceedings of the 2015 ACM SIGPLAN Symposium on Haskell, Haskell '15, Vancouver, BC, Canada: ACM, 2015, pp. 94–105, ISBN: 978-1-4503-3808-0, DOI: 10.1145/2804302.2804319.

自然言語能力

日本語　母語
英語　流暢（参考記録：TOEIC 920 点（2012 年））

プログラミング言語能力

Haskell　エキスパート。公私ともに主要利用言語。
Python　中級（機械学習案件や可視化に利用経験あり。環境管理には Rye を使用）
Agda　趣味（代数的性質の証明や、Cubical Type Theory の利用経験あり）
JavaScript / TypeScript　中級（一部自動化や GitHub Actions などに）
Rust　初心者（非常に興味あり）

C　学部卒程度（Haskell の FFI バインディングライブラリを開発する程度の技能はあり）

技術要素

以下の技術要素に熟達している：

並列・分散処理　スレッド並列、STM、`async` 型ライブラリ、Cloud Haskell、Open MPI
ストリーム処理　Haskell においては `streaming` ライブラリーを利用。`conduit` 系統の利用経験も長い。
EDSL の設計　ドメインに応じた Haskell 内 DSL の設計・実装経験豊富。
線型型　リソース割り当てや管理などに利用。
コンテナ技術　Docker, Singularity
CI/CD　GitHub Actions, GitLab CI/CD
形式手法　性質ベーステスト（QuickCheck, Falsify, hypothesis）リグレッションテスト、依存型によるコンパイル時検証
クラウド技術　AWS, GCP, Azure, さくらのクラウド, Terraform, Packer

オープンソース開発実績（一部）

モノレポ依存関係管理ツール Guardian

URL: `https://github.com/deepflowinc/guardian`

DeepFlow のプロダクトは非常に多くの内製パッケージから成っており，それらの間の依存関係が複雑化することでビルド時間に大きな悪影響が出ていた．こうした状況を改善するため，パッケージを複数のグループに分類し，グループ間の依存関係に関する制約を設定させ CI で絶えずチェックさせることで，依存関係において疎結合・関心の分離を実現するためのツール Guardian の開発を社内で主導し，OSS として公開した．ツールの詳細については Zenn 上のブログ記事 [OSS-1] を参照．

Haskell 用 Cloudflare Worker ライブラリ

URL: `https://github.com/konn/ghc-wasm-earthly`

GHC 9.10 より実装された，WASM バックエンドの JavaScript FFI 機能を用いて，サーバレスウェブアプリ開発環境 Cloudflare Worker 上のアプリを Haskell で開発するためのライブラリを実装している．開発の途上で得られた知見は，Zenn 上のブログ記事 [OSS-2] として公開済である．これは以下の複数のパッケージからなる：

- `ghc-wasm-jsobjects`: 型がついていない `JSVal` を `newtype` で包み，JavaScript のオブジェクト階層を模倣することを企図した Haskell ライブラリ．
- `webidl-codegen-wasm`: WebIDL による JavaScript API の記述から，上記の `ghc-wasm-jsobjects` での利用を念頭に置いた GHC の WASM バックエンドの JavaScript FFI 用の型・ブリッジ関数を生成するツール．
- `cloudflare-worker`: Cloudflare Worker のランタイム API へのバインディング．

これらのライブラリを用いて Servant を Workers 上に移植したり，線型型を用いたよりリソース安全な定式化を追求するのが目下の目標である．

Haskell 向け依存型プログラミング用ライブラリ群

Haskell での依存型を用いたプログラミングを支援するためのライブラリ群をいくつか開発し，業務においても利用されている．

- `ghc-typelits-presburger` [OSS-3]: Presburger 算術（整数計画法の理論）の範囲の型レベル等式・不等式制約を自動的に解消する型検査器プラグイン。Singletons 版もあり。
- `type-natural` [OSS-4]: GHC の型レベル自然数に対する singleton と種々の性質の証明を提供するライブラリ。
- `sized` [OSS-5]: ベクトルなどに型レベルで静的に長さをつけて扱うためのライブラリ。

Linear Haskell 向けユーティリティライブラリ

URL: `https://github.com/konn/linear-extra`

Linear Haskell を実用する上で便利な機能のうち、`linear-base` に欠けているものを提供するライブラリ。参照カウンタや任意ベクトルの線型割り当て機能、Linear Constraint 導入までの代替手段である証拠ベースの資源割り当て機能などを提供する。Haskell における線型型については、Zenn 記事 [OSS-6, OSS-7] の執筆を通して日本語圏での普及に務めている。

Haskell Language Server のメンテナー

Haskell の公式言語サーバ（IDE バックエンド）である Haskell Language Server のメンテナの一人を務める．主な貢献事項は以下の通り：

1. Splice Plugin: Template Haskell マクロを評価し，その結果でマクロ呼び出しを置換するためのプラグインを実装した．
2. Disambiguate Imports: 複数モジュールまたは現在のモジュールで定義され衝突が発生している曖昧な識別子を一意化するための機構を提案し，実装した．
3. Eval Plugin の改良: パーザの効率改善，`:type` や:kind コマンドの実装などを行った．

計算機代数システム `computational-algebra`

Web Site: `https://konn.github.io/computational-algebra`

関数型言語 Haskell の処理系 Glasgow Haskell Compiler（以下，GHC）が提供する先進的型システムを活かし，高度な代数計算を安全に行える計算代数システムを EDSL としてゼロから設計・実装した．具体的には，型システムや形式手法を応用することで以下の長所を実現した：

型安全　係数体や変数の数，順番などを型で表すことで，誤操作を防止する．
拡張性　多項式や代数系を表す型クラスを提供し，内部実装に依らず様々なアルゴリズムを適用可能にした．

直感的　$\mathbb{Q}[x,y,z]$ と $\mathbb{Q}[x,z,y,w]$ は型上区別されるが，これらの間の埋め込み写像を型情報だけから自動的に計算出来る．また，多項式も本来の数式に近い形で `#x ^ 2 + #z * #x - 2` のように書ける．

静的保証　QuickCheck を用いて**アルゴリズムの形式仕様を静的に検証**し，ライブラリの品質を保っている．

上述の **Google Summer of Code 2014** では，高速な Gröbner 基底計算アルゴリズムとして知られる F_4 および F_5 アルゴリズムの実装を試み，複数の行列ライブラリを統一的に扱うための枠組みや，ライブラリの整備を進めた．その他に，有理係数多項式の因数分解や，代数的数の計算機能なども実装されている．

特に，型安全性を最大限実現するために，GHC の型レベル自然数機能を強化する `type-natural` パッケージや，GHC の型検査器に Presburger 算術ソルバを組込むコンパイラプラグイン `ghc-typelits-presburger` を開発した．それらを用い，リストや配列，ベクトルなど任意のシーケンシャルなデータ型から固定長のコンテナ型を創り出せる `sized` パッケージを実装した．

本ライブラリを用いた筑波大学数学類の卒業研究をティーチング・アシスタントとして指導した．また，エクアドル大学のヤチャイ技術大学の研究プロジェクト等でも用いられているようである．

その他の開発活動

- Web 上で高速かつ高品質な数式描画を可能とする **KaTeX** に幾つかのコマンド・環境を実装し，コミッタ権限を得た．これまで本格的な JavaScript 開発は経験していなかったが，`npm` や `browserify`，`flow` などを用いたワークフローには半日程度でキャッチアップ出来た．
- Haskell の **Web フレームワーク Yesod** の OAuth 認証ライブラリの開発（現在は引退）．
- 『型システム入門 —プログラミング言語と型の理論—』『Haskell による並列・並行プログラミング』（ともにオーム社）など定評のある技術書・学術書の邦訳に，出版前レビュワーとして参画・貢献した．
- その他の開発プロジェクトは，GitHub 上（`https://github.com/konn`）にて見ることが出来る．

OSS 参考文献

[OSS-1] konn, **Guardian で巨大 Haskell レポジトリの依存関係を正気に保つ**, 2021, URL: `https://zenn.dev/deepflow_tech/articles/secure-haskell-monorepo-deps-with-guardian`.

[OSS-2] ———, **Serverless Haskell - GHC の WASM バックエンドで Haskell を Cloudflare Workers に載せる**, 2021, URL: `https://zenn.dev/konn/articles/2024-06-22-serverless-haskell`.

[OSS-3] ———, **Ghc-typelits-presburger: Presburger Arithmetic solver for GHC type-level natural numbers**, 2015-2024, URL: `https://hackage.haskell.org/package/ghc-typelits-presburger`.

[OSS-4] ———, **Type-natural: type-level natural and proofs of their properties**, 2013-2024, URL: `https://hackage.haskell.org/package/type-natural`.

[OSS-5] ———, **Sized: sized sequence data-types**, 2014-2024, URL: `https://hackage.haskell.org/package/sized`.

[OSS-6] ———, **Haskell は Rust になれるのか？——2023 年の Linear Haskell 体験記**, 2023, URL: `https://zenn.dev/konn/articles/2023-10-01-linear-haskell-in-2023`.

[OSS-7] ———, **2023 年の Linear Haskell で純粋・並列 FFT を実装する——「Haskell は Rust になれるのか？」補遺**, 2023, URL: `https://zenn.dev/konn/articles/2023-12-14-pure-parallel-fft-in-linear-haskell`.