ISSN 2188-286X

マス・フォア・インダストリ研究 No.17

# 代数・論理・幾何と情報科学
## —— 理論から実世界への展開

編集　河村　彰星
津曲　紀宏
西澤　弘毅
溝口　佳寛

Institute of Mathematics for Industry
Kyushu University

About the Mathematics for Industry Research

The Mathematics for Industry Research was founded on the occasion of the certification of the Institute of Mathematics for Industry (IMI), established in April 2011, as a MEXT Joint Usage/Research Center – the Joint Research Center for Advanced and Fundamental Mathematics for Industry – by the Ministry of Education, Culture, Sports, Science and Technology (MEXT) in April 2013. This series publishes mainly proceedings of workshops and conferences on Mathematics for Industry (MfI). Each volume includes surveys and reviews of MfI from new viewpoints as well as up-to-date research studies to support the development of MfI.

October 2018
Osamu Saeki
Director
Institute of Mathematics for Industry

## ALGI (Algebra, Logic and Geometry in Informatics)

Mathematics for Industry Research No.17, Institute of Mathematics for Industry, Kyushu University
ISSN 2188-286X
Editors: Akitoshi Kawamura, Yoshihiro Mizoguchi, Koki Nishizawa, Norihiro Tsumagari
Date of issue: 10 February 2020
Publisher:
Institute of Mathematics for Industry, Kyushu University
Motooka 744, Nishi-ku, Fukuoka, 819-0395, JAPAN
Tel +81-(0)92-802-4402, Fax +81-(0)92-802-4405
URL http://www.imi.kyushu-u.ac.jp/
Printed by
Kijima Printing, Inc.
Shirogane 2-9-6, Chuo-ku, Fukuoka, 810-0012, Japan
TEL +81-(0)92-531-7102 FAX +81-(0)92-524-4411

# 代数・論理・幾何と情報科学
## ——理論から実世界への展開

編集　：　河村　彰星
津曲　紀宏
西澤　弘毅
溝口　佳寛

# 巻頭言

2019 年 12 月 12 日作成

【背景】

本研究集会は、1995 年から定期的に行われてきた「代数・論理・幾何と情報科学研究集会（ALGI）」の第三十回記念集会として企画されたものである。ALGI は、第六回以降、日本数理科学協会（現 国際数理科学協会）の分科会としても開催されてきた。近年、情報科学と深く関連する数学分野として、論理学や離散数学のみならず代数や幾何も注目されている。ALGI は、そのような分野間の活発な議論をする機会として開催されてきた。

【招待講演】

特に第三十回記念集会である今回は、ALGI の扱う各分野から実世界・産業界への応用につながる研究に携わる専門家として、国立情報学研究所の勝股審也氏、産業技術総合研究所の竹内泉氏、株式会社スクウェア・エニックスの長谷川勇氏、理化学研究所の前原貴憲氏、の4名から招待講演をいただいた。

勝股審也氏の講演「物理情報システムへの数学的アプローチ」では、研究総括補佐を務めておられるERATO 蓮尾メタ数理システムデザインプロジェクトは、ソフトウェアシステム開発のための形式手法を物理情報システムの開発に拡張し、安全で高品質な工業製品の作成に理論・応用の両面から貢献することを目指しているとの紹介があった。そして、勝股氏がグループリーダーを務めるメタ理論的統合グループが進めている、物理情報システムの形式化と構築において基礎となる概念の数学的定式化と分析について、具体的に得られた結果が説明された。

竹内泉氏の講演「数学と変数」では、変数の使用は数学の言語に特有のものであり、変数の使用を分析することによって数学の本質に近付く、という主張のもと、数学の中の変数の具体例の比較と分析が行われた。竹内氏の過去の研究で、
恒等式の中の文字、方程式の中の未知数、多項式論の変数と命題変数、座標変数と確率変数、については分析が済んでおり、今回の講演では、独立変数と従属変数についての新しい分析結果が示された。自由変数と束縛変数、統計学の変数については今後の課題とのことであった。

長谷川勇氏の講演「ゲーム開発への代数・論理・幾何と情報科学の適用」では、近年のゲームの大規模化に伴い、ゲーム開発や、それらの品質保証にかかる工数が増大し、ゲーム開発における問題の一つとなっているという背景を踏まえ、開発工数の増大への対策として、ゲーム開発に数学・情報科学を組み合わせた手法を適用することで、これまで機械化が難しく人手に頼っていた工程を自動化し、生産性を向上できる可能性がある、という主張が示された。また、モデル検査によるスクリプトの検証など、実際にゲーム開発に数学・情報科学を適用した事例が紹介された。

前原貴憲氏の講演「代数的・幾何的・論理的制約下の劣モジュラ関数最大化」では、基本的な組合せ最適化問題であり機械学習・データマイニングなど幅広い領域に応用をもつ、劣モジュラ集合関数の近似最大化問題が紹介された。前原氏は近年、この問題を集合よりも複雑な制約領域、たとえば束・複体・論理式で指定されるグラフなどに拡張してきており、本講演では、現在劣モジュラ最大化がどのような制約で解かれているかについて代数・幾何・論理との関係を中心に紹介された。

【一般講演】

10 件の一般講演が行われた。その中には、研究対象に共通点のある複数の講演が見受けられ、それらを同一セッション内で連続させることにより、講演中と質疑応答時に活発な議論が行われる結果となった。

たとえば、クォンテール（完備べき等半環）という代数について、京都大学の藤井宗一郎氏は、豊穣圏やトロピカル数学との関連を講演で紹介した。一方、神奈川大学の安田康史氏と西澤弘毅氏は、前順序や半群との関係性を講演で分析した。これらの講演者同士の活発な議論は、質疑応答の時間終了後も続いていた。

また、不動点演算を持つ論理・計算の体系として、東京大学の塚田武志氏は、不動点算術（Fixed-Point Arithmetic）という体系を紹介し、プログラム検証のための仕様の階層を分析するために有用であることを示した。一方、京都大学の立木秀樹氏は、IFP (Intuitionistic Fixed Point Logic) という体系を紹介し、非構成的対象を扱った証明からのプログラム抽出に有用であることを示した。質疑応答の時間には、両体系の共通点および相違点について、講演者同士の活発な議論が行われた。

【おわりに】

本研究集会の開催が、ALGI の扱う代数・論理・幾何・情報科学の各分野と、物理情報システム、数学の言語的性質、ゲーム開発、機械学習・データマイニング、といった実世界・産業界への応用分野との相互作用を加速させることに貢献したとすれば幸いである。

世話人
河村彰星（九州大学）
津曲紀宏（崇城大学）
西澤弘毅（神奈川大学）
溝口佳寛（九州大学）

## *Table of Contents*

九州大学 IMI 共同利用・研究集会（Ⅱ）

# 代数・論理・幾何と情報科学――理論から実世界への展開

日時：　2019年8月31日（土）9:45 ～ 17:30
　　2019年9月1日（日）10:00 ～ 17:15
場所：　九州大学 伊都キャンパス ウエスト1号館 D棟4階
　　IMIコンファレンスルーム（W1-D-414）
ホームページ：　https://sites.google.com/site/algimeeting/home/30/program

## 8月31日（土）

9:45-10:00　オープニング＋連絡など

10:00-12:00

１．招待講演者：勝股 審也（国立情報学研究所 ERATO 蓮尾メタ数理システムデザインプロジェクト）
　題目：物理情報システムへの数学的アプローチ

２．講演者：Devi Rahmah Sope（近畿大学）
　題目：On a Fuzzification and Comparison of Clustering Indices

３．講演者：冨田 悠（京都大学 数理解析研究所）
　題目：Constructing non-symmetric closed categories from planar combinatory algebra

12:00-14:00　昼食

14:00-15:30

１．招待講演者：竹内 泉(産業技術総合研究所)
　題目：数学と変数

２．講演者：本浦 庄太（日本電気株式会社）
　題目：On Logic for Conditional Probabilities of Propositional Formulae

15:30-16:00　休憩

16:00-17:30

１．講演者：藤井宗一郎（京都大学 数理解析研究所）
　　題目：Enriched categories and tropical mathematics

２．講演者：安田 康史（神奈川大学）
　　題目：Relationship among orders, semigroups, and quantales

３．講演者：西澤 弘毅（神奈川大学）
　　題目：Correspondences among classes of weak preorders,
　　partial semigroups, and quantales

19:10-　意見交換会

**9月1日（日）**

10:00-12:00

１．招待講演者：長谷川 勇（スクウェア・エニックス）
　　題目：ゲーム開発への代数・論理・幾何と情報科学の適用

２．講演者：塚田 武志（東京大学）
　　題目：不動点算術とプログラム検証

３．講演者：立木 秀樹（京都大学 人間・環境学研究科）
　　題目：非構成的対象を扱った証明からのプログラム抽出

12:00-14:00　昼食

14:00-15:30

１．招待講演者：前原 貴憲（理化学研究所）
　　題目：代数的・幾何的・論理的制約下の劣モジュラ関数最大化

２．講演者：田中康平（信州大学 経法学部）
　　題目：Topological and combinatorial methods in symmetric
　　motion planning

15:30-16:00　休憩

16:00-17:00

１．講演者：山形頼之（産業技術総合研究所）
　　題目：Recent progress of consistency proofs of equational systems inside
　　bounded arithmetics

17:00-17:15　クロージング+連絡など

# 物理情報システムを数学的に捉える

勝股 審也

JST ERATO 蓮尾プロジェクト
国立情報学研究所

2019/08/31

## ERATO蓮尾MMSDプロジェクト

JST ERATO プロジェクト (2016年10月〜)

ERATO MMSD

NII 国立情報学研究所 National Institute of Informatics
JST 国立研究開発法人 科学技術振興機構
大阪大学 OSAKA UNIVERSITY
京都大学 KYOTO UNIVERSITY
九州大学
UNIVERSITY OF WATERLOO

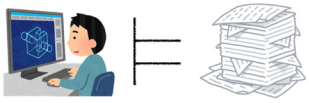
## ERATO蓮尾MMSDプロジェクト

JST ERATO プロジェクト (2016年10月〜)


## 形式手法とは

システム開発を数学・論理学で形式化し、システムの品質と信頼性を向上する技術の総称

(2) 実装　(3) 検証　(1) 仕様



**利点**

- 仕様の曖昧さ・検証の不完全さの除去
- 仕様・システムの不具合の早期発見・自動発見

## 形式手法とは

システム開発を数学・論理学で形式化し、システムの品質と信頼性を向上する技術の総称

- Sony - Felica ファームウェア
- Microsoft - Windows デバイスドライバ
- RATP - パリ地下鉄 14 号線
- Airbus - 航空機システム
- 富士重工業 - ECU ソフトウェア開発
- JAXA - 人工衛星の姿勢制御ソフトウェア

厳密な仕様記述における形式手法成功事例調査報告書 (情報処理推進機構) および
http://formal.mri.co.jp/db/より抜粋

## 形式手法とは

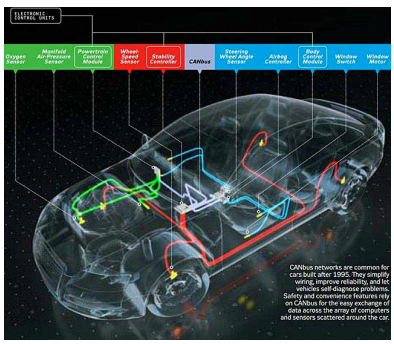
システム開発を数学・論理学で形式化し、システムの品質と信頼性を向上する技術の総称

形式手法を支える理論

- システム検証の論理
- 形式言語理論
- プログラミング言語理論
- 計算量理論, 暗号理論, 他多数

「情報・情報処理・システム」の数学的扱いにより発展

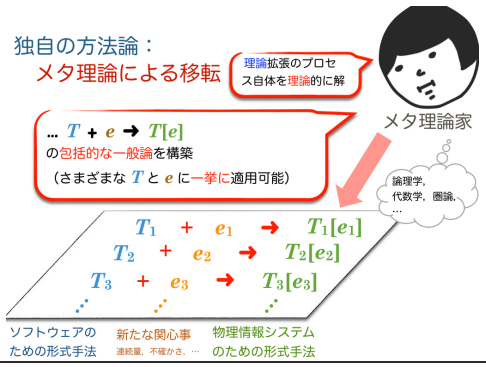
## 物理情報システム




http://www.popularmechanics.com/cars/how-to/a7386/how-it-works-the-computer-inside-your-car/

## 本プロジェクトのゴールと特色

## 本プロジェクトのゴールと特色




## 本プロジェクトの理論的基礎

G0 メタ理論的統合グループ
G1 ヘテロジニアス形式手法グループ
G2 産業応用グループ (Waterloo 大学)
G3 インテリジェンス協働形式手法グループ

**論理学** 様相論理・時相論理・微分動的論理・ホーア論理・ロバストネス意味論・超準解析・型理論・証明支援系

**確率論** 確率的プログラミング言語・グラフィカルモデルの基礎・確率的プログラムの推論・テストケースの効率的な生成・マルチンゲールによる到達可能性解析

**圏論** 状態遷移系の圏論的モデル・上記論理およびプログラミング言語の意味論・束論

## 本プロジェクトの理論的基礎

G0 メタ理論的統合グループ
G1 ヘテロジニアス形式手法グループ
G2 産業応用グループ (Waterloo 大学)
G3 インテリジェンス協働形式手法グループ

**論理学** 様相論理・時相論理・微分動的論理・ホーア論理・ロバストネス意味論・超準解析・型理論・証明支援系

**確率論** 確率的プログラミング言語・グラフィカルモデルの基礎・確率的プログラムの推論・テストケースの効率的な生成・マルチンゲールによる到達可能性解析

**圏論** 状態遷移系の圏論的モデル・上記論理およびプログラミング言語の意味論・束論

## Group 0

圏論・論理学を通じた「抽象化」の担当

### 状態遷移系の数理モデル

1. 余代数的模倣関係のファイバー圏による一般化
2. 圏論的双模倣ゲーム、
3. 形式手法 Event-B の圏論化

### ニューラルネットワークの圏論的モデル

1. リカレントニューラルネットワークの微分

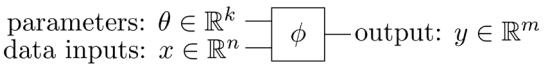
他にも、開写像的模倣の特徴づけ、名前付き様相 $\mu$ 計算の決定可能性、バックプロパゲーションの圏論的扱い等を研究をしています。

Part I

# Differentiable Causal Computation via Delayed Trace

[David Sprunger, K, LICS'19]

## ニューラルネットワーク

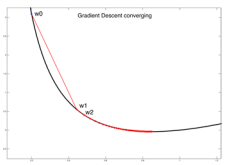



学習: 以下の関数 $e : \mathbb{R}^k \to \mathbb{R}$ を最小にする $\theta$ を見つける

$$e(\theta) \triangleq \sum_{(i,o)\in L} err(\phi(\theta, i), o)$$

$L$ は学習データ

## 勾配降下法




http://www.cs.cornell.edu/courses/cs4780/2015fa/web/lecturenotes/lecturenote07.html

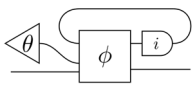
$\theta$ を以下で更新する

$$\theta' = \theta - \alpha \times \nabla e(\theta)$$

勾配 $\nabla e$ を取るには微分が必要

$$\nabla e(p) = (Je(p))^t(1)$$

## Recurrent Neaural Network

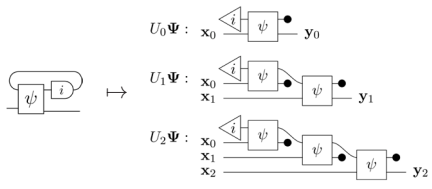



状態関数と出力関数が NN で記述された Mealy 機械

- 入力は時系列データ $x_0, x_1, x_2, \cdots \in \mathbb{R}^i$
- 出力は時系列データ $y_0, y_1, y_2, \cdots \in \mathbb{R}^o$
- $i$ は初期状態
- $\theta \in \mathbb{R}^k$ はパラメータ

RNN の微分とは何か?

## Backpropagation through Time




**問い**

1. 展開して微分することの正当性は?
2. BPTT はなんらかの意味での微分になっているか?

## 本研究の貢献

**問い**

1. 展開して微分することの正当性は?
2. BPTT はなんらかの意味での微分になっているか?

**我々の貢献**

- 遅延フィードバックを取れる計算のモデルを構成
- 遅延フィードバックのある計算の微分 $D^*$ を定義
- $D^*$ が「展開して微分」に一致

## 本研究の貢献

**問い**

1. 展開して微分することの正当性は?
2. BPTT はなんらかの意味での微分になっているか?

**我々の貢献**

- $St$ 構成と遅延トレース
- $\mathbb{C}$ が微分圏ならば $St(\mathbb{C})$ も微分圏
- $D^*$ が「展開して微分」に一致

## カルテシアン微分圏

...[Blute, Cockett, Seely'09] はカルテシアン圏 $\mathbb{C}$ で

1. 各対象 $X$ に可換モノイド構造が備わっており

$$1 \xrightarrow{e_X} X \xleftarrow{m_X} X \times X$$

2. 射の微分 $D : \mathbb{C}(X, Y) \to \mathbb{C}(X \times X, Y)$ が取れるもの
   - $D$ は第一引数に関して線形
   - 連鎖律が成立
   - 二階微分の性質

$Df$ は直感的には

$$Df(\delta, p) \cdots Jf(p) \times \delta$$

## カルテシアン微分圏

**カルテシアン微分圏 $Euc^\infty$**

対象 $0, 1, 2, \cdots \in \mathbb{N}$

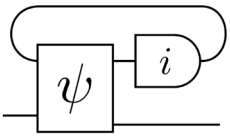
射 $n, m \in \mathbb{N}$ に対し

$$Euc^\infty(n, m) = \{f : \mathbb{R}^n \to \mathbb{R}^m \mid f_1, \cdots, f_m \in C^\infty\}$$

微分 $Df(\delta, p) = Jf(p) \times \delta$

射を多項式関数に制限したものも例となる

## 遅延トレース




**問い**

遅延フィードバックをどう数学的に実現するか?

**答え**

カルテシアン圏 $\mathbb{C}$ に対して遅延トレースを導入する $St$ 構成を与えた

## $St$ 構成

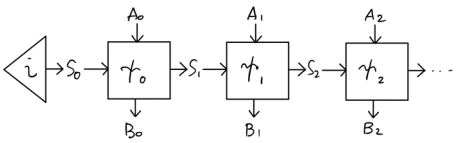
カルテシアン圏 $\mathbb{C}$ に対し、$St(\mathbb{C})$ を以下で定義

対象 $(A_0, A_1, \cdots) \in |\mathbb{C}|^\infty$

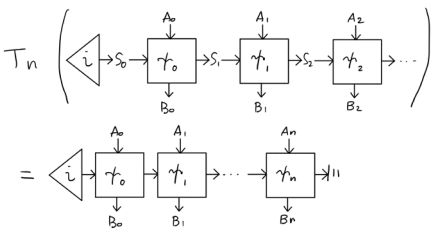
射 $i : 1 \to S_0$

$\psi_n : S_n \times A_n \to S_{n+1} \times B_n \quad (n \in \mathbb{N})$

を外延同値性 (次のスライド) で割ったもの



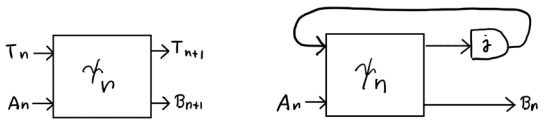
## 射の打ち切り



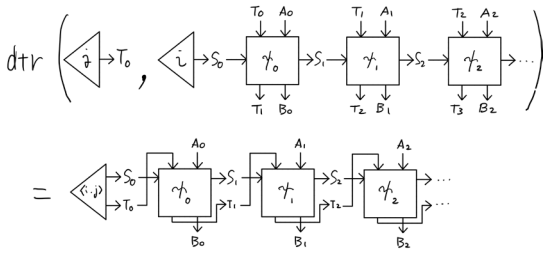

**外延同値性**

$$(i, \psi) \sim (j, \phi) \overset{def}{\iff} \forall n \,.\, T_n(i, \psi) = T_n(j, \phi)$$

## 遅延トレース

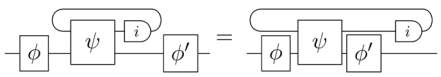


## 遅延トレース



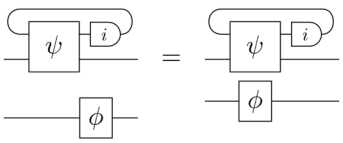
## 遅延トレースの性質

Naturality



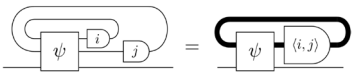
## 遅延トレースの性質

Superposing



## 遅延トレースの性質

Vanishing



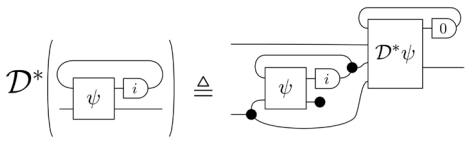

## 主結果

**定理**

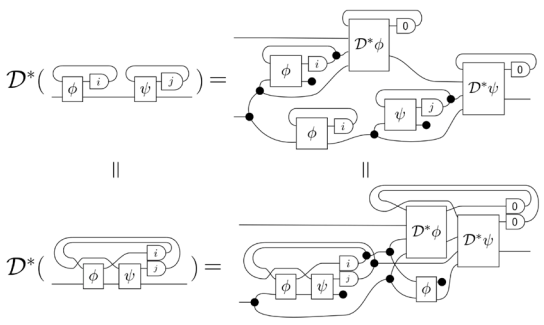
$\mathbb{C}$ がカルテシアン微分圏なら $St(\mathbb{C})$ に微分演算 $D^*$ が入り、$St(\mathbb{C})$ はカルテシアン微分圏となる。

**定理 (BPTT is Differential)**

$T_k(D^*(\phi)) = D(T_k(\phi))$




## RNN の微分の連鎖律




## まとめ

- RNN の微分をとるという操作を圏論的に分析した
- RNN の特徴である遅延フィードバックをモデルするため、$St$ 構成を導入した
- $\mathbb{C}$ がカルテシアン微分圏の時、$St(\mathbb{C})$ をカルテシアン微分圏にする方法を与え、これが BPTT で行われる「展開して微分」と一致することを示した。

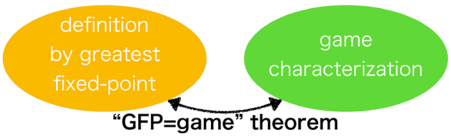
Part II

# Codensity Games for Bisimilarity

[Yuichi Komorida, K, Nick Hu, Bartek Klin, Ichiro Hasuo, LICS'19]

## 背景

双模倣性とそれを特徴づけるゲーム



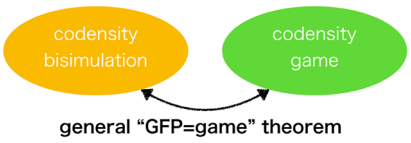

http://group-mmm.org/ y-komorida/slides/20190626LICS.pdf

- LTS の双模倣性 [Park 1981][Milner 1989]
- Markov 鎖の双模倣性 [Larsen & Skou 1991][Fijalkow+ ICALP2017]
- Markov 鎖の双模倣距離 [Desharnais+ 2004][Desharnais+ 2008]

## 本研究の目的

**本研究の目的**

余稠密双模倣性 [Sprunger+'18] を特徴づける余稠密ゲームをファイバー圏という一般的なセッティングで与えた




http://group-mmm.org/ y-komorida/slides/20190626LICS.pdf

## $t : S \to \mathcal{P}S$ 上の余稠密ゲーム

**双模倣同値関係のための余稠密ゲーム**

D(uplicator) 手 $X \in |\mathbf{EqRel}_S|$ → S(poiler) 手 $k : S \to 2$

D が永遠に指せるか、S が指せなくなれば D の勝ち

| 現在の手 | 相手の手 | 条件 |
|---|---|---|
| $X \in \lvert\mathbf{EqRel}_S\rvert$ | $k$ | $\exists (x,y) \in X \,.\, k^{\#}(x) \neq k^{\#}(y)$ |
| $k : S \to 2$ | $X'$ | $\exists (x,y) \in X' \,.\, k(x) \neq k(y)$ |

**定理: 任意の同値関係 $X$ について以下は同値**

- $X$ は双模倣性同値関係に含まれる
- $X$ から初めて D が勝てる

## [Hermida-Jacobs'98] の双模倣の理論

$$\dot{F} \circlearrowright \mathbb{E} \xrightarrow{p} \mathbb{C} \circlearrowright F \qquad p \circ \dot{F} = F \circ p$$

$p$ は **CLat**$_\wedge$-fibration とする

### Hermida-Jacobs'98 の理論の骨子

- 状態遷移系
  $\iff$ $F$-余代数
- 状態遷移系上の双模倣
  $\iff$ $F$-余代数の上にある $\dot{F}$-余代数
- 状態遷移系上の双模倣性 (bisimilarity)
  $\iff$ $F$-余代数の上にある最大の $\dot{F}$-余代数

$\dot{F}$ のとり方で様々な双模倣を導くことができる

## 余稠密持ち上げ [Sprunger,K+'18]

... は **CLat**$_\wedge$-fibration に沿って関手を持ち上げる方法

$$F^{\tau,O} \circlearrowright \mathbb{E} \xrightarrow{p} \mathbb{C} \circlearrowright F$$

パラメータ $\tau : F\Omega \to \Omega$ と $O \in \mathbb{E}_\Omega$ の元、以下で定義

$$F^{\tau,O}X = \bigwedge_{k \in \mathbb{E}(X,O)} (\tau \circ F(pk))^* O$$

$F$-余代数の上の $F^{\tau,O}$-余代数を余稠密双模倣と呼ぶ

### 定理: 以下は同値

- $X$ は $(\tau, O)$-余稠密双模倣性の下界
- $(\tau, O)$-余稠密ゲームで $D$ は $X$ から始めて勝てる

## $F : \mathbb{C} \to \mathbb{C}$ の余代数

... は対象と射の組 $(S \in \mathbb{C}, \quad t : S \to FS)$

### 余代数の例 ($\mathbb{C} =$ **Set**)

| $FX$ | $\mathcal{P}X$ | $\mathcal{D}X$ | $I \Rightarrow (X \times O)$ |
|---|---|---|---|
| $t :$ | $S \to \mathcal{P}S$ | $S \to \mathcal{D}S$ | $S \to I \Rightarrow (S \times O)$ |
| 遷移系 | Kripke フレーム | Markov 鎖 | Mealy 機械 |

$$\mathcal{P}X = \{U \subseteq X\},$$
$$\mathcal{D}X = \{\mu : X \to [0,1] \mid \mu \text{は有限確率分布}\}$$

## 関手の持ち上げ

$F : \mathbb{C} \to \mathbb{C}$ の $p : \mathbb{E} \to \mathbb{C}$ に沿った持ち上げ:

$$\dot{F} \circlearrowright \mathbb{E} \xrightarrow{p} \mathbb{C} \circlearrowright F \qquad p \circ \dot{F} = F \circ p$$

### 双模倣関係のための持ち上げ

... は $\mathcal{P} :$ **Set** $\to$ **Set** の **EqRel** $\to$ **Set** に沿った持ち上げ

$$\dot{\mathcal{P}}(X,R) = \{(U,V) \mid (\forall u \in U\,.\,\exists v \in V\,.\,(u,v) \in R) \wedge (\forall v \in V\,.\,\exists u \in U\,.\,(u,v) \in R)\}$$

## 関手の持ち上げ

$F : \mathbb{C} \to \mathbb{C}$ の $p : \mathbb{E} \to \mathbb{C}$ に沿った持ち上げ:

$$\dot{F} \circlearrowright \mathbb{E} \xrightarrow{p} \mathbb{C} \circlearrowright F \qquad p \circ \dot{F} = F \circ p$$

### Kantorovich 距離

... は $\mathcal{D} :$ **Set** $\to$ **Set** の **EPMet** $\to$ **Set** に沿った持ち上げ

$$\dot{\mathcal{D}}(X,d) = (DX, d_K), \quad d_K(\mu,\nu) \triangleq \sup_k |E_\mu[k] - E_\nu[k]|$$

ここで $k$ は **EPMet**$(X,[0,1])$ を走る

## 持ち上げによる双模倣の定式化

$F : \mathbb{C} \to \mathbb{C}$ の $p : \mathbb{E} \to \mathbb{C}$ に沿った持ち上げを仮定する

$$\dot{F} \circlearrowright \mathbb{E} \xrightarrow{p} \mathbb{C} \circlearrowright F \qquad p \circ \dot{F} = F \circ p$$

### 定義

$F$-余代数 $(S,t)$ 上の $\dot{F}$-双模倣とは $\dot{F}$-余代数 $(\dot{S},\dot{t})$ で以下を満たすもの

$$p\dot{S} = S, \quad p\dot{t} = t$$

### Lemma

$p$ が忠実な時、$(\dot{S},\dot{t})$ と $(\dot{S},\dot{u})$ が $(S,t)$ 上の $\dot{F}$-双模倣ならば $\dot{t} = \dot{u}$

## 余稠密持ち上げ [Sprunger,K'18]

$F : \mathbb{C} \to \mathbb{C}$ の **CLat**$_\wedge$-fibration $p : \mathbb{E} \to \mathbb{C}$ に沿った持ち上げを構成する方法

$$\dot{F}^{\tau,O} \circlearrowright \mathbb{E} \xrightarrow{p} \mathbb{C} \circlearrowright F \qquad p \circ F^{\tau,O} = F \circ p$$

起源は $\top\top$-lifting(Lindley&Stark) に遡る

### **CLat**$_\wedge$-fibration

... とは $\mathbb{C}^{op} \to$ **CLat**$_\wedge$ という関手と同等な fibration

パラメータ $\tau : F\Omega \to \Omega$ と $O \in \mathbb{E}_\Omega$ の元、以下で定義

$$F^{\tau,O}X = \bigwedge_{k \in \mathbb{E}(X,O)} (\tau \circ F(pk))^* O$$

## 余稠密持ち上げの例

### $\mathcal{D} :$ **Set** $\to$ **Set** を **EPMet** $\to$ **Set** に沿って

$$\Omega = [0,1], \quad \tau = E \quad \text{(期待値)}, \quad O = |\cdot - \cdot|$$

による余稠密持ち上げは Kantorovich 距離と一致

$$\mathcal{D}^{\tau,O}(X,d) = (DX, d_K), \quad d_K(\mu,\nu) \triangleq \sup_k |E_\mu[k] - E_\nu[k]|$$

ここで $k$ は **EPMet**$(X,[0,1])$ を走る

$$F^{\tau,O}X = \bigwedge_{k \in \mathbb{E}(X,O)} (\tau \circ F(pk))^* O$$

## 余稠密持ち上げの例

### $\mathcal{P} : \mathbf{Set} \to \mathbf{Set}$ を $\mathbf{EqRel} \to \mathbf{Set}$ に沿って

$$\Omega = 2, \quad \tau = \vee, \quad O = (=_2)$$

による余稠密持ち上げは一見何かわからない:

$$\mathcal{P}^{\tau,O}(X,R) = (\mathcal{P}X, B_R)$$

$$B_R = \left\{ (U,V) \mid \begin{array}{l} \forall k \,.\, (\forall (x,y) \in R \,.\, k(x) = k(y)) \\ \Longrightarrow\ k^{\#}(U) = k^{\#}(V) \end{array} \right\}$$

### 命題

$\mathcal{P}^{\tau,O}$ は双模倣関係のための $\mathcal{P}$ の持ち上げと一致

## 余稠密双模倣

- $F : \mathbb{C} \to \mathbb{C}$ と $\mathbf{CLat}_\wedge$-fibration $p : \mathbb{E} \to \mathbb{C}$
- パラメータ $\Omega \in \mathbb{C}, \tau : F\Omega \to \Omega, O \in \mathbb{E}_\Omega$

### 定義

$F$-余代数 $(S,t)$ の上の $F^{\tau,O}$-双模倣 $\dot{S}$ を $(\tau,O)$-余稠密双模倣と呼ぶ

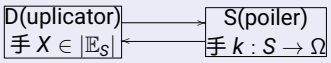
### 命題: 任意の $F$-余代数 $(S,t)$ に対し以下は同値

- $X$ は $(\tau,O)$-余稠密双模倣
- $X \le t^*(F^{\tau,O}X)$
- $\forall X \le k^*O \,.\, X \le (\tau \circ F(pk) \circ t)^*O$

## $(\tau,O)$-余稠密ゲーム

- $F : \mathbb{C} \to \mathbb{C}$ と $\mathbf{CLat}_\wedge$-fibration $p : \mathbb{E} \to \mathbb{C}$
- パラメータ $\Omega \in \mathbb{C}, \tau : F\Omega \to \Omega, O \in \mathbb{E}_\Omega$

### $(\tau,O)$-余稠密ゲーム




D が永遠に指せるか、S が指せなくなれば D の勝ち

| 現在の手 | 相手の手 | 条件 |
|---|---|---|
| $X \in \lvert\mathbb{E}_S\rvert$ | $k : S \to \Omega$ | $X \not\le (\tau \circ F(pk) \circ t)^*O$ |
| $k : S \to \Omega$ | $X' \in \lvert\mathbb{E}_S\rvert$ | $X' \not\le k^*O$ |

## 余稠密ゲームと余稠密双模倣性

### 双模倣同値関係のための余稠密ゲーム

- $F = \mathcal{P}, p : \mathbf{EqRel} \to \mathbf{Set}$
- $\Omega = 2, \tau = \vee, O = (=)_2$

| 現在の手 | 相手の手 | 条件 |
|---|---|---|
| $X \in \lvert\mathbf{EqRel}_S\rvert$ | $k$ | $\exists (x,y) \in X \,.\, k^{\#}(x) \neq k^{\#}(y)$ |
| $k : S \to 2$ | $X'$ | $\exists (x,y) \in X' \,.\, k(x) \neq k(y)$ |

### 定理: 任意の $X \in \mathbb{E}_S$ に対し次は同値

- $X$ は $(\tau,O)$-余稠密双模倣性の下界
- $(\tau,O)$-余稠密ゲームで $D$ は $X$ から始めて勝てる

## 余稠密ゲームのTrimming

| 現在の手 | 相手の手 | 条件 |
|---|---|---|
| $X \in \lvert\mathbb{E}_S\rvert$ | $k : S \to \Omega$ | $X \not\leq (\tau \circ F(pk) \circ t)^* O$ |
| $k : S \to \Omega$ | $X' \in \lvert\mathbb{E}_S\rvert$ | $X' \not\leq k^* O$ |

$\mathbb{E}_S$ の対象すべてが $D$ の指し手というのは大きすぎる

**命題: Trimming**

ファイバー $\mathbb{E}_S$ がある部分集合 $G \subseteq \mathbb{E}_S$ で生成されるならば、$D$ の手を $G$ に制限できる

例えば $\mathbf{EqRel}_S$ は以下から生成される

$$\{Eq_{(x,y)} \mid x, y \in S\}$$

## まとめ

- 余稠密双模倣性の下界を勝利位置とする余稠密ゲームを与えた
- ファイバー圏の言葉で一般化されており、幅広い状態遷移系や双模倣の概念をカバーできる

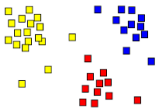
**ALGI 2019**

# On a Fuzzification and Comparison of Clustering Indices

Devi Rahmah Sope , Mitsuhiko Fujio
Kindai University

## Introduction

- Clustering is the grouping of a particular set of objects based on their characteristics, aggregating them according to their similarities.
- The group of similar data points is called a **Cluster.**



## Fuzzy C-Mean Clustering

- **Fuzzy clustering** is a form of clustering in which each data point can belong to more than one cluster
- Object function

$$J = \sum_{k=1}^{C} \sum_{i=1}^{N} \mu_{ik}^{m} ||\boldsymbol{o}_i - \boldsymbol{z}^{k}||^2$$

under the constrain

$$\sum_k \mu_{ik} = 1$$

Algorithm :

- Initialize fuzzy partition randomly $\mu_{ik}$

1. Calculate center $\boldsymbol{z}^{k} = \Gamma^{[k]}[\boldsymbol{o}]$
2. Calculate distance $d_{ik} = ||\boldsymbol{o}_i - \boldsymbol{z}^{k}||$
3. Update fuzzy partition $\mu_{ik} = \frac{1}{\left(\sum_k \frac{d_i}{d_k}\right)^{2/(m-1)}}$
4. Repeat step 1-3 until J became smaller

## Particle Swarm Optimization

Birds Swarm

Fish Swarm

## Particle Swarm Optimization

$$X_{normalize} = \begin{pmatrix} \mu_{11}/\sum_k \mu_{1k} & \cdots & \mu_{1K}/\sum_k \mu_{1k} \\ \vdots & \ddots & \vdots \\ \mu_{n1}/\sum_k \mu_{nk} & \cdots & \mu_{nK}/\sum_k \mu_{nk} \end{pmatrix}$$

- Population based stochastic optimization technique
- Fitness Function

  $C/J$ ($C$ is contant value)

  under the constrain

  $\sum_k \mu_{ik} = 1$

**Algorithm**

- Generate randomly Position of particle is $X(X_1 \dots X_p)$ where $X = (\mu_{ik})$ and velocity $V(V_1 \dots V_p)$

1. Calculate center $z_q^{[k]}$ of each individual $X_q$ and calculate $C/J$
2. Find the best position $pbest_q$ in the history of each particle $X_q$ and the best position $gbest$ in the history of the whole population
3. Update velocity of particle

   $V_q(t+1) = w \cdot Vq(t) + c_1 r_1 (pbest_q - Xq(t)) + c_2 r_2 (gbest_q - Xq(t))$
4. Update position of particle

   $X_q(t+1) = X_q(t) + V_q(t+1)$
5. Repeat step 1 – 4 until $C/J$ become smaller

## Research Purpose

Clustering indices evaluate the quality of cluster

Bernard Desgraupes treated 27 of Crisp Clustering Indices.
Only XBI for fuzzy clustering


To fuzzify the crisp clustering indices

Minimize these clustering indices for optimization problem

## Bernard Desgraupes' Indices

"clusterCrit: Clustering Indices", R package version 1.2.3. http://CRAN.Rproject.org/package=clusterCrit, 2013

| | | | |
|---|---|---|---|
| The Ball-Hall index | The Baker-Hubert Gamma index | The PBM index | The Silhouette index |
| The Banfeld-Raftery index | The GDI index | The Point-Biserial index | The Tau index |
| The C index | The G plus index | The Ratkowsky-Lance index | The Trace W index |
| The Calinski-Harabasz index | The Ksq DetW index | The Ray-Turi index | The Trace WiB index |
| The Davies-Bouldin index | The Log Det Ratio index | The Scott-Symons index | The Wemmert-Gançarski index |
| The Det Ratio index | The Log SS Ratio index | The SD index | The Xie-Beni index |
| The Dunn index | The McClain-Rao index | The S Dbw index | |

## Bernard Desgraupes' Indices

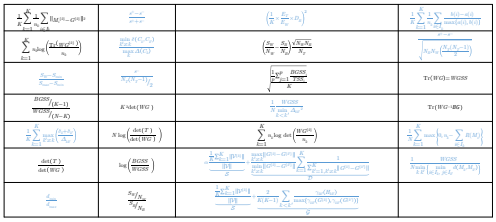
"clusterCrit: Clustering Indices", R package version 1.2.3. http://CRAN.Rproject.org/package=clusterCrit, 2013

| | | | |
|---|---|---|---|
| The Ball-Hall index | The Baker-Hubert Gamma index | The PBM index | The Silhouette index |
| The Banfeld-Raftery index | The GDI index | The Point-Biserial index | The Tau index |
| The C index | The G plus index | The Ratkowsky-Lance index | The Trace W index |
| The Calinski-Harabasz index | The Ksq DetW index | The Ray-Turi index | The Trace WiB index |
| The Davies-Bouldin index | The Log Det Ratio index | The Scott-Symons index | The Wemmert-Gançarski index |
| The Det Ratio index | The Log SS Ratio index | The SD index | The Xie-Beni index |
| The Dunn index | The McClain-Rao index | The S Dbw index | |

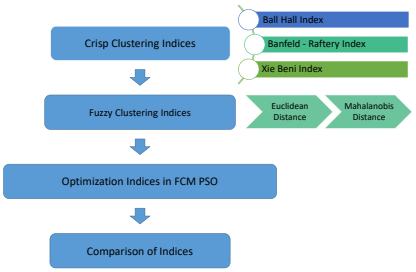
14/27 ⇒ non analytic

## Bernard Desgraupes' Indices

"clusterCrit: Clustering Indices", R package version 1.2.3. http://CRAN.Rproject.org/package=clusterCrit, 2013



## Framework




## Crisp vs. Fuzzy

**K-means**

- Clustering : Disjoint Union

$$\mu_{ik} = 0 \; or \; 1$$

$$\sum_k \mu_{ik} = 1$$

$$0 < \sum_i \mu_{ik} < n$$

**FCM**

- Clustering : Membership degree

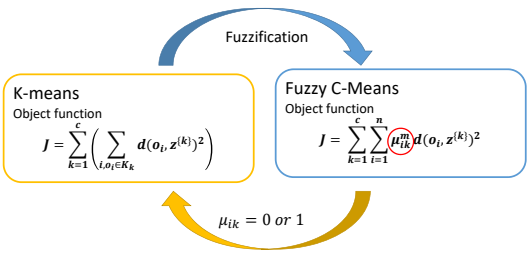
$$0 \leq \mu_{ik} \leq 1$$

$$\sum_k \mu_{ik} = 1$$

$$0 < \sum_i \mu_{ik} < n$$

"Crisp" is a special case of "Fuzzy" through the membership degree

## Crips is a Special Case of Fuzzy



## New Probabilities

- $\mu_{ik}$ : membership degree of the $i$-th object $\boldsymbol{o}_i$ to the $k$-th cluster $K_k$.

$$\Downarrow$$

- $\mu_{ik}^m$ : ratio of the $i$-th object $\boldsymbol{o}_i$ to the $k$-th cluster $K_k$.

- Clusterwize probability $p_i^{\{k\}} = \frac{\mu_{ik}^m}{\sum_{j=1}^n \mu_{jk}^m} \ (1 \le i \le n, 1 \le k \le c)$ $\boxed{\frac{\mu_{ik}}{|I_k|}}$

- Total probability $p_i = \frac{\sum_{\hat{c}=1}^c \mu_{i\hat{c}}^m}{\sum_{\hat{c}=1}^c \sum_{j=1}^n \mu_{j\hat{c}}^m} \ (1 \le i \le n)$ $\boxed{\frac{1}{n}}$

## Statistics

**Total**

- Expectation
$$\mathrm{E}[f(\boldsymbol{o})] = \sum_{i=1}^n p_i f(\boldsymbol{o}_i)$$
- Barycenter
$$\boldsymbol{z} = \mathrm{E}[\boldsymbol{o}]$$
- Variance-covariance
$$\Sigma = \mathrm{E}\left[ {}^t(\boldsymbol{o}-\boldsymbol{z})(\boldsymbol{o}-\boldsymbol{z}) \right]$$

**Clusterwize**

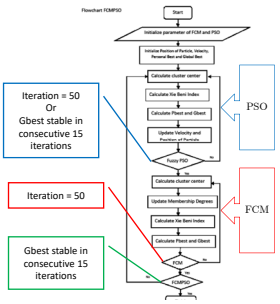
- Expectation
$$\mathrm{E}^{\{k\}}[f(\boldsymbol{o})] = \sum_{i=1}^n p_i^{\{k\}} f(\boldsymbol{o}_i)$$
- Barycenter
$$\boldsymbol{z}^{\{k\}} = \mathrm{E}^{\{k\}}[\boldsymbol{o}]$$
- Variance-covariance
$$\Sigma^{\{k\}} = \mathrm{E}^{\{k\}}\left[ {}^t\left(\boldsymbol{o}-\boldsymbol{z}^{\{k\}}\right)\left(\boldsymbol{o}-\boldsymbol{z}^{\{k\}}\right) \right]$$

## Fuzzified Indices

| Index | Crisp (Desgraupes') | Fuzzy |
|---|---|---|
| $J$ | | $\sum_{k=1}^c \sum_{i=1}^n \mu_{ik}{}^m \lvert\lvert \boldsymbol{o}_i - \boldsymbol{z}^{\{k\}} \rvert\rvert^2$ |
| BHI | $\frac{1}{K}\sum_{k=1}^K \frac{1}{n_k}\sum_{i\in I_k} \lvert\lvert \boldsymbol{o}_i - \boldsymbol{z}^{\{k\}} \rvert\rvert^2$ | $\frac{1}{c}\sum_{k=1}^c \mathrm{E}^{\{k\}}\left[\lVert \boldsymbol{o}-\boldsymbol{z}^{\{k\}} \rVert^2\right]$ |
| $\mathrm{BRI_E}$ | $\sum_{k=1}^K n_k \log\left(\sum_{j=1}^m \mathrm{Var}\left(V_j^{\{k\}}\right)\right)$ | $\sum_{k=1}^c \nu^{\{k\}} \log \mathrm{E}^{\{k\}}\left[\lVert \boldsymbol{o}-\boldsymbol{z}^{\{k\}} \rVert^2\right]$ |
| $\mathrm{BRI_M}$ | | $\sum_{k=1}^c \nu^{\{k\}} \log \mathrm{E}^{\{k\}}\left[ (o-z^{\{k\}})\left(\Sigma^{\{k\}}\right)^{-1} {}^t(o-z^{\{k\}}) \right] = \nu \log d$ |
| XBI | $\frac{\frac{1}{N}\sum_{k=1}^K n^{\{k\}} \sum_{j=1}^p \mathrm{Var}\left(V_j^{\{k\}}\right)}{\min_{k,k'} \left\{ \min_{i\in I_{k'}, j\in I_{k'}} \lvert\lvert \boldsymbol{o}_i - \boldsymbol{o}_j \rvert\rvert^2 \right\}}$ | $\frac{\sum_{k=1}^c \sum_{i=1}^n \mu_{ik}{}^m \lvert\lvert \boldsymbol{o}_i - \boldsymbol{z}^{\{k\}} \rvert\rvert^2}{\nu \min_{k,k'} \lVert \boldsymbol{z}^{\{k\}} - \boldsymbol{z}^{\{\ell\}} \rVert^2}$ |

## Experiment

- Optimization for Indices in FCM PSO
  - Generate matrix $\mu$ for each index
  - Repeat until 10 times
  - Find the minimum value of indices (for normalization)



Algorithm for XBI Index

1. Generate 10 times of matrix $\mu$
2. Find the minimum XBI between matrices
3. Repeat until terminating
   condition(object function no improve until 15 times, $\varepsilon = 1e-2$)
4. Calculate the value for indices of J, BHI, BRI

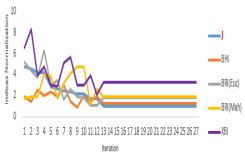
## Experiment Result

**Table1**. Normalization of each indeces

| Function | It | Time (Sec) | J | BHI(Euc) | BRI (Euc) | BRI(Mah) | XBI (Euc) |
|---|---|---|---|---|---|---|---|
| J | 27 | 3.46936 | 1 | 1.204 | 1.603 | 1.853 | 3.263 |
| BHI (Euc) | 31 | 3.52217 | 2.853 | 1 | 1.730 | 1.781 | 2.585 |
| BRI (Euc) | 36 | 3.70953 | 2.468 | 1.645 | 1 | 1.546 | 2.025 |
| BRI (Mah) | 33 | 3.58846 | 2.797 | 1.864 | 1.651 | 1 | 2.025 |
| XBI | 37 | 3.68129 | 3.364 | 1.196 | 2.396 | 2.303 | 1 |

## Indices normalization on J as Object Function

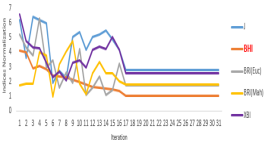
Behavior Indices



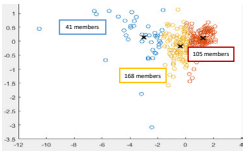

Cluster Form


## Indices normalization on BHI as Object Function

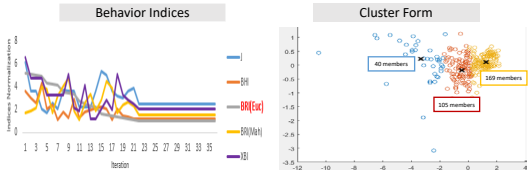
Behavior Indices



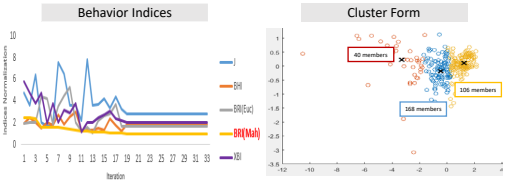

Cluster Form



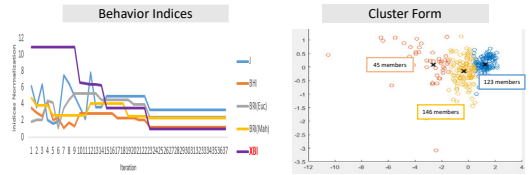
Indices normalization on BRI(Euc) as Object Function




Indices normalization on BRI(Mah) as Object Function




Indices normalization on XBI as Object Function




**Conclusion**

1. The crisp clustering indices can be fuzzified.
2. By minimizing indices, we get directly good clustering and show that amount of members in each indices almost same.
3. From table and graph of indices, we get BHI is a good index because need little iteration.

**Future work**

1. For the future work, we extend experiment for other indices which contain no min/max operators
2. Consider the optimization for which contains min/max. break through is to differentiate min/max

# Constructing non-symmetric closed categories from planar combinatory algebras

Haruka Tomita
special thanks to Masahito Hasegawa

RIMS, Kyoto University

2019/08/31



## Introduction

| Combinatory algebra | | Category (assemblies) |
|---|---|---|
| unit element | $\longrightarrow$ | identity |
| exchange | | symmetry |
| discard | | projection |



## Introduction

| Combinatory algebra | Structure of assemblies |
|---|---|
| SK-algebra | regular category, CCC |
| BCI-algebra | symmetric monoidal closed category |
| new BK $+ (-)^\bullet$-algebra | closed category |
| new BII$^\times + (-)^\bullet$-algebra | closed category |
| new BI $+ (-)^\bullet$-algebra | closed multicategory, skew closed category |
| new BB′II$^\bullet$-algebra | skew closed category |
| new BII$^\bullet + (-)^\circ$-algebra | skew closed category |
| BI-algebra | (ordinal) category |



## Preliminary: Applicative structure

**Definition. Applicative structure**

An **applicative structure** is a pair $(\mathcal{A}, \cdot)$
$\mathcal{A}$ is a set
$\cdot$ is a binary operation on $\mathcal{A}$ (not necessarily total)

## Preliminary: Category of assemblies

**Definition. Category of assemblies Asm($\mathcal{A}$)**

- object $(X, \|\text{-}\|_X)$
  a set $X$ and the **realizers** (for each $x \in X, \|x\|_X \subseteq \mathcal{A}$)
- map $(X, \|\text{-}\|_X) \xrightarrow{f} (Y, \|\text{-}\|_Y)$
  a function $f : X \to Y$ **realized** by some $\mathsf{r} \in \mathcal{A}$
  $$\forall x \in X, \forall \mathsf{a} \in \|x\|_X, \mathsf{r} \cdot \mathsf{a} \in \|f(x)\|_Y$$

$$\begin{array}{ccc} \mathsf{a} & \xrightarrow{\mathsf{r}} & \mathsf{r} \cdot \mathsf{a} \\ x & \xrightarrow{f} & f(x) \end{array}$$



## Previous studies: SK-algebra

**Definition. SK-algebra/PCA(Partial Combinatory Algebra)**

An **SK-algebra** is an applicative structure $(\mathcal{A}, \cdot)$ containing special two elements S and K satisfying

- $\forall \mathsf{x}, \mathsf{y}, \mathsf{z} \in \mathcal{A}$, $\mathsf{S} \cdot \mathsf{x} \cdot \mathsf{y} \cdot \mathsf{z} \simeq \mathsf{x} \cdot \mathsf{z} \cdot (\mathsf{y} \cdot \mathsf{z})$
- $\forall \mathsf{x}, \mathsf{y} \in \mathcal{A}$, $\mathsf{K} \cdot \mathsf{x} \cdot \mathsf{y} \simeq \mathsf{x}$

**Example. untyped $\lambda$-calculus**

$\beta$-equivalence classes of closed $\lambda$-terms form an SK-algebra.
$\mathsf{S} : \lambda xyz.xz(yz)$ $\quad$ $\mathsf{K} : \lambda xy.x$

(Completeness)
$\forall\, M$: lambda term, $\exists\, M^*$ constructed from S , K and elements of $FV(M)$ s.t. $\forall x_1, ..., x_n,\ Mx_1...x_n =_\beta M^*x_1...x_n$



## Previous studies: SK-algebra

**Theorem(Longley 1995)**

$\mathcal{A}$ is an SK-algebra $\Rightarrow$ Asm($\mathcal{A}$) is a regular Cartesian closed category.

$$(X, \|\text{-}\|_X) \times (Y, \|\text{-}\|_Y) := (X \times Y, \|\text{-}\|),$$

where $\|(x, y)\| := \{\lambda z.z\mathsf{xy} \mid \mathsf{x} \in \|x\|_X,\ \mathsf{y} \in \|y\|_Y\}$.
Projections are realized by $\lambda z.z(\lambda xy.x)$ and $\lambda z.z(\lambda xy.y)$

$$(Y, \|\text{-}\|_Y)^{(X, \|\text{-}\|_X)} := (Hom_{\mathsf{Asm}(\mathcal{A})}(X, Y), \|\text{-}\|),$$

where $\|f\| := \{\mathsf{r} \mid \mathsf{r} \text{ realizes } f\}$.
Evaluation maps are realized by $\lambda z.(z(\lambda xy.y))(z(\lambda xy.x))$



## Previous studies: BCI-algebra

**Definition. BCI-algebra**

A **BCI-algebra** is an applicative structure $(\mathcal{A}, \cdot)$ containing special three elements B, C and I satisfying

- $\mathsf{B} \cdot \mathsf{x} \cdot \mathsf{y} \cdot \mathsf{z} = \mathsf{x} \cdot (\mathsf{y} \cdot \mathsf{z})$
- $\mathsf{C} \cdot \mathsf{x} \cdot \mathsf{y} \cdot \mathsf{z} = \mathsf{x} \cdot \mathsf{z} \cdot \mathsf{y}$
- $\mathsf{I} \cdot \mathsf{x} = \mathsf{x}$

**Example. untyped linear $\lambda$-calculus**

$\beta$-equivalence classes of closed linear $\lambda$-terms form a BCI-algebra.
$\mathsf{B} : \lambda xyz.x(yz)$ $\quad$ $\mathsf{C} : \lambda xyz.xzy$ $\quad$ $\mathsf{I} :\ \lambda x.x$

(Completeness)
$\forall M$: linear lambda term, $\exists M^*$ constructed from B, C , I and $FV(M)$ s.t. $\forall x_1, ..., x_n,\ Mx_1...x_n =_\beta M^*x_1...x_n$

## Previous studies: BCI-algebra

**Theorem(Abramsky,Lenisa 2005)**

$\mathcal{A}$ is a BCI-algebra $\Rightarrow$ Asm($\mathcal{A}$) is a symmetric monoidal closed category.

$$(X, \|\text{-}\|_X) \otimes (Y, \|\text{-}\|_Y) := (X \times Y, \|\text{-}\|),$$

where $\|(x, y)\| := \{\lambda z.z\mathsf{xy} \mid \mathsf{x} \in \|x\|_X,\ \mathsf{y} \in \|y\|_Y\}$.
Unitors are realized by $\lambda x.x$ and $\lambda xy.yx$

$$(Y, \|\text{-}\|_Y) \multimap (X, \|\text{-}\|_X) := (Hom_{\mathsf{Asm}(\mathcal{A})}(X, Y), \|\text{-}\|),$$

where $\|f\| := \{\mathsf{r} \mid \mathsf{r} \text{ realizes } f\}$.
Evaluation maps are realized by $\lambda x.x(\lambda y.y)$



## Closed category

**Definition. Closed category (Eilenberg, Kelly, Laplaza)**

A **closed category** consists of the following data;

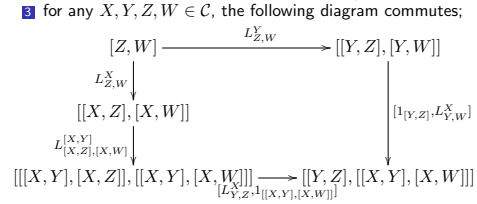
1. a locally small category $\mathcal{C}$
2. a functor $[-,-] : \mathcal{C}^{op} \times \mathcal{C} \to \mathcal{C}$
3. an object $I$, called unit object
4. a natural isomorphism $i_X : [I, X] \to X$
5. an extranatural transformation $j_X : I \to [X, X]$
6. a transformation $L^X_{Y,Z} : [Y, Z] \to [[X, Y], [X, Z]]$ natural in $Y, Z$ and extranatural in $X$

such that some axioms hold.



## Closed category

1. for any $X, Y \in \mathcal{C}$, $L^X_{Y,X} \circ j_Y = j_{[X,Y]}$
2. for any $X, Y \in \mathcal{C}$, $i_{[X,Y]} \circ [j_X, 1_{[X,Y]}] \circ L^X_{X,Y} = 1_{[X,Y]}$
3. for any $X, Y, Z, W \in \mathcal{C}$, the following diagram commutes;




4. for any $X, Y \in \mathcal{C}$, $L^I_{X,Y} \circ [1_X, i_Y] = [i_X, 1_{[I,Y]}]$
5. for any $X, Y \in \mathcal{C}$, the function $\gamma : \mathcal{C}(X, Y) \to \mathcal{C}(I, [X, Y])$ which sends $f : X \to Y$ to $[1_X, f] \circ j_X$ is invertible



## Failure

$\mathcal{A}$ : BI-algebra
$\mathcal{C} =$ Asm($\mathcal{A}$) : locally small category

- a functor $[-,-] : \mathcal{C}^{op} \times \mathcal{C} \to \mathcal{C}$
  × $[f, g]$ is not generally realized
- an object $I$, called unit object
  ○ same as BCI-algebra
- a natural isomorphism $i_X : [I, X] \to X$
  × $i_X$ is not realized
  × $i_X^{-1}$ is not realized
- an extranatural transformation $j_X : I \to [X, X]$
  ○ realized by I
- a transformation $L^X_{Y,Z} : [Y, Z] \to [[X, Y], [X, Z]]$
  ○ realized by B

## Failure

$f : X' \to X,\ g : Y \to Y'$

$$[f,g] : \begin{array}{ccc} [X,Y] & \to & [X',Y'] \\ (h : X \to Y) & \mapsto & (g \circ h \circ f : X' \to Y') \end{array}$$

.....

$\exists\ \mathsf{r}_{[f,g]}$ realizes $[f,g]$
$\Rightarrow\ \mathsf{r}_{[f,g]} \cdot \mathsf{r}_h$ realizes $g \circ h \circ f$

$\mathsf{a} \in \|x'\|_{X'} \quad \mathsf{r}_{[f,g]} \cdot \mathsf{r}_h \cdot \mathsf{a} \longleftrightarrow \mathsf{r}_g \cdot (\mathsf{r}_h \cdot (\mathsf{r}_f \cdot \mathsf{a}))$



## Solution1: B′

$\mathsf{B}' \cdot \mathsf{x} \cdot \mathsf{y} \cdot \mathsf{z} = \mathsf{y} \cdot (\mathsf{x} \cdot \mathsf{z})$

$\mathsf{C} \cdot \mathsf{x} \cdot \mathsf{y} \cdot \mathsf{z} = \mathsf{x} \cdot \mathsf{z} \cdot \mathsf{y}$

$[f,g]$ is realized by $\mathsf{B} \cdot (\mathsf{B} \cdot \mathsf{r}_g) \cdot (\mathsf{B}' \cdot \mathsf{r}_f)$

.....

BB′I•-algebra ⇒ skew closed category
(I• is an element s.t. $\forall \mathsf{x} \in \mathcal{A},\ \mathsf{I}^\bullet \cdot \mathsf{x} = \mathsf{x} \cdot \mathsf{I}$)



## Solution2: $(-)^\bullet$

**Notation: $(-)^\bullet$**

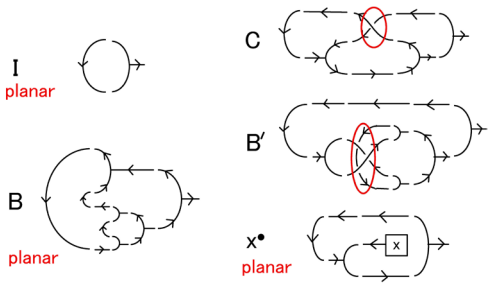
For $\mathsf{x} \in \mathcal{A}$, $\mathsf{x}^\bullet$ is an element s.t. $\forall \mathsf{y} \in \mathcal{A}$,
$\mathsf{x}^\bullet \cdot \mathsf{y} = \mathsf{y} \cdot \mathsf{x}$

**Definition. BI + $(-)^\bullet$-algebra**

A **BI $+(-)^\bullet$-algebra** is an applicative structure containing B, I and $\mathsf{x}^\bullet$ for all $\mathsf{x} \in \mathcal{A}$.



## BI + $(-)^\bullet$-algebra



## BI $+ (-)^\bullet$-algebra

**Example. untyped planar $\lambda$-calculus**

$\beta$-equivalence classes of closed planar $\lambda$-terms form a BI $+ (-)^\bullet$-algebra.

B : $\lambda xyz.x(yz)$ I : $\lambda x.x$ M$^\bullet$ : $\lambda x.x$M

(Completeness)
$\forall M$ : planar lambda term, $\exists M^*$ constructed from B, B$^\bullet$, B$^{\bullet\bullet}$, ..., I, I$^\bullet$, I$^{\bullet\bullet}$, ... and $FV(M)$ s.t. $\forall x_1, ..., x_n$, $Mx_1...x_n =_\beta M^*x_1...x_n$



## BI $+ (-)^\bullet$-algebra

$\mathcal{A}$ : BI $+ (-)^\bullet$-algebra
$\mathcal{C} =$ Asm($\mathcal{A}$) : locally small category

- a functor $[-,-] : \mathcal{C}^{op} \times \mathcal{C} \to \mathcal{C}$
  - ○ $[f, g]$ is realized by B · (B · r$_f{}^\bullet$ · B) · (B · r$_g$)
- an object $I$, called unit object
  - ○ same as BCI-algebra
- a natural isomorphism $i_X : [I, X] \to X$
  - ○ $i_X$ is realized by I$^\bullet$
  - × $i_X^{-1}$ is not realized
- an extranatural transformation $j_X : I \to [X, X]$
  - ○ realized by I
- a transformation $L^X_{Y,Z} : [Y, Z] \to [[X, Y], [X, Z]]$
  - ○ realized by B



## BI$^\times$ $+ (-)^\bullet$-algebra

**Notation: I$^\times$**

I$^\times$ is an element s.t. $\forall$x $\in \mathcal{A}$,
I$^\times$ · x · I = x

**Definition. BI$^\times$ $+ (-)^\bullet$-algebra**

An applicative structure $\mathcal{A}$ is a BI$^\times$ $+ (-)^\bullet$**-algebra** if it contains B, I, I$^\times$ and x$^\bullet$ for all x $\in \mathcal{A}$.

In planar lambda calculus, B $= \lambda xyz.x(yz)$ satisfies the condition of I$^\times$

**Proposition**

$\mathcal{A}$ is a BI$^\times$ $+ (-)^\bullet$-algebra $\Rightarrow$ Asm($\mathcal{A}$) is a closed category.



## BI$^\times$ $+ (-)^\bullet$-algebra

**Proposition**

Asm($\mathcal{A}$) is a closed category $\Rightarrow$ $\mathcal{A}$ is a BI$^\times$ $+ (-)^\bullet$-algebra whenever the following conditions hold;

- $[X, Y] = (Hom_{\mathsf{Asm}(\mathcal{A})}(X, Y), \|\text{-}\|)$ where $\|f\| = \{$r | r tracks $f\}$
- $[f, g] : [X, Y] \to [X', Y']$ is a function which sends $h : X \to Y$ to $g \circ h \circ f$
- $L^X_{Y,Z}$ sends $g : Y \to Z$ to the function $(f : X \to Y) \mapsto (g \circ f : X \to Z)$
- the underlying set of $I$ is the singleton $\{*\}$
- $i_X$ sends a function $(f : * \mapsto x)$ to $x$

## Skew closed category

**Definition. Skew closed category (Street)**

A **skew closed category** consists of the following data;

1. a locally small category $\mathcal{C}$
2. a functor $[-,-] : \mathcal{C}^{op} \times \mathcal{C} \to \mathcal{C}$
3. an object $I$, called unit object
4. a natural transformation $i_X : [I, X] \to X$
5. an extranatural transformation $j_X : I \to [X, X]$
6. a transformation $L^X_{Y,Z} : [Y, Z] \to [[X, Y], [X, Z]]$ natural in $Y, Z$ and extranatural in $X$

such that some axioms hold.



## Skew closed category

- for any $X, Y \in \mathcal{C}$, $L^X_{Y,Y} \circ j_Y = j_{[X,Y]}$
- for any $X, Y \in \mathcal{C}$, $i_{[X,Y]} \circ [j_X, 1_{[X,Y]}] \circ L^X_{X,Y} = 1_{[X,Y]}$
- for any $X, Y, Z, W \in \mathcal{C}$, the following diagram commutes;

$$\begin{array}{ccc} [Z,W] & \xrightarrow{L^Y_{Z,W}} & [[Y,Z],[Y,W]] \\ {\scriptstyle L^X_{Z,W}}\downarrow & & \downarrow{\scriptstyle [1_{[Y,Z]},L^X_{Y,W}]} \\ [[X,Z],[X,W]] & & \\ {\scriptstyle L^{[X,Y]}_{[X,Z],[X,W]}}\downarrow & & \\ [[[X,Y],[X,Z]],[[X,Y],[X,W]]] & \xrightarrow[[L^X_{Y,Z},1_{[[X,Y],[X,W]]}]]{} & [[Y,Z],[[X,Y],[X,W]]] \end{array}$$

- for any $X, Y \in \mathcal{C}$, $[1_{[I,X]}, i_Y] \circ L^I_{X,Y} = [i_X, 1_Y]$
- $i_I \circ j_I = 1_I$



## Skew closed category

Monoidal category $\mathcal{C}$ has natural isomorphisms

$$\rho_X : X \otimes I \to X$$
$$\lambda_X : X \to I \otimes X$$
$$\alpha_{XYZ} : X \otimes (Y \otimes Z) \to (X \otimes Y) \otimes Z$$

Monoidal closed category : $(A \otimes -) \dashv [A, -]$

Skew monoidal category $\mathcal{C}$ has natural transformations

$$\rho_X : X \otimes I \to X$$
$$\lambda_X : X \to I \otimes X$$
$$\alpha_{XYZ} : X \otimes (Y \otimes Z) \to (X \otimes Y) \otimes Z$$

Skew monoidal closed category : $(A \otimes -) \dashv [A, -]$



## Skew closed category

**Proposition**

$\mathcal{A}$ is a BI $+ (-)^\bullet$-algebra $\Rightarrow$ Asm($\mathcal{A}$)is a skew closed category.

The converse does not hold.

## Closed multi-category

**Definition. multi-category (Lambek)**

A **multi-category** $\mathcal{C}$ consists of the following data;

1. a collection $Ob(\mathcal{C})$
2. $\forall n \geq 0$ and $\forall X_1, X_2, ..., X_n, Y \in Ob(\mathcal{C})$, a set $\mathcal{C}(X_1, X_2, ..., X_n; Y)$
3. $\forall X \in Ob(\mathcal{C})$, an element $1_X \in \mathcal{C}(X; X)$, called identity
4. $\forall n, k_1, k_2, ..., k_n \in \mathbb{N}$ and $\forall X_{ij}, Y_i, Z$ $(i \leq n, j \leq k_i)$, a function $\prod_i^n \mathcal{C}(X_{i1}, ..., X_{ik_i}; Y_i) \times \mathcal{C}(Y_1, ..., Y_n; Z) \to \mathcal{C}(X_{11}, ..., X_{1k_1}, ..., X_{n1}, ..., X_{nk_n}; Z)$, it is called composition and the composition of $g \in \mathcal{C}(Y_1, ..., Y_n; Z)$ and $f_i \in \mathcal{C}(X_{i1}, ..., X_{ik_i}; Y_i)$ is written as $g \circ (f_1, ..., f_n)$



## Closed multi-category

**Definition. closed multi-category (Lambek)**

A **closed multi-category** consists of the following data;

1. a multi-category $\mathcal{C}$
2. $\forall X_1, X_2, ..., X_n, Y \in Ob(\mathcal{C})$, an object $\underline{\mathcal{C}}(X_1, X_2, ..., X_n; Y)$, called internal hom object.
3. $\forall X_1, ..., X_n, Y \in Ob(\mathcal{C})$, a map $ev_{X_1,...,X_n;Y} : X_1, ..., X_n, \underline{\mathcal{C}}(X_1, ..., X_n; Y) \to Y$, called evaluation map such that $\forall Z_1, Z_2, ..., Z_m \in Ob(\mathcal{C})$, the function $\varphi_{X_1,...,X_n;Z_1,...,Z_m;Y} : \mathcal{C}(Z_1, ..., Z_m; \underline{\mathcal{C}}(X_1, ..., X_n; Y)) \to \mathcal{C}(X_1, ..., X_n, Z_1, ..., Z_m; Y)$ which sends $f$ to $ev_{X_1,...,X_n;Y} \circ (1_{X_1}, ..., 1_{X_n}, f)$ is invertible.

**Theorem(Manzyuk 2012)**

$\mathrm{ClMulticat}^u$ and $\mathrm{ClCat}$ are equivalent.



## Closed multi-category

**Proposition**

$\mathcal{A}$ is a $\mathsf{BI} + (-)^\bullet$-algebra $\Rightarrow$ $\mathrm{Asm}(\mathcal{A})$ is a closed multi-category.

**Proposition**

$\mathrm{Asm}(\mathcal{A})$ is a closed multi-category $\Rightarrow$ $\mathcal{A}$ is a $\mathsf{BI} + (-)^\bullet$-algebra whenever some conditions hold.



## Conclusion

| Combinatory algebra | Structure of assemblies |
|---|---|
| $\mathsf{BK} + (-)^\bullet$-algebra | closed category |
| $\mathsf{BII}^\times + (-)^\bullet$-algebra | closed category |
| $\mathsf{BI} + (-)^\bullet$-algebra | closed multicategory, skew closed category |
| $\mathsf{BB'II}^\bullet$-algebra | skew closed category |
| $\mathsf{BII}^\bullet + (-)^\circ$-algebra | skew closed category |

## Future work

- Probably we can find more correspondences between algebras and categorical structures.
- At this point, we have few examples of $\mathsf{BI} + (-)^\bullet$-algebra. Some construction to get various $\mathsf{BI} + (-)^\bullet$-algebras is required.

h0904

# 数学と変数

竹内泉

**2019/08/31**

この発表は《数学の言語学》の研究

言語哲学
論理式を駆使して言語の意味を説明する
言語学の一分野

言語哲学者（飯田隆、松阪陽一ら）は〈言語哲学は哲学の一分野〉と言う
《言語哲学》ではなく《論理学的言語学》と呼んだ方がいいのかも知れない

言語哲学と論理学の違い

論理学：短い論理式を長い長い自然言語の文で説明する
例： $x = y \wedge \neg\Box\ x = y$
註：この □ の意味は伝聞であり、必然性ではない

言語哲学：短い自然言語の文を長い長い論理式で説明する
例：
$T = cPV$ なので $\dot{T} = \frac{\partial T}{\partial P}\dot{P} + \frac{\partial T}{\partial V}\dot{V} = cV\dot{P} + cP\dot{V}$

言語哲学の目的は〈知りたい〉という知的好奇心

工学者は、新しい言語処理系を作りたがるかも知れない
教育論者は、新しい言語教育法を作りたがるかも知れない
しかしそれは言語哲学の範囲の外

本研究：数学の中の変数の意味を分析する
　分析の方法は、変数が表れる数学の表現を述語論理に翻訳する

研究の意義

学術的意義
　数学には独立変数、従属変数など様々な変数が現れるにも関わらず、
　フレーゲは束縛変数しか使わない述語論理でどうやって数学を表現したのか
産業技術的効用
　束縛変数と自由変数しか現れない Coq、Mizar、Izabelle 等の証明系によって
　独立変数、従属変数などが現れる数学の方程式を形式化する方法

何の役に立つのか

学術的効用
　数学、論理学、哲学の歴史の研究に役立つ
　変数を使わないアリストテレス、現代的な変数の使用を始めたライプニッツ、
　束縛変数だけしか使わないフレーゲの比較

産業技術的用途
　数学の形式化に役立つ
　数学を使った仕様に対する Coq、Mizar、Isabelle 等の証明系を使った
　形式的検証

数学と産業の橋渡し

本研究の効用は、数学を述語論理に翻訳するための
・自動処理系の開発
・手作業で翻訳する人材の教育
の役に立つ

　（数学の教育の役には立たない
　　内容が高度であり、数学を既に習得している人でなければ理解出来ないから）

・処理系の開発、教育は本研究の範囲外
・本研究は処理系の開発、教育に役に立つ
両立する

組織的対処とは
大きな問題を細かく切って小さい問題に還元し
それぞれの問題をそれぞれの担当者が解決する
ということ

数学から述語論理への翻訳は産業の役に立つのか

米軍とエアバスはソフトウェアの製造に形式手法を義務付けている
現在ではまだ、力学方程式の形式化は求められていないが
近い将来、そういう時代が必ず来る

変数の使用は数学の言語に特有
変数の使用を分析することによって数学の本質に近付く

数学の中の変数の使用
・恒等式の中の文字（済）
・方程式の中の未知数（済）
・多項式論の変数と命題変数（済）
・座標変数と確率変数（済）
・独立変数と従属変数　←ここまで
・自由変数と束縛変数（未着手）
・統計学の変数（未着手）

## 恒等式の変数

恒等式の文字は、全称量化される変数
例：　恒等式：$a+b=b+a$
　　意味　：$\forall a \forall b.\ a+b=b+a$

数学で最も一般的な文字の使用法
数学では文字は適当な範囲を区切ってその範囲で全称量化される傾向がある
例：
$n$ が 2 以上の整数ならば、ある素数 $p$、$q$ があって $2n=p+q$
意味：$\forall n.\ \mathrm{Int}(n)\&\ n \geq 2 \Rightarrow \exists p,q.\ \mathrm{Prime}(p)\&\mathrm{Prime}(q)\&\ 2n=p+q$

## 方程式の未知数

方程式の未知数は、解との同値式が全称量化される

例：　方程式： $x^2 - 1 = 0$　　解： $x = 1$ と $x = -1$

　　意味　：$\forall x.\, x^2 - 1 = 0 \iff (x = 1 \vee x = -1)$

　　あるいは $\{x | x^2 - 1 = 0\} = \{x | x = 1 \vee x = -1\}$

恒等式の文字も、方程式の未知数も

量化子は各変数に対してそれぞれ一つしか登場しないので、変数の値は皆同じ

よって《文中で同一のものを指す記号》とも言える

しかしそれが全てではない　…　全称束縛されるという機能もある

ここまではまだ〈変数〉と言う用語は使わない

## 多項式論の中の変数と命題変数

ここで云う多項式論とは

　〈 $x^2 + 3x + 2$ を因数分解すると $(x + 1)(x + 2)$ 〉

恒等式と因数分解の式の比較

　恒等式　　：$a + b = b + a$

　因数分解の式：$x^2 + 3x + 2 = (x + 1)(x + 2)$

ここでの $a$ 、$b$ と $x$ の地位の違いは何か

恒等式の意味

$a+b=b+a$ の意味： $\forall a \forall b.\ a+b=b+a$

因数分解の式の意味

$x^2+3x+2=(x+1)(x+2)$ の意味：

・ $\forall x.\ x^2+3x+2=(x+1)(x+2)$ ではない

・可換環の公理だけを使って当該の等式が証明できること

即ち、多項式環 $\mathbf{Z}[x]$ の中で $x^2+3x+2$ と $(x+1)(x+2)$ が同一の点である、ということ

（ $\mathbf{Z}$ は無限整域でなので、上記二つは同値だが）

多項式論の変数は、環に対する付加元

可換環の圏 $\mathbf{CRng}$ から集合の圏 $\mathbf{Set}$ への忘却函手 $U$ には左随伴 $F$ がある

$F \dashv U$ 、$F$ は自由生成の函手

多項式環 $\mathbf{Z}[x]$ は生成元の集合 $\{x\}$ の $F$ による像である

$\mathbf{Set}(\{x\}, UR)$ から $\mathbf{CRng}(\mathbf{Z}[x], R)$ への同形写像 $\phi$ は代入を作る

$$\begin{array}{ccc} \mathbf{Set} & & \mathbf{CRng} \\ x \in \{x\} & \overset{F}{\longmapsto} & \mathbf{Z}[x] \ni f \\ \downarrow \quad \downarrow & \overset{\phi}{\longmapsto} & \downarrow \quad \downarrow \\ e \in UR & \underset{U}{\longmapsto} & R \ni f(e) \end{array}$$

$\mathbf{Z}[x]$ に於ける付加元 $x$ は

全ての可換環 $R$ の全ての元 $e$ を代入出来るもの

として特徴付けられる

命題変数も、論理式が成す代数の生成元である

どんな論理式も代入出来る

纏め

多項式論の変数と命題変数は生成元

どんなものも代入出来る

## 座標変数と確率変数

座標変数の用例

「 $y = 4x$ のグラフを書いてみよう。

かき方

$x = 1$ のとき $y = 4$ だから $y = 4x$ のグラフは点 (1,4) を通る。

したがって、原点と点 (1,4) を通る直線が求めるグラフである。」

澤田利夫監修、中学数学 1 、教育出版、2008 年 1 月、104 頁より

冒頭の《 $y = 4x$ 》の $x$ と $y$ は、$x = 1$ と $y = 4$ のことだけを

言っているのではない

$(x = 0, y = 0)$、$(x = 2, y = 8)$、$(x = 3, y = 12)$ 等の全てに言及している

座標とは

一般には、空間の点と数またはその組との一対一対応

平面のデカルト座標とは、平面の X 軸と Y 軸への射影

即ち、

X 軸への射影を $\hat{x}(\ )$ と書き、Y 軸への射影を $\hat{y}(\ )$ と書くと、

点 $p$ の X 座標とは $\hat{x}(p)$、点 $p$ の Y 座標とは $\hat{y}(p)$

これが一対一対応であるとは

・任意の点 $p$ に対し、$\hat{x}(p)$ と $\hat{y}(p)$ が定まる

・任意の $x \in R$、$y \in R$ に対し、$\hat{x}(p) = x, \hat{y}(p) = y$ となる点 $p$ が

唯一定まる

座標変数の用例

$y = 4x$

意味：$\{p | \hat{y}(p) = 4\hat{x}(p)\}$

$x = 1$ のとき $y = 4$

意味：$\hat{x}(p) = 1,\ \hat{y}(p) = 4$

座標変数 $x$、$y$ は、それぞれの文脈で注目している地点 $p$ の

函数値 $\hat{x}(p)$、$\hat{y}(p)$ を表す

座標変数の纏め

座標変数は、それぞれの文脈で注目している地点での、

座標系が定める函数の値を表す

確率変数の用例

普通の賽子を二回振る

一回目に出た目を $x$ と置き、二回目に出た目を $y$ と置く

$2x = y$ である確率は $1/12$ である

この $x$ と $y$ が確率変数

確率変数の値は《確率的に変化する》と言われる

確率変数が登場する式「$2x = y$」も《確率的》であると呼ばれる

――　真偽が《確率的に変化する》

一方、「$2x = y$ である確率は $1/12$ である」は《確率的》ではない

――　真偽は変化しない

確率論では

　確率変数は事象空間から値域への函数である

と教える

確率変数の用例

　普通の賽子を二回振る

　一回目に出た目を $x$ と置き、二回目に出た目を $y$ と置く

　$2x = y$ である確率は $1/12$ である

この場合の事象空間は

　$\Omega = \{1,2,3,4,5,6\} \times \{1,2,3,4,5,6\}$

確率変数 $x$ が表す函数 $\hat{x}(\ )$ とは：$\omega = (\omega_x, \omega_y)$ に対して $\hat{x}(\omega) = \omega_x$

確率変数 $y$ が表す函数 $\hat{y}(\ )$ とは：$\omega = (\omega_x, \omega_y)$ に対して $\hat{y}(\omega) = \omega_y$

確率変数を使った表記と、それが表すもの

| | |
|---|---|
| $x$ | $\hat{x}(\omega)$ |
| $y$ | $\hat{y}(\omega)$ |
| $2x = y$ | $2\hat{x}(\omega) = \hat{y}(\omega)$ |
| $2x = y$ である確率は $1/12$ | $\mu(\{\omega \in \Omega \mid 2\hat{x}(\omega) = \hat{y}(\omega)\}) = 1/12$ |

「$x$」、「$y$」、「$2x = y$」は《確率的》である

　——　右欄では事象を動く変数 $\omega$ が自由に現れている

「$2x = y$ である確率は $1/12$」は《確率的》ではない

　——　右欄では $\omega$ は束縛されている

直線を表す〈 $y = 4x$ 〉

〈 $2x = y$ である確率は〉の中の〈 $2x = y$ 〉

構文上は論理式であるものが

実は隠れた自由変数があり

意味論上は、その自由変数が束縛されて、性質あるいは集合となる

座標変数と確率変数は指標詞に似ている

指標詞とは

　《私》《此処》《今》

　発話者と発話時と発話地点を表すもの

例：「私は今は此処の教員です」

野本氏が 1988 年に北大で発言した場合には、この文の意味は

　《野本氏は 1988 年には北大の教員だった》

指標詞と座標変数の違い
指標詞を含む文はある命題を表す
「私は今は此処の教員です」は
《発話者は発話時に発話地点で教員である》という命題を表す
　例外：鉤括弧で括って発話から切り話せば、性質を表し得る
　『「明日は月曜」と言える日は日曜である』

座標変数を含む構文的に論理式であるものは
地点に関する性質を表す
式 $y = 4x$ は、各点でこの式が成り立つかどうかの性質を表す
数学では時に、その性質を充たす点の集合（即ち直線）の名前
●数学は性質への言及を嫌い、それに代わって集合を好む

# 独立変数と従属変数

中学一年の数学の教科書は、2011 年検定、2012 年出版のものより、
函数に関する説明が中学二年の教科書から降りてきて、かつ、
変数に関する説明が新設された。
「いろいろな値をとる文字を変数という。」
「ともなって変わる２つの変数 $x$、$y$ があって、$x$ の値を決めると、
　それに対応する $y$ の値がただ１つ決まるとき、
　$y$ は $x$ の関数であるという。」

独立変数と従属変数の説明

この用法は計算機科学では余り使わないらしい
物理学などの数学ではよく使われる　「圧力は温度の函数である。」

〈 $dx^2/dx$ 〉や〈 $\partial x^2 y/\partial x$ 〉の
分母の $x$ と分子中の $x$ とは関係付けられている
しかし束縛ではない
変数の名前換えが出来ないから

$y = x^2$ と置くと
$dx^2/dx = 2x$、一方で $dy^2/dy = 2y = 2x^2$

確率変数には暗黙の変数が隠れていた
従属変数にも暗黙の変数があるのか

従属変数は、独立変数、及び独立変数から従属変数を与える函数とから成る
独立変数は従属変数の中で暗黙の変数として働く
即ち

$$y = x^2,\ dy/dx = 2x$$

は

$$y = \hat{y}(x) = x^2,\ D\hat{y}x = D(\lambda x.x^2)x = (\lambda x.2x)x = 2x$$

但し $D$ は微分作用素
独立変数 $x$ は、翻訳後には、束縛変数と自由変数の双方として出現する
《独立変数の二面性》

〈従属変数は、独立変数と及び、独立変数から従属変数を与える函数とから成る〉
従属変数を従属変数で微分する場合はどうなるのか
偏微分はどうなるのか
例

$y_1 = x_1 + x_2,\ y_2 = x_1 - x_2,\ z = y_1y_2 = x_1^2 - x_2^2$
$\partial z/\partial y_1 = \partial y_1y_2/\partial y_1 = y_2 = x_1 - x_2$
独立変数は $x_1$ と $x_2$

$\partial/\partial y_1$ は、$y_2$ を動かさないで $y_1$ で微分する、という記号
$\partial/\partial y_1$ では $y_2$ は明記されていないが言及されている
動かさない変数が明らかでない限り使ってはいけない
不便な記号ではあるが広く普及している
この記号の不便さは今回の話題ではない

例文：
$T = cPV$ なので $\dot{T} = \frac{\partial T}{\partial P}\dot{P} + \frac{\partial T}{\partial V}\dot{V} = cV\dot{P} + cP\dot{V}$

独立変数は時間 $t$ であるが、
$\partial T/\partial P$ や $\partial T/\partial V$ の中では $P$、$V$ もまた独立変数である

従属変数の二つの機能
(1) 値を保持する
$y = x^2$ の下で、$x = 2$ の時 $y = 4$
(2) 独立変数との関係を保持する：独立変数からの函数
$y = x^2$ の下で、$x$ が 2 から 3 へ動く時、$y$ は 4 から 9 へ動く

独立変数の二面性に対応した従属変数の機能
自由変数としての独立変数　…　値の保持
束縛変数としての独立変数　…　独立変数からの函数

数学の変数の述語論理への翻訳

数学の変数 $x$ を表す述語論理の変数を $\mathbf{v}^x$ と書く

… 値の保持に関する翻訳

数学の変数 $y$ が数学の変数 $x$ の函数であるとき
数学の変数 $x$ の値から数学の変数 $y$ の値を返す函数を表す
述語論理の函数記号を $\mathbf{f}^y_x$ と書く

… 独立変数からの函数に対する翻訳

数学の言語「 $y$ は $x$ の函数」を述語論理で書くと「 $\mathbf{v}^y = \mathbf{f}^y_x(\mathbf{v}^x)$ 」

「 $\boldsymbol{T = cPV}$ なので $\dot{\boldsymbol{T}} = \frac{\partial \boldsymbol{T}}{\partial \boldsymbol{P}}\dot{\boldsymbol{P}} + \frac{\partial \boldsymbol{T}}{\partial \boldsymbol{V}}\dot{\boldsymbol{V}} = \boldsymbol{cV}\dot{\boldsymbol{P}} + \boldsymbol{cP}\dot{\boldsymbol{V}}$ 」

明記されていないが、独立変数は時間 $t$

述語論理への翻訳は

$\mathbf{v}^P = \mathbf{f}^P_t(\mathbf{v}^t)$
$\mathbf{v}^V = \mathbf{f}^V_t(\mathbf{v}^t)$
$\mathbf{v}^T = \mathbf{f}^T_t(\mathbf{v}^t) = \mathbf{f}^T_{PV}(\mathbf{v}^P, \mathbf{v}^V)$
$\forall \mathbf{v}^t .\ \mathbf{f}^T_t(\mathbf{v}^t) = \mathbf{f}^T_{PV}(\mathbf{f}^P_t(\mathbf{v}^t), \mathbf{f}^V_t(\mathbf{v}^t))$
$\forall \mathbf{v}^P \mathbf{v}^V .\ \mathbf{f}^T_{PV}(\mathbf{v}^P, \mathbf{v}^V) = c \times \mathbf{v}^P \times \mathbf{v}^V$
$\boldsymbol{D}\mathbf{f}^T_t \mathbf{v}^t$
$= \boldsymbol{D}(\lambda \mathbf{v}^P . \mathbf{f}^T_{PV}(\mathbf{v}^P, \mathbf{v}^V))\mathbf{v}^P \times \boldsymbol{D}\mathbf{f}^P_t \mathbf{v}^t + \boldsymbol{D}(\lambda \mathbf{v}^V . \mathbf{f}^T_{PV}(\mathbf{v}^P, \mathbf{v}^V))\mathbf{v}^V \times \boldsymbol{D}\mathbf{f}^V_t \mathbf{v}^t$
$= c \times \mathbf{v}^V \times \boldsymbol{D}\mathbf{f}^P_t \mathbf{v}^t + c \times \mathbf{v}^P \times \boldsymbol{D}\mathbf{f}^V_t \mathbf{v}^t$

●各従属変数の、独立変数に対する依存関係
$\mathbf{v}^P = \mathbf{f}^P_t(\mathbf{v}^t), \quad \mathbf{v}^V = \mathbf{f}^V_t(\mathbf{v}^t), \quad \mathbf{v}^T = \mathbf{f}^T_t(\mathbf{v}^t) = \mathbf{f}^T_{PV}(\mathbf{v}^P, \mathbf{v}^V)$
● $\boldsymbol{T}$ が $t$ の函数であると同時に $\boldsymbol{P}$、$\boldsymbol{V}$ の函数であることに由来する函数間の関係
$\forall \mathbf{v}^t .\ \mathbf{f}^T_t(\mathbf{v}^t) = \mathbf{f}^T_{PV}(\mathbf{f}^P_t(\mathbf{v}^t), \mathbf{f}^V_t(\mathbf{v}^t))$
●「 $\boldsymbol{T = cPV}$ 」
$\forall \mathbf{v}^P \mathbf{v}^V .\ \mathbf{f}^T_{PV}(\mathbf{v}^P, \mathbf{v}^V) = c \times \mathbf{v}^P \times \mathbf{v}^V$
●「 $\dot{\boldsymbol{T}} = \frac{\partial \boldsymbol{T}}{\partial \boldsymbol{P}}\dot{\boldsymbol{P}} + \frac{\partial \boldsymbol{T}}{\partial \boldsymbol{V}}\dot{\boldsymbol{V}} = \boldsymbol{cV}\dot{\boldsymbol{P}} + \boldsymbol{cP}\dot{\boldsymbol{V}}$ 」
$\boldsymbol{D}\mathbf{f}^T_t \mathbf{v}^t$
$= \boldsymbol{D}(\lambda \mathbf{v}^P . \mathbf{f}^T_{PV}(\mathbf{v}^P, \mathbf{v}^V))\mathbf{v}^P \times \boldsymbol{D}\mathbf{f}^P_t \mathbf{v}^t + \boldsymbol{D}(\lambda \mathbf{v}^V . \mathbf{f}^T_{PV}(\mathbf{v}^P, \mathbf{v}^V))\mathbf{v}^V \times \boldsymbol{D}\mathbf{f}^V_t \mathbf{v}^t$
$= c \times \mathbf{v}^V \times \boldsymbol{D}\mathbf{f}^P_t \mathbf{v}^t + c \times \mathbf{v}^P \times \boldsymbol{D}\mathbf{f}^V_t \mathbf{v}^t$

$\boldsymbol{\partial T / \partial P}$ に於いて、$\boldsymbol{P}$、$\boldsymbol{V}$ は局所的な独立変数である
$\boldsymbol{\partial T / \partial V}$ に於いても同様

局所的な独立変数は
《局所的》に効果があるという点では束縛変数と似ているが
名前換えが出来ないという点で束縛変数とは異なる

今日の結論

●微分式に於ける独立変数を翻訳すると、束縛変数と自由変数の
双方として現れる
…《独立変数の二面性》

●従属変数には《値の保持》と《独立変数からの函数》の二つの機能がある
…《独立変数の二面性》に対応した機能

●独立変数には大域的な独立変数と局所的な独立変数とがある
微分式の分母に表れる変数は分子の中で局所的に独立変数として働く

述語論理への翻訳は説明したが、独立変数の本質へ迫るのはまだこれから

木村大輔氏（東邦大）の批判
「変数の動的な意味を把えていない」　→動的な意味とは何か

五十嵐淳氏（京大）の助言
「述語論理は適していないのではないか」

計算機科学者からの有益な助言が多い
・個人的人間関係
・述語論理に明るい
・数学者ではないが数学に明るい

更なる仮説：仮想的時空

・数学者は〈独立変数を動かす〉という時、《仮想的時空》を考える。
・仮想的時空の中では時間が流れており、各変数の値が時間に沿って変化する
・《仮想的時空》とは、小説や伝聞、回想、未来の想像の際に設定される場と
同様のもの
・仮想的時空の中で数学者は独立変数を動かし、それに従って従属変数が動く
・仮想的時空は仮想的なので再生、複製、
また同時に複数の時空を動かすことなどが出来る
・独立変数を取り直す時、元の仮想的時空は停止され、
新たな仮想的時空が動き出す
・仮想的時空の中の時間の流れが変数の動的意味にとって本質的である

\Orchestrating a brighter world NEC

# 命題論理式の条件付確率の論理と計算

**ALGI 30**
九州大学
**2019 年 8 月 31 日**
日本電気株式会社　中央研究所
本浦庄太

## Motivating example

条件付確率の値やその相互の関係から、指定された条件付確率の値を求める

条件付確率の値：

- $P($５年生存 $\mid$ 肺がん$) = 25\%$
- $P($肺がん $\mid$ 男性 $\wedge$ がん$) = 15\%$
- $P($がん $\mid$ 肺がん$) = 100\%$

条件付確率の相互の関係：

- $P($がん $\mid$ 男性 $\wedge$ 喫煙$) = 1.6 \times P($がん $\mid$ 男性 $\wedge \neg$喫煙$)$
- $P($がん $\mid$ 女性 $\wedge$ 喫煙$) = 1.4 \times P($がん $\mid$ 女性 $\wedge \neg$喫煙$)$

指定された条件付確率の値：

- $X\% \leq P($５年生存 $\mid$ 男性 $\wedge \neg$喫煙 $\wedge$ 肺がん$) \leq Y\%$

## 目次





## 命題論理式の条件付確率の定義（1/2）

$\mathcal{A}$ は命題変数の有限集合とする。

**Definition**

命題論理式 は以下のルールで定義される：

$$\varphi ::= a \mid \neg\varphi \mid \varphi \wedge \psi,$$

ただし、$a \in \mathcal{A}$。命題論理式全体は $\mathcal{L}(\mathcal{A})$ と書く。

**Definition**

論理式代数 $\mathcal{L}(\mathcal{A}) = (\mathcal{L}(\mathcal{A}), \{a\}_{a\in\mathcal{A}}, NOT, AND)$：

- $NOT(\varphi) = \neg\varphi$
- $AND(\varphi, \psi) = \varphi \wedge \psi$

## 命題論理式の条件付確率の定義（2/2）

**Definition (Popper function (Popper 1959))**

$\mathcal{L}(\mathcal{A})$ 上の Popper function は

$$P : \mathcal{L}(\mathcal{A}) \times \mathcal{L}(\mathcal{A}) \to \mathbb{R}$$

で、以下を満たすもの：

PF1 $(\forall xy)(\exists x'y')P(x|y) \neq P(x'|y')$

PF2 $(\forall xy)(((\forall z)P(x|z) = P(y|z)) \to ((\forall w)P(w|x) = P(w|y)))$

PF3 $(\forall xy)P(x|x) = P(y|y)$

PF4 $(\forall xyz)P(x \wedge y|z) \leq P(x|z)$

PF5 $(\forall xyz)P(x \wedge y|z) = P(x|y \wedge z) \cdot P(y|z)$

PF6 $(\forall xy)(\neg(\forall z)P(z|y) = P(y|y)) \to (P(x|y) + P(\neg x|y) = P(y|y))$



## 論理（1/5）

**Definition (言語 $\mathcal{L}_{\mathrm{PF}}$)**

1. ソート：form, real;
2. 変数 ：($\sigma$ = form or real)
   - 一階変数 $x : \sigma$
   - 単項二階変数 $X : \sigma$
3. 関数記号：

   $a$ : form for $a \in \mathcal{A}$　　0 : real, 1 : real
   $f_\neg$ : form $\to$ form　　$+$ : real $\times$ real $\to$ real
   $f_\wedge$ : form $\times$ form $\to$ form　　$\cdot$ : real $\times$ real $\to$ real
   $p$ : form $\times$ form $\to$ real
4. 関係記号：$\leq$ (real, real)
5. 論理結合子：
   - $\neg$, $\wedge$, $(\exists x : \text{form})$, $(\exists x : \text{real})$, $(\exists X : \text{form})$, $(\exists X : \text{real})$

論理記号として等号 $=$ を含む。



## 論理（2/5）

$f_\neg(\varphi)$ や $f_\wedge(\varphi, \psi)$ は、$\neg\varphi$ と $\varphi \wedge \psi$ と書く。
量化子 $(\exists x : \sigma)$ の sort $\sigma$ は文脈から明らかな場合は省略する。

**Example ($\mathcal{L}_{\mathrm{PF}}$ の論理式)**

PFL

PF1 $(\forall xy)(\exists x'y')p(x|y) \neq p(x'|y')$

PF2 $(\forall xy)(((\forall z)p(x|z) = p(y|z)) \to ((\forall w)p(w|x) = p(w|y)))$

PF3 $(\forall xy)p(x|x) = p(y|y)$.

PF4 $(\forall xyz)p(x \wedge y|z) \leq p(x|z)$.

PF5 $(\forall xyz)p(x \wedge y|z) = p(x|y \wedge z) \cdot p(y|z)$

PF6 $(\forall xy)(\neg(\forall z)p(z|y) = p(y|y)) \to (p(x|y) + p(\neg x|y) = p(y|y))$

- $p(a|\top) = 1$
- $(\forall x : \text{form})p(a \wedge b|x) = p(a|x) \cdot p(b|x)$



## 論理（3/5）

**Definition (構造)**

$\mathcal{L}_{\mathrm{PF}}$-構造（構造）$\mathcal{M} = (M_{\text{form}}, M_{\text{real}}, p^{\mathcal{M}})$：

- $M_{\text{form}} = (S, \{a^{\mathcal{M}}\}_{a \in \mathcal{A}}, f_\neg^{\mathcal{M}}, f_\wedge^{\mathcal{M}})$
- $M_{\text{real}} = (U, 0^{\mathcal{M}}, 1^{\mathcal{M}}, +^{\mathcal{M}}, \cdot^{\mathcal{M}}, \leq^{\mathcal{M}})$.
- $p^{\mathcal{M}} : S \times S \to U$

**Example**

Popper function $P : \mathcal{L}(\mathcal{A}) \times \mathcal{L}(\mathcal{A}) \to \mathbb{R}$ は構造 $(\mathcal{L}(\mathcal{A}), \mathbb{R}, P)$ と見ることができる。

## 論理（4/5）

**Definition (割り当て)**

割り当て $\mu$ は以下のように割り当てる：

- 一階変数 $x:\sigma$ に対して元 $\mu(x) \in M_\sigma$
- 単項二階変数 $X:\sigma$ に対して部分集合 $\mu(X) \subseteq M_\sigma$

ただし $\sigma$ は form および real.



## 論理（5/5）

**Definition (解釈)**

$\mathcal{M}$：構造、$\mu$：割り当て

- $a^{\mathcal{M}}[\mu] = a^{\mathcal{M}}$ for any $a \in \mathcal{A}$
- $c^{\mathcal{M}}[\mu] = c^{\mathcal{M}}$ for $c = 0, 1$
- $x^{\mathcal{M}}[\mu] = \mu(x)$ for 一階変数 $x:\sigma$
- $f(t_1, \ldots t_n)^{\mathcal{M}}[\mu] = f^{\mathcal{M}}(t_1{}^{\mathcal{M}}[\mu], \ldots, t_n{}^{\mathcal{M}}[\mu])$ for $f = f_{\neg}, f_{\wedge}, +, \cdot$

| | | |
|---|---|---|
| $\mathcal{M} \models t:\sigma = s:\sigma[\mu]$ | iff | $t^{\mathcal{M}}[\mu] = s^{\mathcal{M}}[\mu]$ |
| $\mathcal{M} \models t:\mathsf{real} \leq s:\mathsf{real}[\mu]$ | iff | $t^{\mathcal{M}}[\mu] \leq^{\mathcal{M}} s^{\mathcal{M}}[\mu]$ |
| $\mathcal{M} \models X(t)[\mu]$ | iff | $t^{\mathcal{M}}[\mu] \in \mu(X)$ |
| $\mathcal{M} \models \neg\varphi[\mu]$ | iff | $\mathcal{M} \not\models \varphi[\mu]$ |
| $\mathcal{M} \models \varphi \wedge \psi[\mu]$ | iff | $\mathcal{M} \models \varphi[\mu]$ and $\mathcal{M} \models \psi[\mu]$ |
| $\mathcal{M} \models (\exists x:\sigma)\varphi[\mu]$ | iff | $\exists e \in M_\sigma$ s.t. $\mathcal{M} \models \varphi[\mu(e/x)]$ |
| $\mathcal{M} \models (\exists X:\sigma)\varphi[\mu]$ | iff | $\exists R \subseteq M_\sigma$ s.t. $\mathcal{M} \models \varphi[\mu(R/X)]$ |

$\mathcal{M} \models \varphi$：任意の $\mu$ に対して $\mathcal{M} \models \varphi[\mu]$



## 定義可能性（1/2）

$$P : \mathcal{L}(\mathcal{A}) \times \mathcal{L}(\mathcal{A}) \to \mathbb{R}$$

**Definition**

論理式代数の公理 FL は以下の集合：

F1 $(\forall xx')\neg x = \neg x' \to x = x'$

F2 $(\forall xy)x \wedge y = x' \wedge y' \to (x = x') \wedge (y = y')$

F3 $(\forall xyz)\neg x \neq y \wedge z$

F4 $(\forall xy)a \neq x \wedge y$ for any $a \in \mathcal{A}$

F5 $(\forall y)a \neq \neg x$ for any $a \in \mathcal{A}$

F6 $(\forall X)[[(\bigwedge_{a\in\mathcal{A}} X(a)) \wedge (\forall x)(X(x) \to X(\neg x)) \wedge (\forall xy)(X(x) \wedge X(y) \to X(x \wedge y))] \to (\forall x)X(x)]$

**Lemma**

$M_{\mathsf{form}} \models \mathsf{FL}$ ならば $M_{\mathsf{form}}$ と $\mathcal{L}(\mathcal{A})$ は同型。



## 定義可能性（2/2）

**Theorem (M.)**

$\mathcal{M}$： 構造, RFL： 実数の公理
このとき、以下は同値：

1. $\mathcal{M} \models \mathsf{FL} + \mathsf{PFL} + \mathsf{RFL}$
2. $\mathcal{M}$ は *a Popper function* に同型

$\mathcal{M}$ は Popper function $P$ に同型：

$$\begin{array}{ccc} M_{\mathsf{form}} \times M_{\mathsf{form}} & \xrightarrow{P^{\mathcal{M}}} & M_{\mathsf{real}} \\ \downarrow{\scriptstyle f \times f} & & \downarrow{\scriptstyle g} \\ \mathcal{L}(\mathcal{A}) \times \mathcal{L}(\mathcal{A}) & \xrightarrow{P} & \mathbb{R} \end{array}$$

for some isomorphisms $f : M_{\mathsf{form}} \to \mathcal{L}(\mathcal{A})$ and $g : M_{\mathsf{form}} \to \mathbb{R}$.

## 決定可能性（1/9）

### Definition

1. Popper formula とは $\mathcal{L}_{\mathrm{PF}}$ の一階述語論理式であって form-terms が $p$ の中にしか現れないもの。
2. 自由変数を持たない Popper formula を Popper sentence という。

### Example (Popper sentences)

PF1. $(\forall xy)(\exists x'y')p(x|y) \neq p(x'|y')$
PF2. $(\forall xy)(((\forall z)p(x|z) = p(y|z)) \to ((\forall w)p(w|x) = p(w|y)))$
PF3. $(\forall xy)p(x|x) = p(y|y)$.
PF4. $(\forall xyz)p(x\wedge y|z) \leq p(x|z)$.
PF5. $(\forall xyz)p(x\wedge y|z) = p(x|y\wedge z)\cdot p(y|z)$
PF6. $(\forall xy)(\neg(\forall z)p(z|y) = 1) \to (p(x|y) + p(\neg x|y) = p(y|y))$

### Theorem (M.)

任意の *Popper sentence* $\varphi$ に対して、以下は決定可能：

$$\textit{Popper functions} \models \varphi$$



## 決定可能性（2/9）

$\varphi$：Popper sentence

⇓

変換 $T_1$

⇓

変換 $T_2$

⇓

変換 $T_3$

⇓

実数の閉論理式の決定可能性に帰着



## 決定可能性（3/9）

### Lemma

$\phi \leftrightarrow \phi'$ と $\psi \leftrightarrow \psi'$ ならば $P(\varphi|\psi) = P(\varphi'|\psi')$ が成立する

### Definition

Lindenbaum-Tarski 代数 $\mathbb{L}(\mathcal{A}) = (L, -, \cdot)$ は $\mathcal{L}(\mathcal{A})/\leftrightarrow$：

- $L = \mathcal{L}(\mathcal{A})/\leftrightarrow$
- $-([\varphi]) = [\neg\varphi]$
- $[\varphi]\cdot[\psi] = [\varphi\wedge\psi]$

### Definition (Quotient Popper functions)

quotient Popper function $Q : \mathbb{L}(\mathcal{A}) \times \mathbb{L}(\mathcal{A}) \to \mathbb{R}$ は PFL を満たす関数。

PF1. $(\forall xy)(\exists x'y')Q(x|y) \neq Q(x'|y')$
PF2. $(\forall xy)(((\forall z)Q(x|z) = Q(y|z)) \to ((\forall w)Q(w|x) = Q(w|y)))$
PF3. $(\forall xy)Q(x|x) = Q(y|y)$.
PF4. $(\forall xyz)Q(x\wedge y|z) \leq Q(x|z)$.
PF5. $(\forall xyz)Q(x\wedge y)|z) = Q(x|y\wedge z)\cdot Q(y|z)$
PF6. $(\forall xy)(\neg(\forall z)Q(z|y) = 1) \to (Q(x|y) + Q(\neg x|y) = Q(y|y))$



## 決定可能性（4/9）

### Definition (言語 $\mathcal{L}_{\mathrm{QPF}}$)

1. ソート：form, real;
2. 変数 ：($\sigma$ = form or real)
   - 一階変数 $x : \sigma$
   - 単項二階変数 $X : \sigma$
3. 関数記号：

$z$ : form for $z \in \mathbb{L}(\mathcal{A})$　　0 : real, 1 : real
$f_\neg$ : form → form　　$+$ : real × real → real
$f_\wedge$ : form × form → form　　$\cdot$ : real × real → real
$p$ : form × form → real

4. 関係記号：$\leq$: (real, real)
5. 論理結合子：
   - $\neg$, $\wedge$, $(\exists x : \text{form})$, $(\exists x : \text{real})$, $(\exists X : \text{form})$, $(\exists X : \text{real})$

論理記号として等号 $=$ を含む。

## 決定可能性（5/9）

**Definition**

変換 $T_1 : \mathcal{L}_{\mathsf{PF}} \to \mathcal{L}_{\mathsf{QPF}}$:

- $T_1(a) = [a]$ for $a \in \mathcal{A}$
- 他の記号については準同型的に定義

**Example**

$$T_1(p(a \wedge b | a \wedge b) = 1) = \text{`}p([a] \wedge [b] | [a] \wedge [b]) = 1\text{'}$$

**Lemma**

$\varphi$ : *Popper sentence*

$$\textit{Popper functions} \models \varphi \iff \textit{quotient PFs} \models T_1(\varphi)$$



## 決定可能性（6/9）

**Definition**

$T_2 : \mathcal{L}_{\mathsf{QPF}} \to \mathcal{L}_{\mathsf{QPF}}$:

- $T_2((\exists x : \mathsf{form})\varphi) = \bigvee_{z \in \mathbb{L}(\mathcal{A})} T_2(\varphi)[z/x]$ for each formula $\varphi$
- そのほかは準同型的に変換

$\mathcal{A}$ が有限だから $\mathbb{L}(\mathcal{A})$ は有限なので、右辺は有限個の論理式の選言。

**Example**

$$T_2((\exists x : \mathsf{form})p(x|[a]) = 1) = \text{`} \bigvee_{z \in \mathbb{L}(\mathcal{A})} p(z|[a]) = 1\text{'}$$

**Lemma**

$\varphi$ : *Popper sentence*

$$\textit{quotient PFs} \models T_1(\varphi) \iff \textit{quotient PFs} \models T_2 \circ T_1(\varphi)$$



## 決定可能性（7/9）

**Definition**

$\mathbb{L}$ : variable-free form-terms $\to \mathbb{L}(\mathcal{A})$:

1. $\mathbb{L}(z) = z$ for any $z \in \mathbb{L}(\mathcal{A})$
2. $\mathbb{L}(\neg t) = -\mathbb{L}(t)$
3. $\mathbb{L}(t \wedge s) = \mathbb{L}(t) \cdot \mathbb{L}(s)$

**Definition**

$T_3$ : form-variable-free $\mathcal{L}_{\mathsf{QPF}} \to \mathcal{L}_{\mathbb{R}}(\mathcal{A})$

- $T_1(p(t|s)) = V_{(\mathbb{L}(t)|\mathbb{L}(s))}$, （実数の一階の変数）
- 他の場合は準同型的に変換

**Example**

$T_3(p([a] \wedge [b] | [a] \wedge [b]) = 1) = \text{`}V_{([a]\cdot[b]|[a]\cdot[b])} = 1\text{'} = \text{`}V_{([a\wedge b]|[a\wedge b])} = 1\text{'}$



## 決定可能性（8/9）

**Lemma**

$\varphi$ : *Popper sentence*
以下は同値：

- *quotient Popper functions* $\models T_2 \circ T_1(\varphi)$
- $\mathbb{R} \models (\forall V_{(x|y)})_{(x,y) \in \mathbb{L}(\mathcal{A}) \times \mathbb{L}(\mathcal{A})}(T_3 \circ T_2 \circ T_1(\mathsf{PFL}) \to T_3 \circ T_2 \circ T_1(\varphi))$

## 決定可能性（9/9）

**Theorem (M.)**

任意の *Popper sentence* $\varphi$ に対して：

*Popper functions* $\models \varphi$ か否かは決定可能。

(Proof)

Popper functions $\models \varphi$
$\iff$ quotient Popper functions $\models T_1(\varphi)$
$\iff$ quotient Popper functions $\models T_2 \circ T_1(\varphi)$
$\iff \mathbb{R} \models (\forall V_{(x|y)})_{(x,y)\in\mathbb{L}(\mathcal{A})\times\mathbb{L}(\mathcal{A})}(T_3 \circ T_2 \circ T_1(\mathrm{PFL}) \to T_3 \circ T_2 \circ T_1(\varphi))$



## 計算（1/1）

$\varphi$：確率の値やその相互の関係の連言
$P(A|B)$：求めたい条件付確率

$\Downarrow$

$$(\exists V_{(x|y)})_{\substack{(x,y)\in\mathbb{L}(\mathcal{A})\times\mathbb{L}(\mathcal{A})\\(x,y)\neq([A],[B])}}(T_3 \circ T_2 \circ T_1(\mathrm{PFL}) \wedge T_3 \circ T_2 \circ T_1(\varphi))$$

$\Downarrow$ 実数の量化子除去

$V_{([A],[B])}$の多項式間の等式・不等式のブール結合

このブール結合が定める領域が $P(A|B)$ が取りうる値の範囲



## Conclusion and future work （1/3）

Conclusion:

- 多ソート単項二階述語論理で Popper functions を定義
- Popper sentence の妥当性の決定可能性を証明
- 条件付確率を計算する方法を提案

Other results:

- 指定された条件付確率群の取りうる値の範囲のグリッド表示
  (0%) ■■■■□□□□□□ (100%)



## Conclusion and future work （2/3）

Future work:

- 未知の命題変数に対する弱さへの対応
  - $0 \leq P(\text{がん} \mid \text{喫煙} \wedge \text{甘党}) \leq 1$
- 一階述語論理での近似（完全性）
  - 一階述語論理式群 ∩（論理式代数+実数+Popper function）
  - ≒局所絶対自由代数 +実閉体+Popper function
- 意味論の論理化
  - sequent calculus の条件付確率的意味論

## Conclusion and future work （3/3）

一階述語論理での近似

$Inc : \mathbb{N} \hookrightarrow \mathbb{R}$

- $\mathbb{N}$ で成立する一階述語論理式群
- 実閉体の一階述語論理式群
- $Inc(0) = 0$
- $Inc(S(n)) = Inc(n) + 1$

$Inc : \mathbb{N} \hookrightarrow {}^*\mathbb{R}$

- 上の 4 つを満たす

アルキメデス性

- $Inc : \mathbb{N} \hookrightarrow \mathbb{R} \models (\forall x : \mathrm{real})(\exists y : \mathrm{natural}) x \leq y$
- $Inc : \mathbb{N} \hookrightarrow {}^*\mathbb{R} \not\models (\forall x : \mathrm{real})(\exists y : \mathrm{natural}) x \leq y$



ご清聴ありがとうございました

本浦　庄太
（もとうら　しょうた）

NEC　中央研究所
motoura@ct.jp.nec.com



\Orchestrating a brighter world

NEC

# Enriched categories and tropical mathematics

Soichiro Fujii

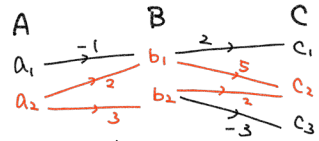
## §1. Introduction

What is tropical mathematics ?

| ordinary math. | tropical math. |
|---|---|
| $\mathbb{R}$, $\mathbb{C}$ | idempotent semiring (esp. quantale) |
| | $\overline{\mathbb{R}}_{\min,+}$, $\overline{\mathbb{R}}_{\max,+}$, $\mathcal{P}(M)$, ... |

## Shortest path and $\overline{\mathbb{R}}_{\min,+}$ [Cuninghame-Green, 1991]

Q. Find the length of a shortest path.




$\min\{2+5,\ 3+2\}$

$= (2 \odot 5) \oplus (3 \odot 2)$

$= 7 \oplus 5$

$= 5$

$\oplus$ ... min

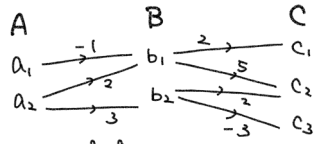
$\odot$ ... $+$

$\overline{\mathbb{R}}_{\min,+} = ([-\infty, \infty], \infty, 0, \min, +)$

zero one $\oplus$ $\odot$

## Shortest path and $\overline{\mathbb{R}}_{\min,+}$ [Cuninghame-Green, 1991]

Q. Find the length of a shortest path.

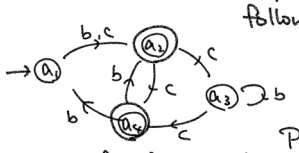



$$X = \begin{array}{c} \\ b_1 \\ b_2 \end{array} \begin{array}{c} \begin{array}{cc} a_1 & a_2 \end{array} \\ \begin{bmatrix} -1 & 2 \\ \infty & 3 \end{bmatrix} \end{array} \qquad Y = \begin{array}{c} \\ c_1 \\ c_2 \\ c_3 \end{array} \begin{array}{c} \begin{array}{cc} b_1 & b_2 \end{array} \\ \begin{bmatrix} 2 & \infty \\ 5 & 2 \\ \infty & -3 \end{bmatrix} \end{array}$$

$$Y \circ X = \begin{bmatrix} 2 & \infty \\ 5 & 2 \\ \infty & -3 \end{bmatrix} \begin{bmatrix} -1 & 2 \\ \infty & 3 \end{bmatrix} = \begin{array}{c} \\ c_1 \\ c_2 \\ c_3 \end{array} \begin{array}{c} \begin{array}{cc} a_1 & a_2 \end{array} \\ \begin{bmatrix} (2 \odot -1) \oplus (\infty \odot \infty) & (2 \odot 2) \oplus (\infty \odot 3) \\ (5 \odot -1) \oplus (2 \odot \infty) & (5 \odot 2) \oplus (2 \odot 3) \\ (\infty \odot -1) \oplus (-3 \odot \infty) & (\infty \odot 2) \oplus (-3 \odot 3) \end{bmatrix} \end{array}$$

Automata and $P(M)$.

What is the set of words accepted by the following NFA?




$\Sigma = \{b, c\}$

$P(\Sigma^*)$

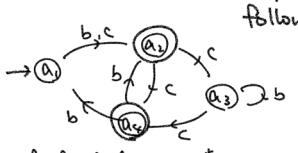
$X = \begin{array}{c} \\ q_1 \\ q_2 \\ q_3 \\ q_4 \end{array} \begin{array}{c} \begin{array}{cccc} q_1 & q_2 & q_3 & q_4 \end{array} \\ \begin{bmatrix} \emptyset & \emptyset & \emptyset & \{b\} \\ \{b,c\} & \emptyset & \emptyset & \{b\} \\ \emptyset & \{c\} & \{b\} & \{b\} \\ \emptyset & \{c\} & \{c\} & \emptyset \end{bmatrix} \end{array}$

$= (P(\Sigma^*), \underbrace{\emptyset}_{\text{zero}}, \underbrace{\{\varepsilon\}}_{\text{one}}, \underbrace{\cup}_{\oplus}, \underbrace{\cdot}_{\odot})$

$S \cdot T = \{vw \mid v \in S, w \in T\}$

Automata and $P(M)$.

What is the set of words accepted by the following NFA?




$\Sigma = \{b, c\}$

$X = \begin{array}{c} \\ q_1 \\ q_2 \\ q_3 \\ q_4 \end{array} \begin{array}{c} \begin{array}{cccc} q_1 & q_2 & q_3 & q_4 \end{array} \\ \begin{bmatrix} \emptyset & \emptyset & \emptyset & \{b\} \\ \{b,c\} & \emptyset & \emptyset & \{b\} \\ \emptyset & \{c\} & \{b\} & \{b\} \\ \emptyset & \{c\} & \{c\} & \emptyset \end{bmatrix} \end{array}$

$X^* = I \oplus X \oplus X^2 \oplus \cdots$

$I = \begin{bmatrix} \{\varepsilon\} & & \\ & \{\varepsilon\} & \\ & & \{\varepsilon\} \\ & & & \{\varepsilon\} \end{bmatrix}$

$\begin{bmatrix} \emptyset & \{\varepsilon\} & \emptyset & \{\varepsilon\} \end{bmatrix} X^* \begin{bmatrix} \{\varepsilon\} \\ \emptyset \\ \emptyset \\ \emptyset \end{bmatrix}$.

## §2. Quantales.

**Def.**

A quantale $Q$ consists of:

- a complete lattice $(Q, \leq)$
- a monoid $(Q, I, \circ)$

s.t. $y \circ (\bigvee_{i \in I} x_i) = \bigvee_{i \in I} (y \circ x_i)$

$(\bigvee_{i \in I} y_i) \circ x = \bigvee_{i \in I} (y_i \circ x)$.

**Rmk**

A quantale $Q$ gives rise to an idempotent semiring $(Q, \bot, I, \vee, \circ)$.

**Def.**

$P = (P, \leq_P)$, $Q = (Q, \leq_Q)$ : poset

An adjunction is a pair of functions

$$P \underset{u}{\overset{f}{\rightleftarrows}} Q$$

s.t. $\forall p \in P, \forall q \in Q,$

$$f(p) \leq_Q q \iff p \leq_P u(q).$$

$f$: left adjoint of $u$

$u$: right adjoint of $f$. $f \dashv u$.

**Rmk**

Left (right) adjoint of a function is unique if exists.

**Prop**

$L$ : complete lattice $P$: poset

$f: L \to P$ : function

$f$ has a right adjoint $u$

iff $f$ preserves arbitrary sups : $f(\bigvee_{i\in I} x_i) = \bigvee_{i\in I} f(x_i)$.

**Cor**

$Q = (Q, \leq, I, \circ)$ : quantale.

$\forall x \in Q$, $(-)\circ x$ has a right adjoint $(-)\swarrow x$.
(right extension along $x$)

$\forall y \in Q$, $y \circ (-)$ has a right adjoint $y \searrow (-)$.
(right lifting along $y$)

$$Q \underset{(-)\swarrow x}{\overset{(-)\circ x}{\rightleftarrows}} Q \qquad Q \underset{y\searrow(-)}{\overset{y\circ(-)}{\rightleftarrows}} Q.$$

$$y \leq z \swarrow x \iff y \circ x \leq z \iff x \leq y \searrow z.$$

In particular,

$$z \swarrow x \geq y \iff x \leq y \searrow z$$

$$Q \underset{(-)\searrow z}{\overset{z\swarrow(-)}{\rightleftarrows}} Q^{op} = (Q, \geq)$$

**Ex.**

1. The min-plus quantale $\overline{\mathbb{R}}_{\min,+} = ([-\infty, \infty], \geq, 0, +)$
   $\min = \bigvee$ wrt. $\geq$.
   $(-\infty) + \infty = \infty + (-\infty) = \infty$
   $y \searrow z = z \swarrow y = z - y$.
2. The max-plus quantale $\overline{\mathbb{R}}_{\max,+} = ([-\infty, \infty], \leq, 0, +)$
3. $M = (M, e, \cdot)$ : monoid (e.g. $\Sigma^*$)
   The free quantale over $M$ $\mathcal{P}(M) = (\mathcal{P}(M), \subseteq, \{\}, \cdot)$
   $S \cdot T = \{m \cdot n \mid m \in S, n \in T\}$

## §3. Q-matrices

**Def.**

$Q$ : quantale

$A, B$ : (possibly infinite) sets

A $Q$-matrix $X : A \nrightarrow B$ is a $(B \times A)$-indexed family $X = (X_{ba})_{b\in B, a\in A}$ of elements of $Q$.

$$X = \begin{matrix} \\ B \end{matrix} \overbrace{\begin{bmatrix} \vdots \\ \cdots X_{ba} \cdots \\ \vdots \end{bmatrix}}^{A}$$

**Def**

$X : A \nrightarrow B$, $Y : B \nrightarrow C$ : $Q$-matrix.

The composite $Y \circ X : A \nrightarrow C$ is given by

$$(Y \circ X)_{c,a} = \bigvee_{b \in B} (Y_{c,b} \circ X_{b,a}).$$

$$\left( = \bigoplus_{b \in B} (Y_{c,b} \otimes X_{b,a}) \right).$$

---

The set $Q\text{-Mat}(A,B) = \{Q\text{-matrices } A \nrightarrow B\}$ has an entrywise order $\leq_{A,B}$:

$$X \leq_{A,B} X' \iff \forall_{b \in B} \forall_{a \in A} \; X_{ba} \leq X'_{ba}.$$

$$\Rightarrow (Q\text{-Mat}(A,B), \leq_{A,B}) \cong Q^{B \times A} : \text{complete lattice.}$$

Composition preserves sups :

$$\forall_{X : A \nrightarrow B} \forall_{Y_i : B \nrightarrow C} \; \left(\bigvee_{i \in I} Y_i\right) \circ X = \bigvee_{i \in I} (Y_i \circ X).$$

$$\forall_{X_i : A \nrightarrow B} \forall_{Y : B \nrightarrow C} \; Y \circ \left(\bigvee_{i \in I} X_i\right) = \bigvee_{i \in I} (Y \circ X_i).$$

---

- $\forall_{X : A \nrightarrow B}$, $\forall_{\text{set } C}$,

$(-) \circ X : Q\text{-Mat}(B,C) \longrightarrow Q\text{-Mat}(A,C)$

has a right adjoint

$(-) \swarrow X : Q\text{-Mat}(A,C) \longrightarrow Q\text{-Mat}(B,C)$

(right extension along $X$) $\quad (Z \swarrow X)_{c,b} = \bigwedge_{a \in A} (Z_{c,a} \swarrow X_{b,a})$

- $\forall_{Y : B \nrightarrow C}$, $\forall_{\text{set } A}$.

$Y \circ (-) : Q\text{-Mat}(A,B) \longrightarrow Q\text{-Mat}(A,C)$

has a right adjoint

$Y \searrow (-) : Q\text{-Mat}(A,C) \to Q\text{-Mat}(A,B)$

(right lifting along $Y$) $\quad (Y \searrow Z)_{b,a} = \bigwedge_{c \in C} (Y_{c,b} \searrow Z_{c,a})$

---

**Ex.** (Legendre-Fenchel transform) [Willerton, 2015]

The Legendre-Fenchel transform.

$$\left(f : \mathbb{R}^n \to [-\infty, \infty]\right) \mapsto \left(f^* : \underset{\cup}{(\mathbb{R}^n)^*} \to [-\infty, \infty]\right)$$

$$\underset{\cup}{p} \mapsto \sup_{v \in \mathbb{R}^n} \{\langle p, v \rangle - f(v)\}.$$

$Q = \overline{\mathbb{R}}_{\min,+} = ([-\infty, \infty], \geq, 0, +)$

$\langle -, - \rangle : \mathbb{R}^n \nrightarrow (\mathbb{R}^n)^* : Q$-matrix.

$$\frac{f : \mathbb{R}^n \to [-\infty, \infty]}{f : \mathbb{R}^n \nrightarrow 1}$$

$f^* = \langle -, - \rangle \swarrow f.$ $\quad \left( \sup_{v \in \mathbb{R}^n} \{\langle p, v \rangle - f(v)\} = \bigwedge_{v \in \mathbb{R}^n} (\langle p, v \rangle \swarrow f(v)) \right)$

$\forall X : A \nrightarrow B,\ \forall Y : B \nrightarrow C.\ \forall Z : A \nrightarrow C.$

$Y \le_{B,C} Z \swarrow X \iff Y \circ X \le_{A,C} Z \iff X \le_{A,B} Y \searrow Z.$

In particular,

$$Z \swarrow X \ge_{B,C} Y \iff X \le_{A,B} Y \searrow Z.$$

$$\mathcal{Q}\text{-Mat}(A,B) \underset{(-)\searrow Z}{\overset{Z\swarrow(-)}{\rightleftarrows}} \mathcal{Q}\text{-Mat}(B,C)^{op}$$

## §4. The Isbell hull

In general, given an adjunction between posets

$$P \underset{u}{\overset{f}{\rightleftarrows}} Q$$

a fixed point of $f \dashv u$ is a pair

- $p \in P$
- $q \in Q$

s.t. $f(p) = q$ and $p = u(q)$.

$\mathrm{Fix}(f \dashv u) = \{\text{fixed points of } f \dashv u\}$.

**Def**

$Z : A \nrightarrow C$ : $\mathcal{Q}$-matrix.

The Isbell hull of $Z$ is the set of fixed points of the adjunction

$$\mathcal{Q}\text{-Mat}(A,1) \underset{(-)\searrow Z}{\overset{Z\swarrow(-)}{\rightleftarrows}} \mathcal{Q}\text{-Mat}(1,C)^{op}$$

$\mathrm{Isb}(Z) = \mathrm{Fix}(Z\swarrow(-) \dashv (-)\searrow Z)$.

$(X,Y) \in \mathrm{Isb}(Z) \iff$
- $X : A \nrightarrow 1$
- $Y : 1 \nrightarrow C$
- $Z \swarrow X = Y,\ X = Y \searrow Z.$

**Ex.**

Recall the Legendre-Fenchel transform. [Willerton, 2015]

$$\{\text{functions } \mathbb{R}^n \to [-\infty,\infty]\} \underset{(-)^*}{\overset{(-)^*}{\rightleftarrows}} \{\text{functions } (\mathbb{R}^n)^* \to [-\infty,\infty]\}$$

For $f : \mathbb{R}^n \to [-\infty,\infty]$,

$f$: lower semicontinuous convex
$\iff \exists g : (\mathbb{R}^n)^* \to [-\infty,\infty].\ f = g^*$
$\iff f = (f^*)^*$.

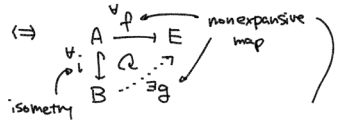
So $\mathrm{Isb}(\langle -,- \rangle) \cong \{\text{lower semicontinuous convex functions } \mathbb{R}^n \to [-\infty,\infty]\}$.
$\cong \{\text{"} \quad (\mathbb{R}^n)^* \to [-\infty,\infty]\}$.

**Ex.** (Directed tight span) [Willerton, 2013]

$(A, d_A)$: generalised metric space [Lawvere, 1973]

( $d_A : A \times A \to [0, \infty]$, $d_A(a,a) = 0$, $d_A(a,b) + d_A(b,c) \geq d_A(a,c)$ )

The **directed tight span** of $(A, d_A)$ is [Hirai & Kaichi 2012, Komajan, Kenzi & Otsuki 2012]
the "smallest" injective gms containing $(A, d_A)$.

( E: injective $\Leftrightarrow$ 

)

---

**Ex.** (Directed tight span) [Willerton, 2013]

$(A, d_A)$: generalised metric space

( $d_A : A \times A \to [0, \infty]$, $d_A(a,a) = 0$, $d_A(a,b) + d_A(b,c) \geq d_A(a,c)$ )

$\overline{\mathbb{R}}^{\geq 0}_{\min,+} = ([0,\infty], \geq, 0, +)$ : nonnegative min-plus quantale.

$\Rightarrow d_A : A \nrightarrow A : \overline{\mathbb{R}}^{\geq 0}_{\min,+}$-matrix.

$\mathrm{Isb}(d_A)$ (+ canonical metric)
= the directed tight span of $(A, d_A)$.

---

**Ex.** (Tropical polytope) [Elliott, 2017]

$v_1, \cdots, v_r \in [-\infty, \infty]^n$

The (unprojectivised) **tropical polytope** generated by [Develin & Sturmfels, 2004]
$v_1, \cdots, v_r$ is the set of points $x \in [-\infty, \infty]^n$ s.t.

$$x = \max\{c_1 + v_1, \cdots, c_r + v_r\}$$
$$= (c_1 \odot v_1) \oplus \cdots \oplus (c_r \odot v_r)$$

for some $c_i \in [-\infty, \infty]$.

---

**Ex.** (Tropical polytope) [Elliott, 2017]

r points $v_1, \cdots, v_r \in [-\infty, \infty]^n$

$\overline{\mathbb{R}}_{\min,+}$-matrix $Z : r \nrightarrow n$.

$$Z = \begin{matrix} r \\ \{ \end{matrix} \underbrace{\begin{bmatrix} v_1 \\ \vdots \\ v_r \end{bmatrix}}_{n}$$

**Prop.** [Elliott, 2017]

The (unprojectivised) tropical polytope generated by $v_1, \cdots, v_r$ is $\mathrm{Isb}(Z)$.

**Ex.** (Tropical polytope) [Elliott, 2017]

**Thm.** [Develin & Sturmfels, 2004]

There is a correspondence between

tropical polytopes in $[-\infty,\infty]^n$ generated by $r$ points

and

tropical polytopes in $[-\infty,\infty]^r$ generated by $n$ points.

$r$ points in $[-\infty,\infty]^n$ — $n$ points in $[-\infty,\infty]^r$

$Z : r \to n \quad \longleftrightarrow \quad Z' : n \to r$

1-1 correspondence via transposition $(-)^T$

$\mathrm{Isb}(Z) \cong \mathrm{Isb}(Z^T)$.

## §5. Q-categories

**Def**

$Q$: quantale

A **Q-category** is a $Q$-matrix $\mathcal{C} : \mathrm{ob}(\mathcal{C}) \nrightarrow \mathrm{ob}(\mathcal{C})$

s.t. $\quad I_{\mathrm{ob}(\mathcal{C})} \le \mathcal{C}$

$\mathcal{C} \circ \mathcal{C} \le \mathcal{C}$.

A $Q$-category consists of:

- Set $\mathrm{ob}(\mathcal{C})$ of objects.
- $\forall c, c' \in \mathrm{ob}(\mathcal{C})$, an element $\mathcal{C}(c,c') \in Q$

s.t.

- $\forall c \in \mathrm{ob}(\mathcal{C}).\ I \le \mathcal{C}(c,c)$
- $\forall c, c', c'' \in \mathrm{ob}(\mathcal{C}).\ \mathcal{C}(c',c'') \circ \mathcal{C}(c,c') \le \mathcal{C}(c,c'')$

**Ex.**

1. $Q = 2 = (\{0,1\}, \le, 1, \wedge)$.

A 2-category $\mathcal{C}$ has ...

- a set $\mathrm{ob}(\mathcal{C})$.
- a function $\mathcal{C} : \mathrm{ob}(\mathcal{C}) \times \mathrm{ob}(\mathcal{C}) \to \{0,1\}$ / $R \subseteq \mathrm{ob}(\mathcal{C}) \times \mathrm{ob}(\mathcal{C})$.

s.t.

- $\forall c \in \mathcal{C}.\quad 1 \le \mathcal{C}(c,c)$ / $c \mathrel{R} c$
- $\forall c, c', c'' \in \mathcal{C}.\quad \mathcal{C}(c',c'') \wedge \mathcal{C}(c,c') \le \mathcal{C}(c,c'')$ / $c' R c'' \wedge c R c' \Rightarrow c R c''$

$\Rightarrow$ 2-category = preordered set.

**Ex.**

2. $Q = \overline{\mathbb{R}}^{\geq 0}_{\min,+} = ([0,\infty], \ge, 0, +)$.

An $\overline{\mathbb{R}}^{\geq 0}_{\min,+}$-category $\mathcal{C}$ has ...

- a set $\mathrm{ob}(\mathcal{C})$
- a function $\mathcal{C} : \mathrm{ob}(\mathcal{C}) \times \mathrm{ob}(\mathcal{C}) \to [0,\infty]$

s.t.

- $\forall c \in \mathcal{C}.\quad 0 \ge \mathcal{C}(c,c)$ $\left(\Leftrightarrow 0 = \mathcal{C}(c,c)\right)$
- $\forall c, c', c'' \in \mathcal{C}.\quad \mathcal{C}(c',c'') + \mathcal{C}(c,c') \ge \mathcal{C}(c,c'')$ (triangle inequality)

$\Rightarrow$ Every metric space gives rise to an $\overline{\mathbb{R}}^{\geq 0}_{\min,+}$-category [Lawvere, 1973]

$\mathcal{C}$ : $Q$-category

$\Rightarrow$ Define a relation $\leq_{\mathcal{C}} \subseteq ob(\mathcal{C}) \times ob(\mathcal{C})$ by

$$c \leq_{\mathcal{C}} c' \iff I \leq_Q \mathcal{C}(c,c').$$

Then $\leq_{\mathcal{C}}$ : preorder.

**Def.**

A $Q$-category $\mathcal{C}$ is called

- skeletal iff $\leq_{\mathcal{C}}$ : partial order
- complete iff $(\mathcal{C}, \leq_{\mathcal{C}})$ has all sups and $\mathcal{C}$ : tensored

$$\left(\forall c \in \mathcal{C}.\ \forall x \in Q.\ \exists x \otimes c \in \mathcal{C} \right.$$
$$\left. \text{s.t.}\ \forall d \in \mathcal{C}.\ \mathcal{C}(x \otimes c, d) = x \searrow \mathcal{C}(c,d) \right)$$

**Ex.**

$Q = 2 = (\{0,1\}, \leq, 1, \wedge)$

A 2-category $\mathcal{C}$ is skeletal

iff it is a partially ordered set.

" is skeletal and complete

iff it is a complete lattice.

**Def.**

$\mathcal{C}, \mathcal{D}$ : $Q$-category

A $Q$-functor $f : \mathcal{C} \to \mathcal{D}$ is a function $f : ob(\mathcal{C}) \to ob(\mathcal{D})$ s.t.

- $\forall c, c' \in \mathcal{C}.\quad \mathcal{C}(c,c') \leq \mathcal{D}(fc, fc')$.

A $Q$-functor $f : \mathcal{C} \to \mathcal{D}$ is fully faithful iff

- $\forall c, c' \in \mathcal{C}.\quad \mathcal{C}(c,c') = \mathcal{D}(fc, fc')$.

**Ex.**

- 2-functor = monotone map.
- $\overline{\mathbb{R}}^{op}_{min,+}$-functor = nonexpansive map.

**Thm.** [Banaschewski & Bruns, 1967]

A preorder (= 2-category) is a complete lattice (= skeletal and complete) iff it is injective.

$\mathcal{E}$ : injective $\iff$ $\forall f$: $A \to E$, $\forall i$: embedding $A \to B$, $\exists g$: $B \to E$

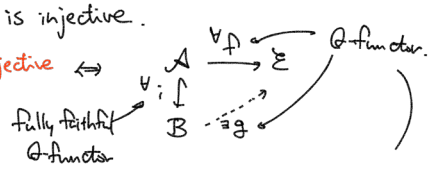
(monotone map (= 2-functor))

(= fully faithful 2-functor.)

**Thm.**

$Q$: quantale

A $Q$-category is skeletal and complete iff it is injective.

($\mathcal{E}$: injective $\Leftrightarrow$ 

)

---

$Q$-category $\mathcal{E}$

$Q$-matrix $\mathcal{E}: ob(\mathcal{E}) \nrightarrow ob(\mathcal{E})$

Isbell hull $Isb(\mathcal{E})$

↑ admits a $Q$-category str. and a fully faithful $Q$-functor $J_{\mathcal{E}}: \mathcal{E} \to Isb(\mathcal{E})$

**Prop.**

For any $Q$-category $\mathcal{E}$, $Isb(\mathcal{E})$ is the injective hull [Adámek, Herrlich, Rosický & Tholen, 2002] of $\mathcal{E}$.

## 順序と半群とQuantaleの関係性について

安田康史(神奈川大学)

2019/08/31
ALGI at 九州大学

1

## 背景

- 定義:
  Unital Quantale とは以下を満たす
  $(Q,\leq,\vee,\odot,1)$
  - $(Q,\leq,\vee)$ は完備上半束
  - $(Q,\odot,1)$ はモノイド
  - Qの任意の部分集合SとQの任意の要素aに対し $(\vee S)\odot a=\vee\{b\odot a \mid b\in S\}$
  - Qの任意の部分集合SとQの任意の要素aに対し $a\odot(\vee S)=\vee\{a\odot b \mid b\in S\}$

2

## 背景

- Unital Quantale の例:$(Rel(A),\leq,\vee,\odot,1)$
  - Rel(A) は集合A上の二項関係の全体
  - $\leq$は二項関係どうしの包含関係
  - $\vee$は和集合
  - $\odot$は二項関係の合成
  - 1 は恒等関係
- ここでは、この例を関係的quantaleと呼ぶ。

3

## Unital Quantale から 関係的 Quantale への埋め込み

| | $\odot$を合成演算に移す | 1を恒等関係に移す | non-involutive な Quantale にも適用可能 |
|---|---|---|---|
| Brown and Gurr 1993 | ○ | × | ○ |
| Palmigiano and Re, 2011 | ○ | ○ | × |
| 西澤・古澤 2012 | ○ | ○ | ○ |

4

## Unital Quantale から 関係的 Quantale への埋め込み

| | ⊙を合成演算に移す | 1を恒等関係に移す | non-involutive な Quantale にも適用可能 |
|---|---|---|---|
| Brown and Gurr 1993 | ○ | × | ○ |
| Palmigiano and Re, 2011 | ○ | ○ | × |
| 西澤・古澤 2012 | ○ | ○ | ○ |

ただし、
べき集合
Quantaleに
対してのみ
適用可能

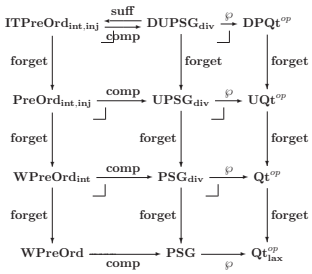
5

## 本研究の目的

- [西澤（神奈川大学）・古澤（鹿児島大学） 2012] で、
  べき集合 Quantale Q から
  関係的 Quantale への埋め込みが
  なぜ得られたのか、
  理由を明らかにしたい

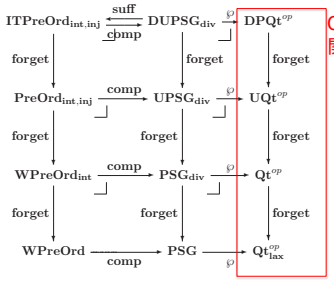
6

## 西澤・古澤・安田により 今のところ得られている結果




7

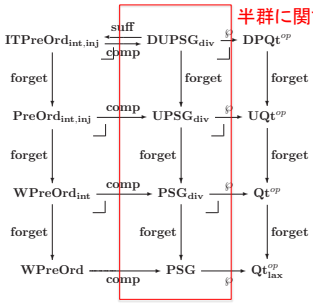
## 西澤・古澤・安田により 今のところ得られている結果




Quantale に
関する圏たち

8

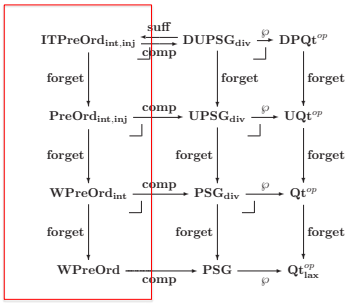
## 西澤・古澤・安田により
## 今のところ得られている結果




9

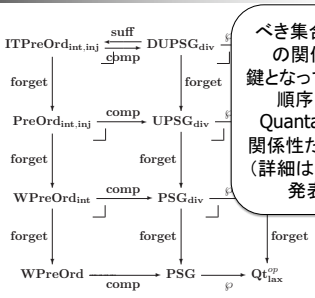
## 西澤・古澤・安田により
## 今のところ得られている結果

順序に関する圏たち




10

## 西澤・古澤・安田により
## 今のところ得られている結果

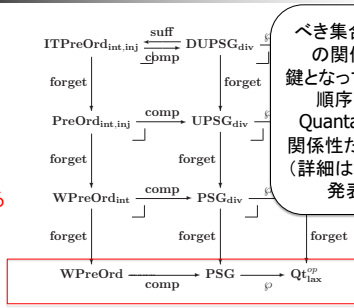



べき集合Quantaleの関係表現の鍵となっているのは、順序と半群とQuantale の間の関係性だと思われる（詳細は次の西澤の発表にて）

11

## 西澤・古澤・安田により
## 今のところ得られている結果

安田による本発表で説明する部分




べき集合Quantaleの関係表現の鍵となっているのは、順序と半群とQuantale の間の関係性だと思われる（詳細は次の西澤の発表にて）

12

## 本発表の目次

- べき集合Quantaleの関係表現の鍵となる、順序と半群とQuantale の関係性の一部を証明
  1. Quantale を対象とし、lax homomorphism を射とする圏を定義
  2. Partial semigroup と、それを対象とする圏を定義
  3. 2. の圏から 1. の圏への反変関手を定義
  4. Weak preorder と、それを対象とする圏を定義
  5. 4. の圏から 2. の圏への共変関手を定義

13

## 本発表の目次

- べき集合Quantaleの関係表現の鍵となる、順序と半群とQuantale の関係性の一部を証明
  1. Quantale を対象とし、lax homomorphism を射とする圏を定義
  2. Partial semigroup と、それを対象とする圏を定義
  3. 2. の圏から 1. の圏への反変関手を定義
  4. Weak preorder と、それを対象とする圏を定義
  5. 4. の圏から 2. の圏への共変関手を定義

14

## Quantale

- 定義

  Quantaleとは以下のことを満たす$(Q,\leq,\vee,\odot)$
  - $(Q,\odot)$が半群
  - $(Q,\leq,\vee)$が完備上半束
  - Qの任意の部分集合SとQの任意の要素qに対し $(\vee S)\odot q = \vee\{s\odot q \mid s \in S\}$を満たす.
  - Qの任意の部分集合SとQの任意の要素qに対し $q\odot(\vee S) = \vee\{q\odot s \mid s \in S\}$を満たす.

15

## $Qt_{lax}$

- 定義

  $Qt_{lax}$とは以下のものからなる圏
  - objectはquantale $(Q,\leq,\vee,\odot)$
  - $(Q,\leq,\vee,\odot)$ から $(Q',\leq',\vee',\odot')$へのarrowは lax homomorphism $f : Q\to Q'$.すなわち
    1. Qの任意の部分集合Sに対し $f(\vee S) = \vee'\{f(s)|s \in S\}$.
    2. Qの任意の要素$q_1$, $q_2$に対し $f(q_1 \odot q_2) \geq f(q_1) \odot' f(q_2)$.
  - arrowの合成$\circ$は写像の合成
  - idは恒等写像

16

## 本発表の目次

- べき集合Quantaleの関係表現の鍵となる、順序と半群とQuantale の関係性の一部を証明
  1. Quantale を対象とし、lax homomorphism を射とする圏を定義
  2. Partial semigroup と、それを対象とする圏を定義
  3. 2. の圏から 1. の圏への反変関手を定義
  4. Weak preorder と、それを対象とする圏を定義
  5. 4. の圏から 2. の圏への共変関手を定義

17

## Partial semigroup

- 定義

  partial semigroupとは以下のことを満たす(X,・)
  - Xは集合
  - ・はX上のpartial binary function
  - x・y と (x・y)・z が定義される
    ⇔ y・z と x・(y・z) が定義される
  - (x・y)・z = x・(y・z)

18

## PSG

- 定義

  PSGとは以下のものからなる圏
  - objectはpartial semigroup(X,・)
  - (X,・) から (X′,・′) へのarrowは,写像 f : X → X′ で
    x・yが定義されているようなx,yについて,
    f(x)・′f(y) が定義され、かつ
    f(x)・′f(y) = f(x・y) となるもの
  - arrowの合成。は写像の合成
  - idは恒等写像

19

## 本発表の目次

- べき集合Quantaleの関係表現の鍵となる、順序と半群とQuantale の関係性の一部を証明
  1. Quantale を対象とし、lax homomorphism を射とする圏を定義
  2. Partial semigroup と、それを対象とする圏を定義
  3. 2. の圏から 1. の圏への反変関手を定義
  4. Weak preorder と、それを対象とする圏を定義
  5. 4. の圏から 2. の圏への共変関手を定義

20

## $PSG^{op} \to Qt_{lax}$

- 定理
  以下のデータPが$PSG^{op}$から$Qt_{lax}$への関手となる
  - object(X,・)に対し, P(X,・)$\stackrel{def}{=}$(P(X),⊆,∪,⊙)
    ここで
    $S_1 \odot S_2 \stackrel{def}{=} \{s_1 \cdot s_2 \mid s_1 \in S_1,\ s_2 \in S_2,\ s_1 \cdot s_2$が定義される$\}$
  - arrow g : (X,・)→(X′,・′) in PSG に対し,
    P(g):P(X′,・′)→P(X,・)はSを$g^{-1}(S) = \{x \in X \mid f(x) \in S\}$に送る写像



定義(P: $PSG^{op} \to Qt_{lax}$)
- object:
  P(X,・)$\stackrel{def}{=}$(P(X),⊆,∪,⊙)
  $S_1 \odot S_2 \stackrel{def}{=}$
  $\{s_1 \cdot s_2 \mid s_1 \in S_1,\ s_2 \in S_2,\ s_1 \cdot s_2$ が定義される$\}$
- arrow:
  g : (X′,・′)→(X,・) in PSG に対し
  P(g):P(X,・)→P(X′,・′) は
  $P(g)(S) = g^{-1}(S) = \{x' \in X' \mid g(x') \in S\}$

定義($Qt_{lax}$のarrow).
f : (Q,≤,∨,⊙) → (Q′,≤′,∨′,⊙′) は lax homomorphism すなわち
1. Qの任意の部分集合 A に対し f(∨A) = ∨′{f(a)|a ∈ A}.
2. Qの任意の要素$q_1$, $q_2$に対し $f(q_1 \odot q_2) \geq f(q_1) \odot' f(q_2)$.

- (任意の g in PSG に対し P(g) が lax homomorphism)
  1. P(X) の任意の部分集合 A に対し
     P(g)(∪A)
     $= g^{-1}(\cup A)$ (Pの定義により)
     $= \{x' \in X' \mid g(x') \in \cup A\}$ ($g^{-1}$ の定義により)
     $= \{x' \in X' \mid$ ある $S \in A$ が存在して $g(x') \in S\}$
     $= \cup\{\{x' \in X' \mid g(x') \in S\} \mid S \in A\}$
     $= \cup\{g^{-1}(S) \mid S \in A\}$ ($g^{-1}$ の定義により)
     $= \cup\{P(g)(S) \mid S \in A\}$ (Pの定義により)



定義(P: $PSG^{op} \to Qt_{lax}$)
- object:
  P(X,・)$\stackrel{def}{=}$(P(X),⊆,∪,⊙)
  $S_1 \odot S_2 \stackrel{def}{=}$
  $\{s_1 \cdot s_2 \mid s_1 \in S_1,\ s_2 \in S_2,\ s_1 \cdot s_2$ が定義される$\}$
- arrow:
  g : (X′,・′)→(X,・) in PSG に対し
  P(g):P(X,・)→P(X′,・′) は
  $P(g)(S) = g^{-1}(S) = \{x' \in X' \mid g(x') \in S\}$

定義($Qt_{lax}$のarrow).
f : (Q,≤,∨,⊙) → (Q′,≤′,∨′,⊙′) は lax homomorphism すなわち
1. Qの任意の部分集合 A に対し f(∨A) = ∨′{f(a)|a ∈ A}.
2. Qの任意の要素$q_1$, $q_2$に対し $f(q_1 \odot q_2) \geq f(q_1) \odot' f(q_2)$.

- (任意の g in PSG に対し P(g) が lax homomorphism)
  2. P(X) の任意の要素 $S_1$, $S_2$ に対し
     $P(g)(S_1 \odot S_2) \supseteq P(g)(S_1) \odot' P(g)(S_2)$を示す.
     $x' \in P(g)(S_1) \odot' P(g)(S_2)$をとる.
     $s_1' \in P(g)(S_1)$ と $s_2' \in P(g)(S_2)$ が存在し $x' = s_1' \cdot' s_2'$ である.
     P(g)の定義により、$g(s_1') \in S_1$ と $g(s_2') \in S_2$ を満たす.
     $g(x') = g(s_1' \cdot' s_2')$ ($x' = s_1' \cdot' s_2'$ により)
     $= g(s_1') \cdot g(s_2')$ (g in PSG により)
     $\in S_1 \odot S_2$ (⊙の定義により)
     よって P(g)の定義により、$x' \in P(g)(S_1 \odot S_2)$である.



定義(P: $PSG^{op} \to Qt_{lax}$)
- object:
  P(X,・)$\stackrel{def}{=}$(P(X),⊆,∪,⊙)
  $S_1 \odot S_2 \stackrel{def}{=}$
  $\{s_1 \cdot s_2 \mid s_1 \in S_1,\ s_2 \in S_2,\ s_1 \cdot s_2$ が定義される$\}$
- arrow:
  g : (X′,・′)→(X,・) in PSG に対し
  P(g):P(X,・)→P(X′,・′) は
  $P(g)(S) = g^{-1}(S) = \{x' \in X' \mid g(x') \in S\}$

- (Pが合成を保つ)
  h : (X,・) → (X′,・′) in PSG , k : (X′,・′) → (X″,・″) in PSGをとり,
  P(k ∘ h) = P(h) ∘ P(k)となることを示す.
  S″をP(X″)の任意の要素とする.
  P(k ∘ h)(S″)
  $= (k \circ h)^{-1}(S'')$ (Pの定義より)
  $= \{x \in X \mid (k \circ h)(x) \in S''\}$ ($(k \circ h)^{-1}$の定義より)
  $= \{x \in X \mid k(h(x)) \in S''\}$ (写像の合成より)
  $= \{x \in X \mid h(x) \in \{x' \in X' \mid k(x') \in S''\}$ (自明)
  $= h^{-1}(\{x' \in X' \mid k(x') \in S''\})$ ($h^{-1}$の定義より)
  $= P(h)(\{x' \in X' \mid k(x') \in S''\})$ (Pの定義より)
  $= P(h)(k^{-1}(S''))$ ($k^{-1}$の定義より)
  $= P(h)(P(k)(S''))$ (Pの定義より)
  $= (P(h) \circ P(k))(S'')$ (合成の定義より)

定義 (P: $PSG^{op} \to Qt_{lax}$)
- object:
  P(X,・)≝(P(X),⊆,∪,⊙)
  $S_1 \odot S_2$≝
  $\{s_1 \cdot s_2 \mid s_1 \in S_1,\ s_2 \in S_2,\ s_1 \cdot s_2$ が定義される$\}$
- arrow:
  g : (X′,・′)→(X,・) in PSG に対し
  P(g):P(X,・)→P(X′,・′) は
  $P(g)(S) = g^{-1}(S) = \{x' \in X' \mid g(x') \in S\}$

■ (Pがidを保つ)
任意の (X,・) ∈ PSG について、$P(id_{(X,\cdot)}) = id_{P(X,\cdot)}$ を示す.
SはP(X)の任意の要素とする.

| | |
|---|---|
| $P(id_{(X,\cdot)})(S)$ | |
| $= (id_{(X,\cdot)})^{-1}(S)$ | (Pの定義より) |
| $= \{x \in X \mid id_{(X,\cdot)}(x) \in S\}$ | ($(id_{(X,\cdot)})^{-1}$の定義より) |
| $= \{x \in X \mid x \in S\}$ | (恒等写像より) |
| $= S$ | (自明) |
| $= id_{P(X,\cdot)}(S)$ | (恒等写像より) |

25

## 本発表の目次

- べき集合Quantaleの関係表現の鍵となる、順序と半群とQuantale の関係性の一部を証明
  1. Quantale を対象とし、lax homomorphism を射とする圏を定義
  2. Partial semigroup と、それを対象とする圏を定義
  3. 2. の圏から 1. の圏への反変関手を定義
  4. Weak preorder と、それを対象とする圏を定義
  5. 4. の圏から 2. の圏への共変関手を定義

26

## Weak preorder

- 定義
  X上のweak preorderとは,
  X上の推移的関係Rであり,
  Xの任意の要素 x に対して y が存在し,
  (x,y)∈R または (y,x)∈R を満たしているものとする.
- 定義
  集合Xと,X上のweak preorder Rの組(X,R)を
  weak preordered setという.

27

## WPreOrd

- 定義
  WPreOrdとは以下のものからなる圏
  - objectはweak preordered set
  - (X,R)から(X′,R′)へのarrowは単調写像
    すなわちf: X→X′で(x,y)∈Rのとき,(f(x),f(y))∈ R′.
  - arrowの合成 ∘ は写像の合成
  - idは恒等写像

28

## 本発表の目次

- べき集合Quantaleの関係表現の鍵となる、順序と半群とQuantale の関係性の一部を証明
  1. Quantale を対象とし、lax homomorphism を射とする圏を定義
  2. Partial semigroup と、それを対象とする圏を定義
  3. 2. の圏から 1. の圏への反変関手を定義
  4. Weak preorder と、それを対象とする圏を定義
  5. 4. の圏から 2. の圏への共変関手を定義

29

## WPreOrd → PSG

- 定理
  以下のデータcompがWPreOrdからPSGへの関手となる
  - object(X,R)に対し,comp(X,R)≅(R,;).
    ここで(w,x);(y,z)はx=yの時だけ定義され (w, x);(y, z) = (w, z).
  - arrow g : (X,R) → (X′,R′) in WPreOrd に対し,
    comp(g) : (R,;) → (R′,;) は(x,y) を(g(x),g(y))に送る写像.

30

定義（comp: WPreOrd → PSG ）
- object:
  (X,R)に対し,comp(X,R)≅(R,;).
  ここで(w,x);(y,z)はx=yの時だけ定義され
  (w, x);(y, z) = (w, z).
- arrow:
  g : (X,R) → (X′,R′) in WPreOrd に対し,
  comp(g) : (R, ;) → (R′, ;) は
  comp(g)(x, y) = (g(x),g(y)) に送る写像.

定義(PSGのarrow)
(X,・)から(X′,・′)へのarrowは,
1. 写像 f : X → X′であり,
2. x・yが定義されているような
   x, yについて,
   ① f(x) ・′ f(y) が定義され、
   ② f(x) ・′ f(y)=f(x・y) となる.

- (任意のg in WPreOrdに対しcomp(g)がPSGのarrow)
  g : (X, R) → (X′, R′) in WPreOrdをとる.
  comp(g) がPSGの(R, ;)から(R′, ;) へのarrowになることを示す.
  1. comp(g)がRからR′への写像であることを示す. (x,y) ∈ Rをとる.
     comp(g)(x,y) = (g(x),g(y)) (compの定義より)
     ∈ R′ (gの単調性より)

31

定義（comp: WPreOrd → PSG ）
- object:
  (X,R)に対し,comp(X,R)≅(R,;).
  ここで(w,x);(y,z)はx=yの時だけ定義され
  (w, x);(y, z) = (w, z).
- arrow:
  g : (X,R) → (X′,R′) in WPreOrd に対し,
  comp(g) : (R, ;) → (R′, ;) は
  comp(g)(x, y) = (g(x),g(y)) に送る写像.

定義(PSGのarrow)
(X,・)から(X′,・′)へのarrowは,
1. 写像 f : X → X′であり,
2. x・yが定義されているような
   x, yについて,
   ① f(x) ・′ f(y) が定義され、
   ② f(x) ・′ f(y)=f(x・y) となる.

- (任意のg in WPreOrdに対しcomp(g)がPSGのarrow)
  g : (X, R) → (X′, R′) in WPreOrdをとる.
  comp(g) がPSGの(R, ;)から(R′, ;) へのarrowになることを示す.
  2. (w,x);(y,z) が定義されるような (w,x),(y,z) ∈ Rをとる.
     ① (w,x);(y,z) が定義されることより、x = y が成り立つ. comp の定義より、それぞれ comp(g)(w,x) = (g(w),g(x)) および comp(g)(y,z) = (g(y),g(z)) = (g(x),g(z)) である. ; の定義から, (g(w),g(x)) ; (g(x),g(z)) は定義される.
     よって, comp(g)(w,x) ; comp(g)(y,z) が定義される.
     ② comp(g)(w,x) ; comp(g)(y,z)
     = (g(w),g(x)) ; (g(x),g(z)) (①より)
     = (g(w), g(z)) (; の定義より)
     = comp(g)(w,z) (comp の定義より)
     = comp(g)((w,x) ; (y,z))
     よって, comp(g)(w,x) ; comp(g)(y,z) = comp(g)((w,x) ; (y,z)) となる.

32

定義（comp: WPreOrd → PSG ）
- object:
  (X,R)に対し,comp(X,R)≅(R,;).
  ここで(w,x);(y,z)はx=yの時だけ定義され
  (w, x);(y, z) = (w, z).
- arrow:
  g : (X,R) → (X′,R′) in WPreOrd に対し,
  comp(g) : (R, ;) → (R′, ;) は
  comp(g)(x, y) = (g(x),g(y)) に送る写像.

■ (compが合成を保つ)
p : (X, R) → (X′, R′) in WPreOrd,
q : (X′, R′) → (X″, R″) in WPreOrd をとり、
comp(q ∘ p) = comp(q) ∘ comp(p) となることを示す.

comp(q ∘ p)(x, y)
= ((q ∘ p)(x),(q ∘ p)(y)) (compの定義より)
= (q(p(x)), q(p(y))) (写像の合成より)
= comp(q)(p(x), p(y)) (compの定義より)
= comp(q)(comp(p)(x, y)) (compの定義より)
= (comp(q) ∘ comp(p))(x, y) (写像の合成より)

33

定義（comp: WPreOrd → PSG ）
- object:
  (X,R)に対し,comp(X,R)≅(R,;).
  ここで(w,x);(y,z)はx=yの時だけ定義され
  (w, x);(y, z) = (w, z).
- arrow:
  g : (X,R) → (X′,R′) in WPreOrd に対し,
  comp(g) : (R, ;) → (R′, ;) は
  comp(g)(x, y) = (g(x),g(y)) に送る写像.

■ (compがidを保つ)
任意の (X, R) ∈ WPreOrd について,
$comp(id_{(X, R)}) = id_{comp(X, R)}$ を示す.
(x, y)はRの任意の要素とする.

$comp(id_{(X, R)})(x, y)$
$= (id_{(X, R)}(x), id_{(X, R)}(y))$ (compの定義より)
$= (x, y)$ (恒等写像より)
$= id_{comp(X, R)}(x, y)$ (恒等写像より)

34

## 本発表のまとめ

- べき集合Quantaleの関係表現の鍵となる、順序と半群とQuantale の関係性の一部を証明
  1. Quantale を対象とし、lax homomorphism を射とする圏を定義
  2. Partial semigroup と、それを対象とする圏を定義
  3. 2. の圏から 1. の圏への反変関手を定義
  4. Weak preorder と、それを対象とする圏を定義
  5. 4. の圏から 2. の圏への共変関手を定義

35

# Correspondences among classes of weak preorders, partial semigroups, and quantales

Koki Nishizawa (Kanagawa University)

2019/08/31

ALGI at Kyushu University



## Background [Nishizawa Furusawa 2012]

- Def.
  A unital quantale Q is called powerset quantale, if its underlying complete join semilattice is isomorphic to the powerset of some set.
- E.g. A powerset as a frame
  - $(P(X),\subseteq,\cup,\cap,X)$
- Th.(Relational representation)
  - For a powerset quantale Q, there exists an injective homomorphism $\eta: Q \to P(\leqslant)$ of unital quantales for some preorder $\leqslant$ on some set X.

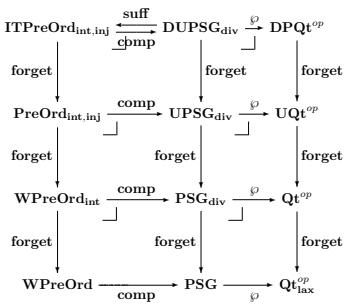


## Background [Nishizawa Furusawa 2012]

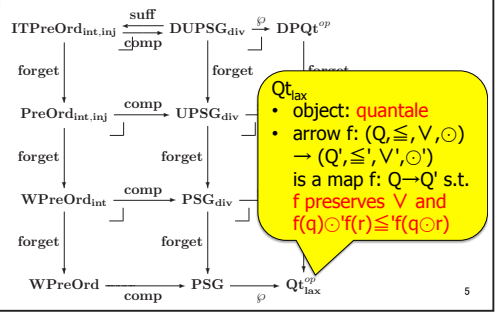
- The leading example of our relational representation
  $\eta : (P(Nat),\subseteq,\cup,\oplus,\{0\}) \to (P(\leqslant_{Nat}),\subseteq,\cup,[;],id)$
  $\eta(S)=\{(m,n) \mid m \leqslant_{Nat} n,\ n-m \in S\}$
  - $R \oplus S=\{r+s \mid r\in R,\ s\in S\}$
  - $R[;]S=\{(x,z) \mid \exists y.\ (x,y)\in R,\ (y,z)\in S\}$
- Future goal: To give a dual equivalence between
  - the category of powerset quantales
  - the category of some class of preorders
- Approach: quantales induced by partial semigroups
  - $\eta : P(Nat) \to P(\leqslant_{Nat})$ as $\text{minus}^{-1}$ of minus : $(\leqslant_{Nat}, ;) \to (Nat, +)$



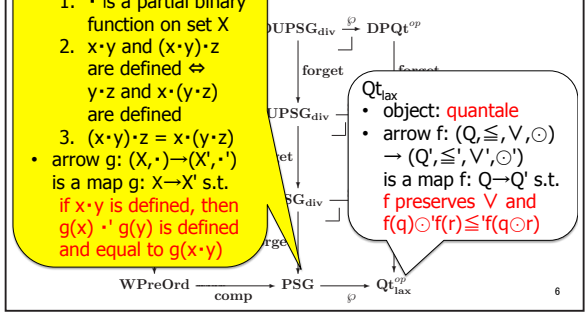
## Today's Results (joint work with Yasuda, Furusawa)



## lax homos, partial semigroups, and weak preordered sets

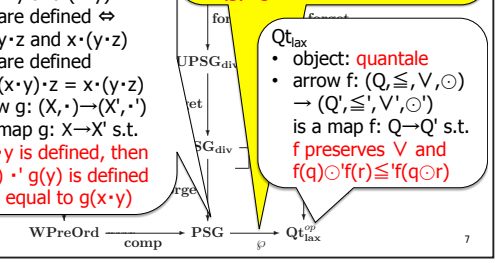



PSG
- object:
  partial semigroup, i.e., (X,・) s.t.
  1. ・ is a partial binary function on set X
  2. x・y and (x・y)・z are defined ⇔ y・z and x・(y・z) are defined
  3. (x・y)・z = x・(y・z)
- arrow g: (X,・)→(X',・') is a map g: X→X' s.t.
  if x・y is defined, then g(x) ・' g(y) is defined and equal to g(x・y)



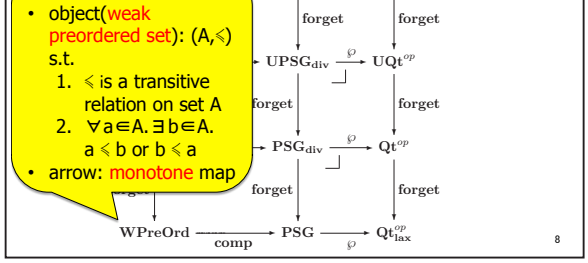

PSG
- object:
  partial semigroup, i.e., (X,・) s.t.
  1. ・ is a partial binary function on set X
  2. x・y and (x・y)・z are defined ⇔ y・z and x・(y・z) are defined
  3. (x・y)・z = x・(y・z)
- arrow g: (X,・)→(X',・') is a map g: X→X' s.t.
  if x・y is defined, then g(x) ・' g(y) is defined and equal to g(x・y)

P: PSG$^{op}$ → Qt$_{lax}$
- object:
  P(X,・)=(P(X),⊆,∪,[・])
  where S[・]T=
  {s・t | s∈S,t∈T, s・t is defined}
- arrow:
  P(g)=g$^{-1}$




## lax homos, partial semigroups, and weak preordered sets

WPreOrd
- object(weak preordered set): (A,⩽) s.t.
  1. ⩽ is a transitive relation on set A
  2. ∀a∈A. ∃b∈A. a ⩽ b or b ⩽ a
- arrow: monotone map

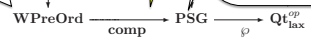


## lax hom… …ups, and we…

comp: WPreOrd→PSG
- object: comp(A,⩽)=(⩽, ;) where (a,b);(c,d) is defined and equal to (a,d) when b=c
- arrow: comp(h)(a,b)=(h(a),h(b))

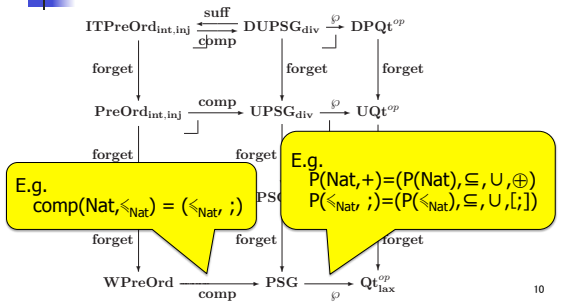
WPreOrd
- object(weak preordered set): (A,⩽) s.t.
  1. ⩽ is a transitive relation on set A
  2. ∀a∈A. ∃b∈A. a ⩽ b or b ⩽ a
- arrow: monotone map

PSG
- object: partial semigroup
- arrow g: (X,・)→(X',・') is a map g: X→X' s.t. if x・y is defined, then g(x) ・' g(y) is defined and equal to g(x・y)




9

## lax homos, partial semigroups, and weak preordered sets

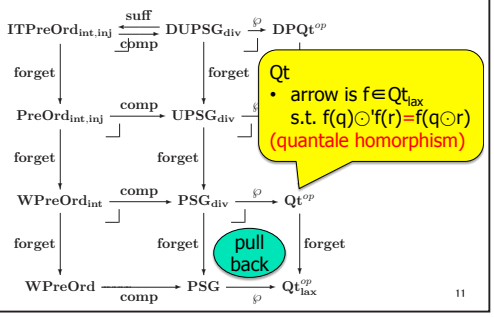



E.g.
comp(Nat,⩽Nat) = (⩽Nat, ;)

E.g.
P(Nat,+)=(P(Nat),⊆,∪,⊕)
P(⩽Nat, ;)=(P(⩽Nat),⊆,∪,[;])

10

## quantale homomorphisms and dividing maps




Qt
- arrow is f∈Qt_lax s.t. f(q)⊙'f(r)=f(q⊙r)

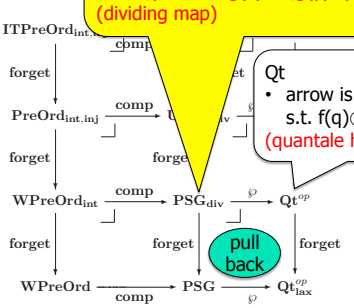
(quantale homorphism)

11

## quanta… dividin…

PSG_div
- arrow is g: (X,・)→(X',・') in PSG s.t. if x' ・' y' = g(z), then ∃x,y∈X s.t. g(x)=x',g(y)=y',and x・y=z

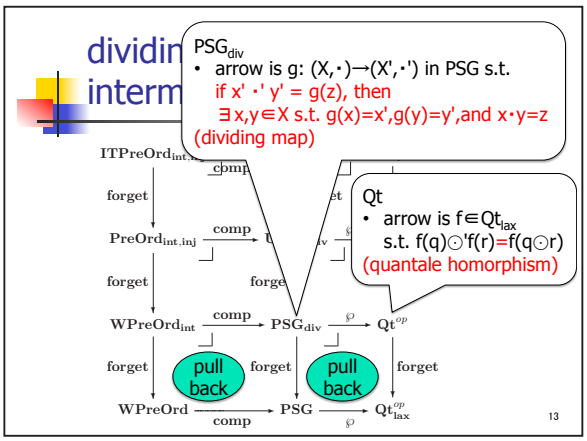
(dividing map)

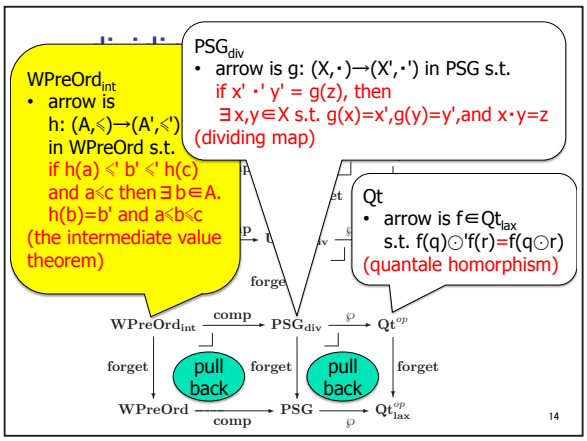


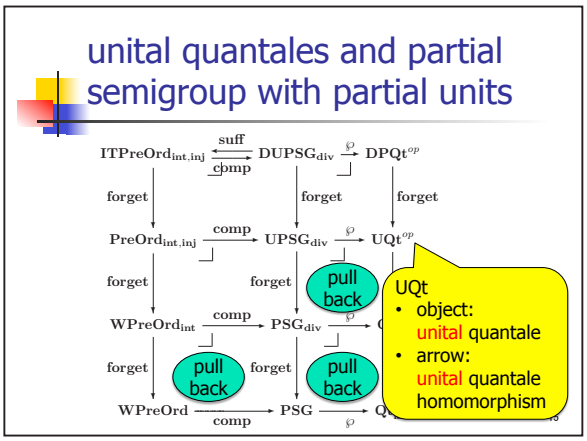

Qt
- arrow is f∈Qt_lax s.t. f(q)⊙'f(r)=f(q⊙r)

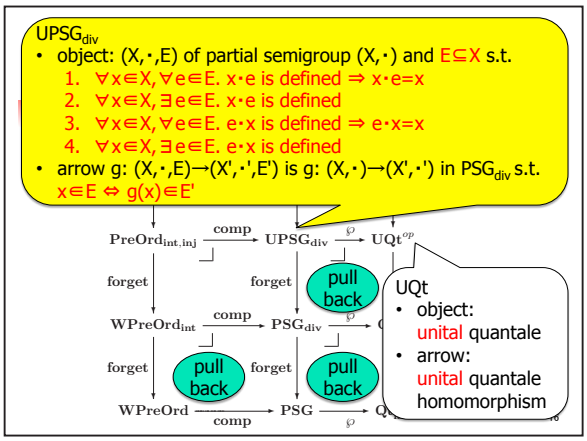
(quantale homorphism)

12

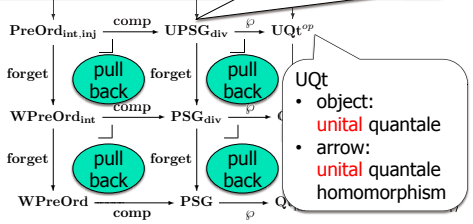

dividin
interm
PSG$_{div}$
• arrow is g: (X,·)→(X',·') in PSG s.t.
if x' ·' y' = g(z), then
∃x,y∈X s.t. g(x)=x',g(y)=y',and x·y=z
(dividing map)
Qt
• arrow is f∈Qt$_{lax}$
s.t. f(q)⊙'f(r)=f(q⊙r)
(quantale homorphism)
ITPreOrd$_{int,inj}$
PreOrd$_{int,inj}$
WPreOrd$_{int}$
PSG$_{div}$
Qt$^{op}$
WPreOrd
PSG
Qt$^{op}_{lax}$
comp
forget
pull back
13


WPreOrd$_{int}$
• arrow is
h: (A,⩽)→(A',⩽')
in WPreOrd s.t.
if h(a) ⩽' b' ⩽' h(c)
and a⩽c then ∃b∈A.
h(b)=b' and a⩽b⩽c
(the intermediate value theorem)
PSG$_{div}$
• arrow is g: (X,·)→(X',·') in PSG s.t.
if x' ·' y' = g(z), then
∃x,y∈X s.t. g(x)=x',g(y)=y',and x·y=z
(dividing map)
Qt
• arrow is f∈Qt$_{lax}$
s.t. f(q)⊙'f(r)=f(q⊙r)
(quantale homorphism)
WPreOrd$_{int}$
PSG$_{div}$
Qt$^{op}$
WPreOrd
PSG
Qt$^{op}_{lax}$
comp
forget
pull back
14


unital quantales and partial semigroup with partial units
suff
ITPreOrd$_{int,inj}$
DUPSG$_{div}$
DPQt$^{op}$
PreOrd$_{int,inj}$
UPSG$_{div}$
UQt$^{op}$
WPreOrd$_{int}$
PSG$_{div}$
WPreOrd
PSG
comp
forget
pull back
UQt
• object:
unital quantale
• arrow:
unital quantale
homomorphism

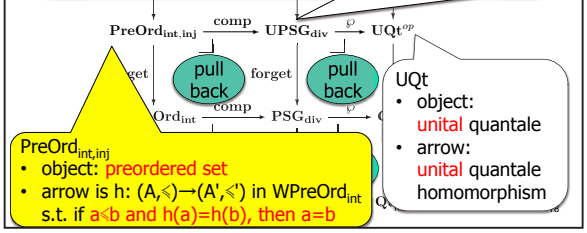

UPSG$_{div}$
• object: (X,·,E) of partial semigroup (X,·) and E⊆X s.t.
1. ∀x∈X,∀e∈E. x·e is defined ⇒ x·e=x
2. ∀x∈X,∃e∈E. x·e is defined
3. ∀x∈X,∀e∈E. e·x is defined ⇒ e·x=x
4. ∀x∈X,∃e∈E. e·x is defined
• arrow g: (X,·,E)→(X',·',E') is g: (X,·)→(X',·') in PSG$_{div}$ s.t.
x∈E ⇔ g(x)∈E'
PreOrd$_{int,inj}$
UPSG$_{div}$
UQt$^{op}$
WPreOrd$_{int}$
PSG$_{div}$
WPreOrd
PSG
comp
forget
pull back
UQt
• object:
unital quantale
• arrow:
unital quantale
homomorphism

$\mathrm{UPSG}_{\mathrm{div}}$
- object: $(X,\cdot,E)$ of partial semigroup $(X,\cdot)$ and $E\subseteq X$ s.t.
  1. $\forall x\in X,\forall e\in E.\ x\cdot e$ is defined $\Rightarrow x\cdot e=x$
  2. $\forall x\in X,\exists e\in E.\ x\cdot e$ is defined
  3. $\forall x\in X,\forall e\in E.\ e\cdot x$ is defined $\Rightarrow e\cdot x=x$
  4. $\forall x\in X,\exists e\in E.\ e\cdot x$ is defined
- arrow $g$: $(X,\cdot,E)\to(X',\cdot',E')$ is $g$: $(X,\cdot)\to(X',\cdot')$ in $\mathrm{PSG}_{\mathrm{div}}$ s.t. $x\in E \Leftrightarrow g(x)\in E'$




UQt
- object: unital quantale
- arrow: unital quantale homomorphism

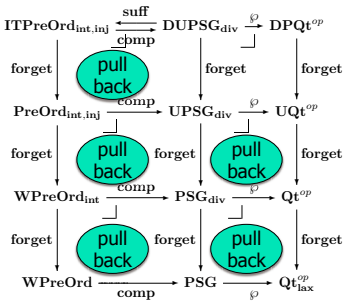
$\mathrm{UPSG}_{\mathrm{div}}$
- object: $(X,\cdot,E)$ of partial semigroup $(X,\cdot)$ and $E\subseteq X$ s.t.
  1. $\forall x\in X,\forall e\in E.\ x\cdot e$ is defined $\Rightarrow x\cdot e=x$
  2. $\forall x\in X,\exists e\in E.\ x\cdot e$ is defined
  3. $\forall x\in X,\forall e\in E.\ e\cdot x$ is defined $\Rightarrow e\cdot x=x$
  4. $\forall x\in X,\exists e\in E.\ e\cdot x$ is defined
- arrow $g$: $(X,\cdot,E)\to(X',\cdot',E')$ is $g$: $(X,\cdot)\to(X',\cdot')$ in $\mathrm{PSG}_{\mathrm{div}}$ s.t. $x\in E \Leftrightarrow g(x)\in E'$




$\mathrm{PreOrd}_{\mathrm{int,inj}}$
- object: preordered set
- arrow is h: $(A,\leqslant)\to(A',\leqslant')$ in $\mathrm{WPreOrd}_{\mathrm{int}}$ s.t. if $a\leqslant b$ and $h(a)=h(b)$, then $a=b$

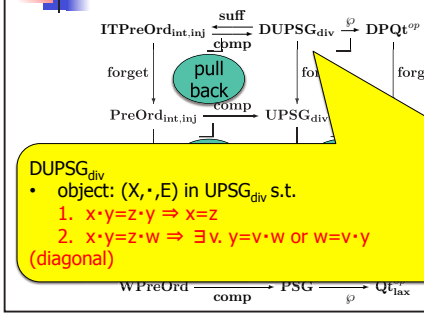
UQt
- object: unital quantale
- arrow: unital quantale homomorphism

# diagonal partial semigroups and interval-total preorders



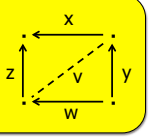

19

# diagonal partial semigroups and interval-total preorders

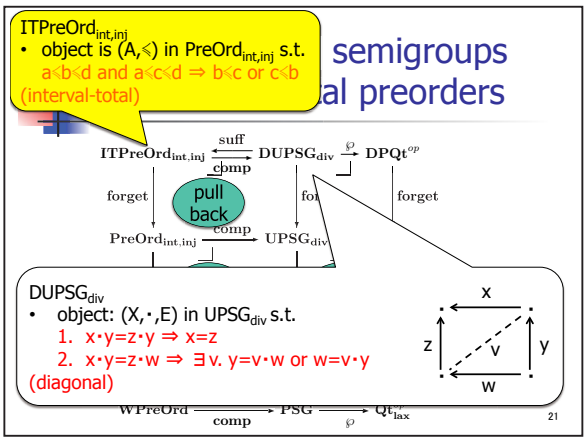


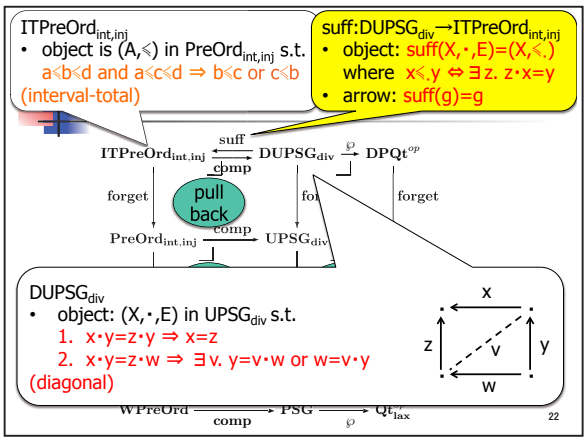

$\mathrm{DUPSG}_{\mathrm{div}}$
- object: $(X,\cdot,E)$ in $\mathrm{UPSG}_{\mathrm{div}}$ s.t.
  1. $x\cdot y=z\cdot y \Rightarrow x=z$
  2. $x\cdot y=z\cdot w \Rightarrow \exists v.\ y=v\cdot w$ or $w=v\cdot y$

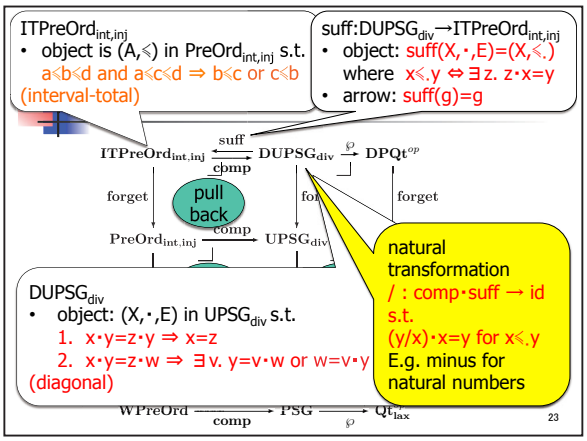
(diagonal)

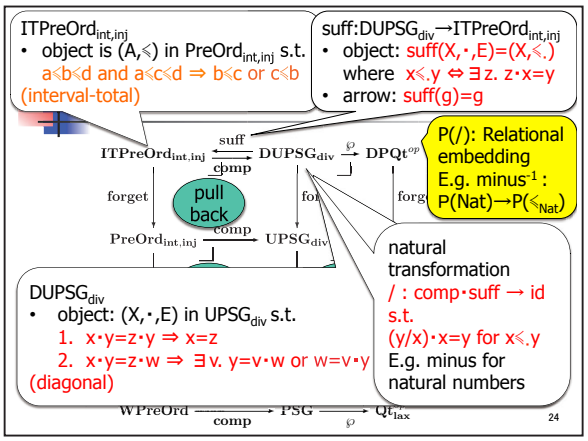



20

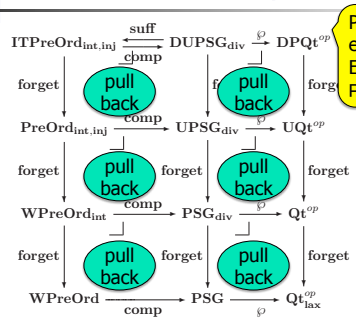

ITPreOrd$_{int,inj}$
• object is (A,⩽) in PreOrd$_{int,inj}$ s.t.
a⩽b⩽d and a⩽c⩽d ⇒ b⩽c or c⩽b
(interval-total)
semigroups
al preorders
suff
comp
ITPreOrd$_{int,inj}$
DUPSG$_{div}$
DPQt$^{op}$
forget
pull back
PreOrd$_{int,inj}$
UPSG$_{div}$
DUPSG$_{div}$
• object: (X,・,E) in UPSG$_{div}$ s.t.
1. x・y=z・y ⇒ x=z
2. x・y=z・w ⇒ ∃v. y=v・w or w=v・y
(diagonal)
x
y
z
v
w
WPreOrd
PSG
Qt$^{op}_{lax}$
21


ITPreOrd$_{int,inj}$
• object is (A,⩽) in PreOrd$_{int,inj}$ s.t.
a⩽b⩽d and a⩽c⩽d ⇒ b⩽c or c⩽b
(interval-total)
suff:DUPSG$_{div}$→ITPreOrd$_{int,inj}$
• object: suff(X,・,E)=(X,⩽.)
where x⩽.y ⇔ ∃z. z・x=y
• arrow: suff(g)=g
suff
comp
ITPreOrd$_{int,inj}$
DUPSG$_{div}$
DPQt$^{op}$
forget
pull back
PreOrd$_{int,inj}$
UPSG$_{div}$
DUPSG$_{div}$
• object: (X,・,E) in UPSG$_{div}$ s.t.
1. x・y=z・y ⇒ x=z
2. x・y=z・w ⇒ ∃v. y=v・w or w=v・y
(diagonal)
x
y
z
v
w
WPreOrd
PSG
Qt$^{op}_{lax}$
22


ITPreOrd$_{int,inj}$
• object is (A,⩽) in PreOrd$_{int,inj}$ s.t.
a⩽b⩽d and a⩽c⩽d ⇒ b⩽c or c⩽b
(interval-total)
suff:DUPSG$_{div}$→ITPreOrd$_{int,inj}$
• object: suff(X,・,E)=(X,⩽.)
where x⩽.y ⇔ ∃z. z・x=y
• arrow: suff(g)=g
suff
comp
ITPreOrd$_{int,inj}$
DUPSG$_{div}$
DPQt$^{op}$
forget
pull back
PreOrd$_{int,inj}$
UPSG$_{div}$
DUPSG$_{div}$
• object: (X,・,E) in UPSG$_{div}$ s.t.
1. x・y=z・y ⇒ x=z
2. x・y=z・w ⇒ ∃v. y=v・w or w=v・y
(diagonal)
natural transformation
/ : comp・suff → id
s.t.
(y/x)・x=y for x⩽.y
E.g. minus for natural numbers
WPreOrd
PSG
Qt$^{op}_{lax}$
23


ITPreOrd$_{int,inj}$
• object is (A,⩽) in PreOrd$_{int,inj}$ s.t.
a⩽b⩽d and a⩽c⩽d ⇒ b⩽c or c⩽b
(interval-total)
suff:DUPSG$_{div}$→ITPreOrd$_{int,inj}$
• object: suff(X,・,E)=(X,⩽.)
where x⩽.y ⇔ ∃z. z・x=y
• arrow: suff(g)=g
P(/): Relational embedding
E.g. minus$^{-1}$ : P(Nat)→P(⩽$_{Nat}$)
suff
comp
ITPreOrd$_{int,inj}$
DUPSG$_{div}$
DPQt$^{op}$
forget
pull back
PreOrd$_{int,inj}$
UPSG$_{div}$
DUPSG$_{div}$
• object: (X,・,E) in UPSG$_{div}$ s.t.
1. x・y=z・y ⇒ x=z
2. x・y=z・w ⇒ ∃v. y=v・w or w=v・y
(diagonal)
natural transformation
/ : comp・suff → id
s.t.
(y/x)・x=y for x⩽.y
E.g. minus for natural numbers
WPreOrd
PSG
Qt$^{op}_{lax}$
24

## diagonal partial semigroups and interval-total preorders




P(/): Relational embedding
E.g. minus$^{-1}$ :
P(Nat)→P($\leqslant_{Nat}$)



## Summary

- To entend the relational embedding of powerset quantales to a natural transformation
  - There are counter examples.
- To define categories for
  1. weak preorders
  2. partial semigroups
  3. quantales
- To show pullbacks including the correspondence among
  1. maps satisfying the intermediate value theorem
  2. dividing maps
  3. quantale homomorphisms

# Game development with ALGI

Isamu HASEGAWA (SQUARE ENIX)
Tomoyuki YOKOGAWA (Okayama Prefectural University)



## Agenda

- Introduction
  - Game Development and Visual Script
  - Model Checking
- Body
  - Motivating example
  - Approach
  - Method
  - Preliminary Evaluation
- Conclusion



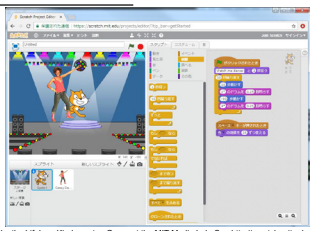
## What is “visual script”?

- Write logic by assembling visual elements.
  - scratch (MIT Media Lab.): assembling blocks
  - node graph based visual script: nodes and edges
- Advantages
  - Readable
  - Easy to develop
- Target / Purpose
  - For education
  - For non programmer



## What is “visual script”? (contd.)

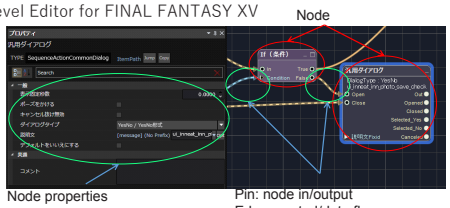
- Visual script for education
  - scratch
  - Blockly
  - Programin
    (プログラミン)



Scratch is developed by the Lifelong Kindergarten Group at the MIT Media Lab. See http://scratch.mit.edu

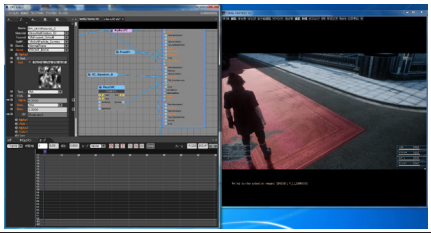
## Node graph based visual script

Level Editor for FINAL FANTASY XV



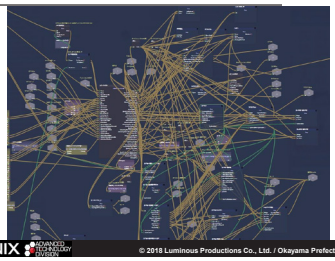



## Visual Script in VFX Development





## Typical Results





## Conventional Workflow

## What is Model Checking?

- Verifying correctness properties of a finite-state system
  - Automatic
  - Exhaustive
- Usually verify with the following steps:
  - Modeling: Describe the target system as a state transition graph.
  - Specification: Describe the properties that the system must satisfy as a temporal logic formula.
  - Verification: Verify whether the model satisfies the specification by graph search algorithm.



## What is Model Checking? (Contd.)

- Variety of implementations by prior researches.
  - SPIN：For verification of asynchronous distributed systems.
  - SMV：Efficient verification by symbolic model checking.
  - UPPAAL：For verification of real-time systems.
  - LTSA：Describe models as graphs.
- Pros: Automatic, Exhaustive, Counter Example
- Cons: State explosion.
  Not easy to learn writing models.



## NuSMV example

Model (sw.smv) :

```
MODULE main
VAR
  sw : {on, off};
ASSIGN
  init(sw) := {on, off};
  next(sw) := case
    sw = on : off;
    TRUE : sw;
  esac;
SPEC AG (AF sw = on)
```

Declare variable(s)

Define initial state and transition rule(s).

Execution result:

```
> NuSMV sw.smv
-- specification AG (AF sw = on)  is false
-- as demonstrated by the following execution sequence
Trace Description: CTL Counterexample
Trace Type: Counterexample
  -> State: 1.1 <-
    sw = on
  -- Loop starts here
  -> State: 1.2 <-
    sw = off
  -> State: 1.3 <-
```

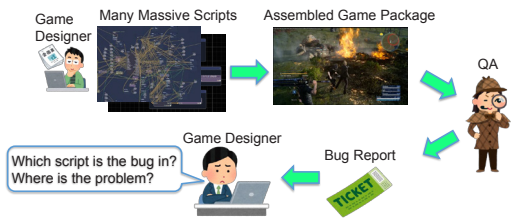
Verificatio result

Counter example:
sw = on,off,off,...

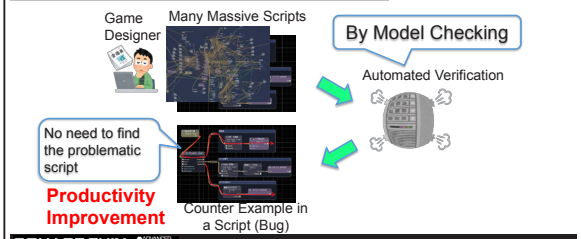
Describe specification
AG: for all series of transitions
AF: for the last state of all series of transitions



## Conventional Workflow

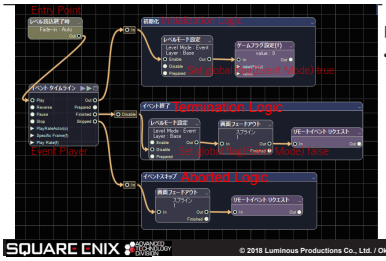


## Proposed workflow






## Approach – Example




Post Condition
- Global flag(Event Mode) must be false

Verification with Model Checking

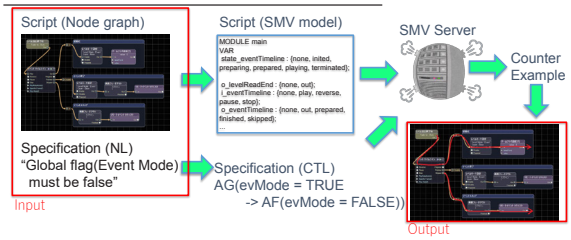


## Why Model Checking?

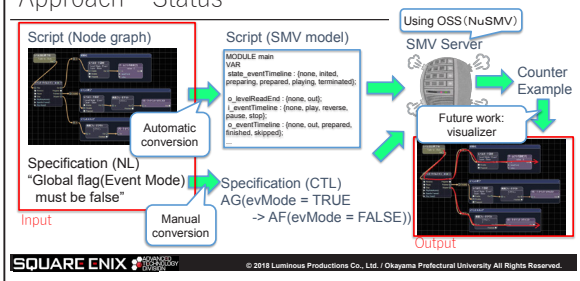
- Unit Test?
- Runtime Checking?
- AI / Machine Learning?



## Approach – Overview

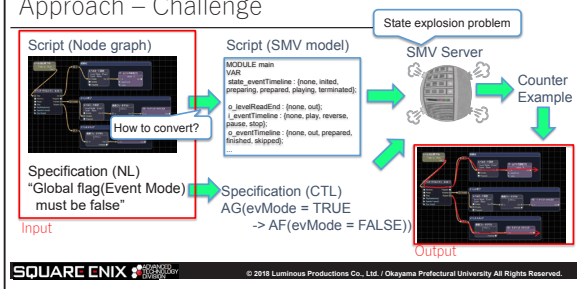


## Approach – Status






## Approach – Challenge



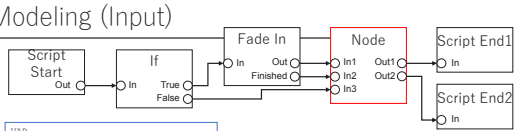



## Modeling Visual Scripts

- Properties that we want to verify:
  - There are no invalid control-flow that violates specification.
- Modeling approach
  - Describe each node model…
    - Define In/Out variables for input/output of the node respectively.
    - Define the Out variable transition according to the node semantics.
    - Abstract node semantics.
      E.g. "IF" node => non-deterministically transit to "TRUE" or "FALSE".
  - Then combine them.
    - Define the In variable transition according to the node connections.



## Modeling (Input)

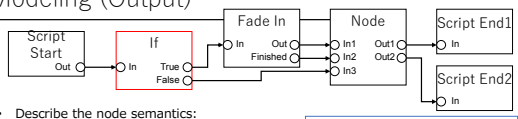



```
VAR
 Node_In  : {none, In1, In2, In3};
 Node_Out : {none, Out1, Out2};

ASSIGN
 init(Node_In) := none;
 next(Node_In) := case
  FadeIn_Out = Out       : In1;
  FadeIn_Out = Finished : In2;
  If_Out = False_        : In3;
  TRUE : none;
 esac;
```

- Define In/Out variable for node input/output.
- "none" means that no in(out) port is active.
- Describe the transition of In value according to the Out value that the In port connect to.

## Modeling (Output)

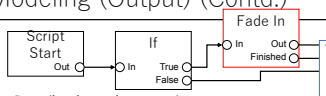



- Describe the node semantics:
  - If the node has no state transition
    ⇒ Assume that the output value is selected non-deterministically.

```
VAR
 If_In  : {none, In};
 If_Out : {none, True_, False_};
…
ASSIGN
 init(If_Out) := none;
 next(If_Out) := case
  If_in = In : {True_, False_};
  TRUE : none;
 esac;
```



## Modeling (Output) (Contd.)




- Describe the node semantics:
  - If the node has state transition
    ⇒ Add a state transition variable.

  e.g. Fade In node has the following states.
  - init: Initial state
  - out: Start processing state
  - wait: Waiting state
  - fin: Finished state

```
VAR
 FadeIn_In  : {none, In};
 FadeIn_Out : {none, Out, Finished};
 FadeIn_St  : {ini, out, wait, fin};

ASSIGN
 init(FadeIn_St) := ini;
 next(FadeIn_St) := case
  FadeIn_In = In  : out;
  FadeIn_St = out : wait;
  FadeIn_St = wait : {wait, fin};
  TRUE : ini;
 esac;
 init(FadeIn_Out) := none;
 next(FadeIn_Out) := case
  FadeIn_St = out : Out;
  FadeIn_St = fin : Finished;
  TRUE : none;
 esac;
```



## Modeling (Node semantics)

- E.g. EventMode node and EventMode flag.
  - Define variable for global flag.
  - Update the variable according to the input of EventMode node.
  - … so that we can use the global flag as the variable in specification.


```
VAR
 g_EventMode  : {enabled, disabled};

ASSIGN
 init(g_EventMode) := disabled;
 next(g_EventMode) := case
  EventMode_In = Enable  : enabled;
  EventMode_In = Disable : disabled;
  TRUE : g_EventMode;
 esac;

…

SPEC AG(g_EventMode = enabled
        -> AF(g_EventMode = disabled))
```



## Preliminary Evaluation

- Apply visual scripts used in FF15 (randomly selected).
- Verify that EventMode is correctly transited.
- If NuSMV outputs counter examples, confirm that it is a bug.

## Results (Preliminary Evaluation)

| No | # of Nodes | # of Vars | Conversion [s] | Verification [s] | Counter Example | Bug |
|---|---|---|---|---|---|---|
| 1 | 156 | 356 | 5.434 | 192.786 | | - |
| 2 | 94 | 214 | 3.878 | 3.330 | | - |
| 3 | 37 | 84 | 1.746 | 0.056 | | - |
| 4 | 49 | 119 | 2.301 | 0.111 | | - |
| 5 | 177 | 414 | 6.625 | 36.675 | Yes | Yes |
| 6 | 73 | 162 | 2.768 | 0.173 | | - |
| 7 | 162 | 408 | 9.187 | 98.102 | Yes | Yes |
| 8 | 430 | 980 | 13.286 | - | | - |



## Conclusion

- Our method detects actual bugs.
- The majority of the scripts can be verified within a few minutes.
- The verification of some huge scripts haven't finished within a few hours.

Future work:
- Optimization for smaller model.
- Compositional verification

# Fixed-point Arithmetic and Program Verification

T. Tsukada (U. Tokyo)

1st Sep 2019
ALGI, Kyushu University

## Program verification

A family of problems to decide, given

$P$ : a **program** $\phi$ : a **specification**,

whether $P$ **satisfies** $\phi$

**Example**

$P =$

```
int sum(n) {
  if(n == 0) { return 0 }
  else { return n + sum(n-1) } }

void main(n) {
  assert(n <= sum(n)) }
```

$\phi$ = "the assertion does not fail for all inputs"

## Logical approaches to verification

Verification Problem ⟶ Satisfiability, Validity
Model checking, ...

**Example**

**Safety** verification → Sat. of **constrained Horn clauses**

```
int sum(n) {
  if(n==0){ return 0 }
  else{ return n+sum(n-1) }
}

void main(n) {
  assert(n <= sum(n))
}
```

never fails

⟼

There exists a predicate $H$ on integers such that

$$\texttt{true} \Rightarrow H(0,0)$$
$$H(n-1,m) \Rightarrow H(n,m+n)$$
$$H(n,m) \Rightarrow n \leq m$$

## Classification of properties

Safety / liveness / $\omega$-regular

- Safety = **something bad** will never occur
- Liveness = **something good** will happen
- $\omega$-regular ≒ **parity condition**

$(n_i)_{i\in\omega} \in \{0,1,\dots,k\}^\omega$ satisfies the *parity condition* if and only if $\max\{\, p \mid \#\{\, i \mid n_i = p\} = \infty\}$ is even.

Linear-time / branching-time

- Linear-time ≒ **nondeterministic**
- Branching-time ≒ **alternating**

## This talk

On a logic suitable for verification of
**branching-time $\omega$-regular properties**

**Fact** The verification problem is in $\Delta^1_2$. It is $\Pi^1_1$-hard and $\Sigma^1_1$-hard.

**Thm** The verification problem is **polynomial-time many-one equivalent** to the validity problem of **fixed-point arithmetic**

**Remark** Existing logical approaches reduce the problem to strictly more difficult problems

## Outline

**Preliminaries**

- **Fixed-point Arithmetic**
- Verification problem

From program verification to fixed-point arithmetic

From fixed-point arithmetic to program verification

Analysis of existing logical methods

## Fixed-point arithmetic [Lubarsky 1993]

First-order arithmetic + (least/greatest) fixed-points

- Quantifiers over natural numbers are definable

$$
\begin{aligned}
s, t &::= x \mid \mathsf{Z} \mid \mathsf{S}\, t \\
\varphi, \psi &::= (s = t) \mid (s \neq t) \mid \varphi \wedge \psi \mid \varphi \vee \psi \\
&\quad \mid \Phi\, t_1, \ldots, t_k \\
\Phi, \Psi &::= X \mid \lambda x_1 \ldots x_k.\varphi \mid \mu X.\Phi \mid \nu X.\Phi
\end{aligned}
$$

## Semantics

The standard interpretation in the full model

- A valuation $\rho$ maps
  - a term variable to a natural number
  - a predicate variable of arity $k$ to a subset of $\mathbb{N}^k$
- The interpretation of formulas is straightforward:

$$
\begin{aligned}
[\![x]\!]_\rho &:= \rho(x) \\
[\![\lambda x_1 \ldots x_k.\varphi]\!]_\rho &:= \{(n_1, \ldots, n_k) \mid [\![\varphi]\!]_{\rho[\vec{x} \mapsto \vec{n}]} = \top\} \\
[\![\nu X.\Phi]\!]_\rho &:= \bigcup \{A \subseteq \mathbb{N}^k \mid A = [\![\Phi]\!]_{\rho[X \mapsto A]}\} \\
&\vdots
\end{aligned}
$$

## Definability of negation

Negation on **1st-order predicates** is definable [Loze, 15]

- Let $\neg\varphi$ be the formula obtained from $\varphi$ by replacing

$$(s = t) \leftrightarrow (s \neq t)$$
$$(\varphi \wedge \psi) \leftrightarrow (\varphi \vee \psi)$$
$$(\mu X.\Phi) \leftrightarrow (\nu X.\Phi)$$

For formulas **with free set variables**,
negation is not expressive

## Example 0: Truth values

$$[\![\mu X.X]\!] = \bot$$
$$[\![\nu X.X]\!] = \top$$

## Example 1: Quantifiers over numbers

**Existential**

Let $E_x(\varphi) := \mu X.\lambda x.(\varphi \vee X\,(\mathtt{S}\,x))$

Then $E_x(\varphi)\,n$ holds if and only if $\exists x \geq n.\varphi$

**Universal**

Let $A_x(\varphi) := \nu X.\lambda x.(\varphi \wedge X\,(\mathtt{S}\,x))$

Then $A_x(\varphi)\,n$ holds if and only if $\forall x \geq n.\varphi$

**Notation**

$$\exists x.\varphi \;:=\; E_x(\varphi)\,\mathtt{Z} \qquad \forall x.\varphi \;:=\; A_x(\varphi)\,\mathtt{Z}$$

## Example 2: Addition/multiplication

**Addition**

Let $\mathit{plus}$ be the formula defined by

$$\mu X.\lambda x y z.\left(\begin{array}{l}((y = \mathtt{Z}) \wedge (x = z)) \\ \quad \vee \exists y'.\exists z'.((y = \mathtt{S}\,y') \wedge (z = \mathtt{S}\,z') \wedge (X\,x\,y'\,z'))\end{array}\right)$$

Then $(n, m, \ell) \in [\![\mathit{plus}]\!]$ iff $n + m = \ell$

Cf. (A prolog program for addition)

$$\mathit{plus}(X, Y, Z) \leftarrow Y = 0 \wedge X = Z$$
$$\mathit{plus}(X, \mathtt{S}(Y), \mathtt{S}(Z)) \leftarrow \mathit{plus}(X, Y, Z)$$

## Example 3: Well-foundedness

Let $nwf(\Phi)$ be the formula defined by

$$\nu X.\lambda x.\exists y.(\Phi\, x\, y) \wedge (X\, y)$$

Then

$n \in [\![nwf(\Phi)]\!]_\rho$ iff $[\![\Phi]\!]_\rho$ has an infinite path from $n$

Given a closed binary predicate $\Phi$, let

$$wf(\Phi) \;:=\; \mu X.\lambda x.\forall y.((\neg\Phi)\, x\, y) \vee (X\, y)$$

Then $[\![\forall x.wf(\Phi)\, x]\!] = \top$ iff $[\![\Phi]\!]$ is well-founded

Remark

## Constrained Horn clauses = $\nu$-arithmetic

$$\begin{array}{r}
\texttt{true} \Rightarrow H(0,0) \\
H(n,m) \Rightarrow H(n+1, m+n+1) \\
H(n,m) \Rightarrow n \leq m
\end{array}$$

$$\downarrow$$

$$\begin{array}{r}
x = 0 \wedge y = 0 \Rightarrow H(x,y) \\
x = \mathtt{S}(n) \wedge y = x + m \wedge H(n,m) \Rightarrow H(x,y) \\
H(n,m) \Rightarrow n \leq m
\end{array}$$

$$\downarrow$$

$$\begin{array}{r}
x = 0 \wedge y = 0 \Rightarrow H(x,y) \\
(\exists n.\exists m.x = \mathtt{S}(n) \wedge y = x + m \wedge H(n,m)) \Rightarrow H(x,y) \\
H(n,m) \Rightarrow n \leq m
\end{array}$$

Remark

## Constrained Horn clauses = $\nu$-arithmetic

$$\begin{array}{r}
x = 0 \wedge y = 0 \Rightarrow H(x,y) \\
(\exists n.\exists m.x = \mathtt{S}(n) \wedge y = x + m \wedge H(n,m)) \Rightarrow H(x,y) \\
H(n,m) \Rightarrow n \leq m
\end{array}$$

$$\downarrow$$

$$\begin{array}{r}
(x = 0 \wedge y = 0) \vee \\
(\exists n.\exists m.x = \mathtt{S}(n) \wedge y = x + m \wedge H(n,m)) \Rightarrow H(x,y) \\
H(n,m) \Rightarrow n \leq m
\end{array}$$

$$\downarrow$$

$$\mu H.\left(\begin{array}{r}
\mu H.(x = 0 \wedge y = 0) \vee \\
(\exists n.\exists m.x = \mathtt{S}(n) \wedge y = x + m \wedge H(n,m))
\end{array}\right) \Rightarrow n \leq m$$

## Outline

**Preliminaries**

- Fixed-point Arithmetic
- **Verification problem**

From program verification to fixed-point arithmetic

From fixed-point arithmetic to program verification

Analysis of existing logical methods

## Verification problem

A subclass of the problem in Kobayashi *et al.* [ESOP 18]

**Input**

- A CbN functional program of atomic type $\mathtt{o}$
  - Primitives include $\wedge, \vee : \mathtt{o} \to \mathtt{o} \to \mathtt{o}$ and **events** $\mathtt{e}_\ell : \mathtt{o} \to \mathtt{o},\ \ell \in \{0, 1, \ldots, L\}$

**Output**

- Whether Proponent wins the **reduction game**

A general verification problem can be reduced to this

## Target language

A CbN programming language with natural numbers

$$\begin{aligned} M, N, L := {} & x \mid \lambda x^A.M \mid M\ N \mid \mathbf{Y}M \\ & \mid \mathtt{Z} \mid \mathtt{S}\ M \mid \mathtt{ifeq}\ M\ N\ L_1\ L_2 \\ & \mid M \wedge N \mid M \vee N \\ & \mid \mathtt{e}_\ell(M) \end{aligned}$$

$$\begin{pmatrix} \mathtt{Z} : \mathtt{N} \\ \mathtt{S} : \mathtt{N} \to \mathtt{N} \\ \mathtt{ifeq} : \mathtt{N} \to \mathtt{N} \to \mathtt{o} \to \mathtt{o} \to \mathtt{o} \\ \wedge, \vee : \mathtt{o} \to \mathtt{o} \to \mathtt{o} \\ \mathtt{e}_\ell : \mathtt{o} \to \mathtt{o} \end{pmatrix}$$

## Verification problem

A subclass of the problem in Kobayashi *et al.* [ESOP 18]

**Input**

- A CbN functional program of atomic type $\mathtt{o}$
  - Primitives include $\wedge, \vee : \mathtt{o} \to \mathtt{o} \to \mathtt{o}$ and **events** $\mathtt{e}_\ell : \mathtt{o} \to \mathtt{o},\ \ell \in \mathbb{N}$

**Output**

- Whether Proponent wins the **reduction game**

A general verification problem can be reduced to this

## Game

**Def** A **game graph** is a tuple $G = (V_\oplus, V_\ominus, E)$

- $V_\oplus$ and $V_\ominus$ are disjoint sets of P- and O-nodes
  - $V := V_\oplus \cup V_\ominus$
- $E \subseteq V \times V$ is the edge relation
  - We assume $\forall x \exists y\ E(x, y)$

**Def** A **game** is a tuple $(G, v_0, W)$ of

- a game graph $G$,
- an initial node $v_0 \in V$, and
- a **winning criterion** $W \subseteq V^\omega$

## Induced game

**Def** The **reduction game graph** $\mathcal{R}$ is given by

$$V_{\oplus} := \{M \vee N \mid \vdash M : \mathsf{o},\ \vdash N : \mathsf{o}\}$$
$$V_{\ominus} := \{M \mid \vdash M : \mathsf{o},\ M \neq N \vee L\}$$
$$V := \{M \mid \vdash M : \mathsf{o}\}$$
$$E := \{(M, N) \mid M \longrightarrow N\}$$
$$\cup \{(M \square N, M), (M \square N, N) \mid \square \in \{\wedge, \vee\}\}$$
$$\cup \{(\mathsf{e}_\ell(M), M) \mid \vdash M : \mathsf{o}\}$$

Let $W$ be the parity condition, where the priority is

$$priority(M) = \begin{cases} \ell & (\text{if } M = \mathsf{e}_\ell(M)) \\ 0 & (\text{otherwise}) \end{cases}$$

## Verification problem

A subclass of the problem in Kobayashi *et al.* [ESOP 18]

**Input**

- A CbN functional program $\vdash M : o$
  - Primitives include $\wedge, \vee : \mathsf{o} \to \mathsf{o} \to \mathsf{o}$ and **events** $\mathsf{e}_\ell : \mathsf{o} \to \mathsf{o},\ \ell \in \mathbb{N}$

**Output**

- Whether P wins the game $(\mathcal{R}, M, W)$

A general verification problem can be reduced to this

## Verification problem is in $\Delta^1_2$

The reduction game graph is computable

P wins the reduction game just if:

Resulting play

$\exists^1 f\ \forall^1 g\ \forall^1 h$. "$h$ is the play that conforms with $f, g$"

O-strategy

⇒ "$h$ satisfies P.C."

P-strategy

$\forall^1 g\ \exists^1 f\ \exists^1 h$. "$h$ is the play that conforms with $f, g$"

∧ "$h$ satisfies P.C."

## Outline

Preliminaries

- Fixed-point Arithmetic
- Verification problem

**From program verification to fixed-point arithmetic**

From fixed-point arithmetic to program verification

Analysis of existing logical methods

## From verification to fix. arith.

1. By coding, every program $P$ can be efficiently translated to a first-order program. For simplicity, assume that the first-order program is of the form:
$$(\mathbf{Y}\,(\lambda X^{\mathtt{N}\to\cdots\to\mathtt{N}\to\mathtt{o}}.M))\,\lfloor P\rfloor\,\mathtt{Z}\,\ldots\,\mathtt{Z}$$
where $M$ is recursion-free and normal.
2. Apply the rewriting rules such as
$$\mathtt{e}_\ell(M\wedge N)\mapsto \mathtt{e}_\ell(M)\wedge \mathtt{e}_\ell(N)$$
$$\mathtt{e}_\ell(\mathtt{ifeq}\,M\,N\,L\,L')\mapsto \mathtt{ifeq}\,M\,N\,(\mathtt{e}_\ell(L))\,(\mathtt{e}_\ell(L'))$$
which do not change the winner.

## From verification to fix. arith.

3. Then occurrences of events is restricted to the form
$$\mathtt{e}_{\ell_1}(\ldots\mathtt{e}_{\ell_n}(X\,t_1\,\ldots\,t_m)\ldots)$$
which we replace with
$$X_{\max\{\ell_1,\ldots,\ell_n\}}\,t_1\,\ldots\,t_m$$
4. For a sufficiently large even number $k$,
$$(\nu X_k.\mu X_{k-1}.\ldots.\mu X_1.\nu X_0.M'')\,\lfloor P\rfloor\,\mathtt{Z}\,\ldots\,\mathtt{Z}$$
is valid iff P wins the reduction game.

## Outline

Preliminaries

- Fixed-point Arithmetic
- Verification problem

From program verification to fixed-point arithmetic

**From fixed-point arithmetic to program verification**

Analysis of existing logical methods

## Reduction graph of formulas

**Reduction of formulas** can be naturally defined

$$(\lambda\bar{x}.\varphi)\,\bar{t}\longrightarrow\varphi[\bar{t}/\bar{x}]$$
$$\varphi_1\Box\varphi_2\longrightarrow\varphi_i \qquad \Box\in\{\wedge,\vee\},i\in\{1,2\}$$
$$\alpha X.\varphi\longrightarrow\varphi[(\alpha X.\varphi)/X] \qquad \alpha\in\{\mu,\nu\}$$
$$t=t\longrightarrow\nu X.X$$
$$t=s\longrightarrow\mu X.X \qquad \text{if } t\neq s$$

So **a formula can be seen as a program**

- What is left is to insert appropriate events

## From fix. arith. to verification

$$(t = u)^\dagger := \mathtt{ifeq}\, t\, u\, (\mathbf{Y}\, (\lambda f.\mathtt{e}_0(f)))\, (\mathbf{Y}\, (\lambda y.\mathtt{e}_1(f)))$$
$$(\varphi \wedge \psi)^\dagger := \varphi^\dagger \wedge \psi^\dagger$$
$$(\mu X.\Phi)^\dagger := \mathbf{Y}\, (\lambda X.\lambda \bar{x}.(\mathtt{e}_\ell(\Phi^\dagger\, \bar{x})))$$
$\ell$ is an odd number that is greather than all numbers in $\Phi^\dagger$
$$(\nu X.\Phi)^\dagger := \mathbf{Y}\, (\lambda X.\lambda \bar{x}.(\mathtt{e}_\ell(\Phi^\dagger\, \bar{x})))$$
$\ell$ is an even number that is greather than all numbers in $\Phi^\dagger$

## Outline

Preliminaries

- Fixed-point Arithmetic
- Verification problem

From program verification to fixed-point arithmetic

From fixed-point arithmetic to program verification

**Analysis of existing logical methods**

## Logics for verifying higher-order programs

- Verification by 2nd-order arithmetic [Unno *et al.* POPL18]
  - Essentially relies on coding of programs
  - 2nd-order arith. is more difficult than fix. arith.
- $\mu$-calculus based approach [Nanjo *et al.* LICS18]
  - More difficult than 2nd-order arithmetic because of quantifiers over infinite strings

## Higher-order fixed-point arithmetic

Kobayashi *et al.* [ESOP 2018] gave a reduction from the verification problem to **higher-order** fixed-point arith.

**Thm** The sets definable by higher-order fixed-point arithmetic is in $\Delta^1_2$

Proof) Similar to the first-order case: The validity of a given formula can be characterized by a gave over the reduction graph (with a more complicated winning criterion).

## Higher-order fixed-point arithmetic

Kobayashi *et al.* [ESOP 2018] gave a reduction from the verification problem to **higher-order** fixed-point arith.

**Thm** The sets definable by higher-order fixed-point arithmetic is in $\Delta_2^1$

**Thm** The set of true order-k fix. arith. sentences definable by an order-(k+1) formula

**Cor** For $k \geq 2$, the validity problem of **order-k fix. arith. is strictly more difficult** than verification

## Horn clauses for temporal verification

[Beyene *et al.* POPL14]

Reduces a game solving to satisfaction problem for collections of constraints of the form

$$\forall \vec{x}.\left(P_1(\vec{v}_1) \wedge \cdots \wedge P_n(\vec{v}_n) \wedge \varphi \rightarrow (\exists \vec{y}.Q_1(\vec{w}_1) \vee \cdots \vee Q_m(\vec{w}_m))\right)$$
$$\text{well-founded}(P)$$

where $P_i$ and $Q_i$ are predicate variables.

- It is more difficult than validating $\Sigma_2^1$ formulas
- Subproblems w/o ∃,∨ and w/o wf are strictly easier than the verification problem

## Conclusion and future work

**Fixed-point arithmetic** is a nice logic for verification

- It is **as difficult as the verification problem**

**Other logical approaches** reduce the verification problem to **strictly more difficult** problems

**Future work**

Find a nice proof system

- higher-order logic / cyclic proof system

Fix. arith. has a tight connection to **game quantifier** [Bradfield 1999]. How about the higher-order fix. arith.?

# 非構成的対象を扱った証明からのプログラム抽出

立木 秀樹 (京都大学 人間・環境学研究科)

IFP (Intuitionistic Fixed Point Logic) は，Swansea 大学の Ulrich Berger が中心となって，協力者らとともに作ってきたプログラム抽出のための論理システムである [1-5] 。発表者も近年そのプロジェクトに参加している [6]。

IFP の特徴は、実数などの公理的に定義された数学的対象に関する抽象的な証明からプログラムを抽出できることである。IFP の論理は，多ソート直観主義一階述語論理を，帰納法と余帰納法が使えるように拡張したものである。実数などの証明で扱う対象はソートとして与えられ，その性質は公理を通じて定義され，通常の数学と同様の形で IFP の中で証明があたえられる。実数などに関しては，計算を行うための表現方法は定めずに，あくまでも抽象的な数学的対象として考えていることに注意して欲しい。

それに対し，プログラミング言語は，直積 (Pair)，直和 (Left, Right)，自明なリアライザ (Nil) のそれぞれを示す構成子と，再帰 (rec) の演算子を含んだ型なし $\lambda$ 計算として与えられており，データやプログラムはこれらの演算子を用いて表現される。IFP のプログラムは Haskell のプログラムに直接的に変換することができ，Haskell システムの中で実行することができる。

通常，証明抽出システムでは，実現可能性解釈 (BHK 解釈) という直観主義論理の計算的な意味づけを通じてプログラムの抽出を行う。IFP もそれに従っているが，多少，通常の BHK 解釈とは異なっている。プログラム $c$ が論理式 $A$ を実現する という関係 $c\,\mathbf{r}\,A$ を，述語定数 $P$ に対しては

- $c\,\mathbf{r}\,P(\vec{t}) = (c = \mathbf{Nil} \wedge P(\vec{t}))$

と定義し，$A \wedge B$, $A \vee B$, $A \to B$ に対しては，

- $c\,\mathbf{r}\,(A \wedge B) = \exists a, b\,(c = \mathbf{Pair}(a, b) \wedge a\,\mathbf{r}\,A \wedge b\,\mathbf{r}\,B)$
- $c\,\mathbf{r}\,(A \vee B) = \exists a\,(c = \mathbf{Left}(a) \wedge a\,\mathbf{r}\,A) \vee \exists b\,(c = \mathbf{Right}(b) \wedge b\,\mathbf{r}\,B)$
- $c\,\mathbf{r}\,(A \to B) = \forall a\,(a\,\mathbf{r}\,A \to (c\,a)\,\mathbf{r}\,B)$

と定義する。すなわち，$A \vee B$ の証明があれば，そこから，$A$ が成り立つか $B$ が成り立つかという情報が得られ，それを **Left**(), **Right**() という構成子でもってコード化することによりプログラムが作られるという仕組みである。ここまでは，通常の BHK 解釈によるものと同じである。それに対し，$\forall$ と $\exists$ に対するものは，通常と異なった次のような解釈を考える。

- $c\,\mathbf{r}\,\forall x\,A(x) = \forall x\,(c\,\mathbf{r}\,A(x))$

- $c\,\mathbf{r}\,\exists x\,A(x) = \exists x\,(c\,\mathbf{r}\,A(x))$

これらは，通常の BHK 解釈 $c$ **pf** $A$ においては，次のように定義されるものである。

- $c$ **pf** $\forall x\,A(x) \quad \cdots \quad \forall x \in D \forall a(a$ **pf** $x \to c(a)$ **pf** $A(x))$ .
- $c$ **pf** $\exists x\,A(x) \quad \cdots \quad \exists x \in D \exists a, b(a$ **pf** $x \wedge c = \mathbf{Pair}(a, b) \wedge b$ **pf** $A(x))$.

ここで，$D$ と書いたのは，Basic Domain と言われる，変数の動く領域である。$a$ **pf** $x$ という記述が示すように，Basic Domain の要素 $x$ は，その解釈として，プログラムとしての表現が与えられている必要がある。言い換えると，論理の扱う対象は，最初からプログラムとしての表現が与えられたものでないといけない。そのために，多くの研究では，実数などを扱う時には，Cauchy 列表現などの表現を一つ固定して，その表現を通じて実数を考えるといったことが行われている。

しかし，数学でいう実数は表現とは独立に抽象的に考えられる対象であり，数学的な証明は，表現は考えずに公理系から推論規則を用いて行われるものである。よって，我々は，$\forall$ と $\exists$ に対する解釈として，上に述べたような一様な解釈を考えることにする。通常の BHK 解釈は，対応する表現を持っているという述語 $D(x)$ を考えて，変数を $D(x)$ に相対化することによって得られる。

- $\forall x\,A(x) \quad \cdots \quad \forall x \in D\,A(x) = \forall x\,(D(x) \to A(x))$.
- $\exists x\,A(x) \quad \cdots \quad \exists x \in D\,A(x) = \exists x\,(D(x) \wedge A(x))$.

と考えると

- $$\begin{aligned} c\,\mathbf{r}\,\forall x(D(x) \to A(x)) &= \forall x(c\,\mathbf{r}\,D(x) \to A(x)) \\ &= \forall x \forall a(a\,\mathbf{r}\,D(x) \to c(a)\,\mathbf{r}\,A(x)) \end{aligned}$$
- $$\begin{aligned} c\,\mathbf{r}\,(\exists x\ D(x) \wedge A(x)) &= \exists x\,(c\,\mathbf{r}\,D(x) \wedge A(x)) \\ &= \exists x \exists a, b(c = \mathbf{Pair}(a, b) \wedge a\,\mathbf{r}\,D(x) \wedge b\,\mathbf{r}\,A(x)) \end{aligned}$$

という具合に，通常の BHK 解釈に相当するものが得られる。

$D(x)$ の述語は，帰納法や余帰納法を用いて定義される。自然数でも実数でも，その上で計算を行うための帰納的な構造は 1 種類ではなく，その場その場で変わってくるのが普通である。特に，実数の時には，例えば $x \neq 0$ の時のみ考えるなど，定義域が制限されることも多く，その制限された定義域に対する（余）帰納的な構造を考えることになる。よって，一つの表現だけを考えてそのもとで計算を考えるのよりも，公理としては数学的な公理だけを用いて，計算のために使える（余）帰納的な構造は，その都度与えるようにしている。

通常の BHK 解釈では，$A \vee B$ の証明だけではなく，$\exists x\,A(x)$ の証明も，$A(x)$ を満たす $x$ は何かを計算するプログラム (上記の $a$) を与えてくれていた。それに対し，我々の解釈では，$A \vee B$ だけが計算のための情報のもととなっていることに注意して欲しい。

実際には，上記の $c\,\mathbf{r}\,A$ の定義は単純化されたものであり，自明なプログラムしか実現可能なものとして持たない論理式を意味する Harrop formula という概念を導入し，それ

を用いて，本当に必要な計算だけを行うプログラムが抽出されるようなシステムとなっている。また，IFP には型の概念があり，論理式から導かれる型に型付けされるプログラムのみが抽出される。論理式 $A$ の証明が与えられると，$c\,\mathbf{r}\,A$ を満たす $c$ が，その証明から抽出される。

IFP の使用例として，実数の公理を一つ与えて，そのもとで，

$$\mathbf{N}(x) \stackrel{\mu}{=} x = 0 \vee \mathbf{N}(x-1)$$

という形で，$x$ が自然数であるということを表現する述語を最小不動点として表現し，

$$\mathbf{N}(x) \wedge \mathbf{N}(y) \to \mathbf{N}(x+y)$$

の証明を与え，その証明から足し算のプログラムを抽出できることを示される。

また，実数に対して，符号付き 2 進表現をもっていることを示す述語 $\mathbf{S}(x)$，グレイコード表現を持っていることを示す述語 $\mathbf{G}(x)$ を余帰納法を用いて定義し，$\forall x(\mathbf{S}(x) \to \mathbf{G}(x))$ の IFP における証明から，符号付き 2 進表現からグレイコード表現への変換プログラムを抽出することができる。

## 参考文献


[1] U. Berger. From coinductive proofs to exact real arithmetic. In E. Grädel and R. Kahle, editors, *Computer Science Logic*, volume 5771 of *Lecture Notes in Comput. Sci.*, pages 132–146. Springer Verlag, Berlin, Heidelberg, New York, 2009.

[2] U. Berger. Realisability for induction and coinduction with applications to constructive analysis. *Jour. Universal Comput. Sci.*, 16(18):2535–2555, 2010.

[3] U. Berger. From coinductive proofs to exact real arithmetic: theory and applications. *Logical Methods in Comput. Sci.*, 7(1):1–24, ?2011.

[4] U. Berger and O. Petrovska. Optimized program extraction for induction and coinduction. In *CiE 2018*, volume 10936 of *LNCS*, pages 70–80. Springer Verlag, Berlin, Heidelberg, New York, 2018.

[5] U. Berger and M. Seisenberger. Proofs, programs, processes. *Theory of Computing Systems*, 51(3):213–329, 2012.

[6] U. Berger and H. Tsuiki. Intuitionistic Fixed Point Logic. *In preparation.*

# 代数的・幾何的・論理的制約下の劣モジュラ関数最大化

前原貴憲 (RIKEN AIP)

2019 年 9 月 1 日

## 自己紹介

前原貴憲 (Takanori MAEHARA)
http://www.prefield.com

ユニットリーダー, 離散最適化ユニット, 理研 AIP

| | |
|---|---|
| 2012 | 博士 (情報理工学), 東京大学 |
| 2012–2015 | 特任研究員, 国立情報学研究所 |
| 2015–2017 | 助教, 静岡大学 |
| 2017–現在 | 現職 |

研究興味
- 離散構造論 (離散最適化・組合せ論・ etc)
- 数値解析 (行列・テンソル分解)
- 機械学習への応用



## 理研 AIP

理研：1917 年設立の自然科学 研究機関

理研 AIP：人工知能技術に関する研究センター

2016 年内閣府「人工知能技術戦略」を受けて設立
⇒ 予算が政府から直接出ている

2025 年までの時限付きプロジェクト

- 3 つの研究グループ（理論・応用・社会）
- 50 以上の研究チーム（深層学習・テンソル・ベイズ・ etc）
- 700 人以上の研究員 (フルタイムは 200, 1/3 は非日本人)



## 理研 AIP

日本橋駅至近 (東京駅から歩いて 10 分)
羽田空港・成田空港どちらからも直結

## 離散最適化ユニット

離散最適化：有限候補の組合せの中で最良のものを選ぶ問題
(代表的な問題：経路案内・配送計画・配線設計・ etc)

特に AI 関連分野では不確実性のある最適化問題 が重要

- AI 応用に動機をもつ離散最適化
- 離散最適化を AI に利用

数学的背景：計算量，凸解析，多面体，グラフ，群論，...





## Introduction

## Standard Discrete Optimization Problem

$E$: finite set; $f$: $2^E \to \mathbb{R}$: objective ft; $\mathcal{D} \subseteq 2^E$: feasible domain

$$\begin{array}{ll} \text{maximize} & f(X) \\ \text{subject to} & X \in \mathcal{D} \end{array}$$

- Solvable in a finite time because $2^E$ is finite; however, it is impractical ($2^{70}$ operations in 1GBHz PC takes the age of the universe)
- In general, $2^E$ time is necessary

**Research Purpose of Discrete Optimization Theory**
What $f$ and $\mathcal{D}$ can be optimized?



## cf: Continuous Optimization Problem

$F$: $\mathbb{R}^E \to \mathbb{R}$: objective ft; $\Omega \subseteq \mathbb{R}^E$: feasible domain

$$\begin{array}{ll} \text{maximize} & F(x) \\ \text{subject to} & x \in \Omega \end{array}$$

Solvable class: convex optimization problem

- $F(x)$ is a concave function;
  $F((1-t)x + ty)) \geq (1-t)F(x) + tF(y)$
- $\Omega$ is a convex set; $(1-t)x + ty \in \Omega$

Under regularity assumptions the problem is efficiently solvable (by a gradient descent method)

**Discrete Convex Analysis**
Discrete version of convex analysis
e.g., concave function $\Rightarrow$ submodular function

## Submodular Functions

$f\colon 2^E \to \mathbb{R}$ is submodular if

**Submodular Inequality (mid-point convexity)**

$$f(X) + f(Y) \geq f(X \cup Y) + f(X \cap Y)$$

or equivalently,

**Diminishing Return Property (concavity along positive directions)**

$$f(X \cup \{e\}) - f(X) \geq f(Y \cup \{e\}) - f(Y), \qquad X \subseteq Y$$

($\Downarrow$ easy; $\Uparrow$ induction)



## Example of Submodular Functions

- Coverage: $E$ set of shapes, $f(X)$: $X$ area covered by $X$
- Information: $E$ set of info. sources, $f(X)$: information of $X$
- Utility: $E$ set of items, $f(X)$: value of $X$
- Cut: $E$ set of edges of a graph, $f(X)$: number of edges from $X$
- ...

Submodular functions appear everywhere!



## Submodular Maximization Problem

$f\colon 2^E \to \mathbb{R}$ submodular

$$\begin{array}{ll} \text{maximize} & f(X) \\ \text{subject to} & X \in \mathcal{D} \end{array}$$

- Fundamental discrete optimization problem
- Recently studied in AI/DM domains
  - sensor placement (coverage ft)
  - document summarization (information)
  - budget allocation (utility ft)
  - maximum cut problem (cut ft)
  - ...



## Hardness / Approximability

Hardness

- It requires exponentially many oracle calls even if $f$ is monotone and $\mathcal{D}$ is a cardinality constraint
- The sensor placement problem is NP-hard

Approximability

- Constant factor approximations for several constraints
  - cardinality constraint
  - matroid constraint
  - knapsack constraint
  - ...

## TM's Research Interest

**Research Question**
What $\mathcal{D}$ allows a (non-trivial) approximation algorithm?

"Basic" constraints are already analyzed (later)
$\Rightarrow$ I want to consider more complicated constraints

Today's topic

- Subgraph of a graph specified by a logic
  (finite model theory, algorithmic metatheorem)
- Constraint on lattice
  (lattice theory / metric modular)



## Today's topic

- "standard" submodular maximization
- submodular maximization on logic
- submodular maximization on lattice

In the following, we assume that $f$ is normalized ($f(\emptyset) = 0$), monotone ($f(X) \leq f(Y)$ if $X \subseteq Y$), and submodular ($f(X) + f(Y) \geq f(X \cap Y) + f(X \cup Y)$)



# Standard Submodular Maximization

## Two Basic Algorithms

- Greedy Algorithm
  (easy to analyze; easy to extend; used in practice)
- Continuous Greedy Algorithm
  (very strong; used in theory)

## Greedy Algorithm for Submodular Maximization

Example: cardinality constraint (select at most $k$ elements)

- Start from $X = \emptyset$
- Select $e$ such that $f(X \cup \{e\})$ is maximized
- Update $X \leftarrow X \cup \{e\}$ and repeat $k$ times



## Analysis

$$\begin{aligned} f(X^*) - f(X_t) &\le f(X_t \cup X^*) - f(X_t) = f(X_t \cup \{e_1^*, \dots, e_l^*\}) - f(X_t) \\ &\le f(X \cup \{e_1^*, \dots, e_l^*\}) - f(X \cup \{e_1^*, \dots, e_{l-1}^*\}) \\ &+ f(X_t \cup \{e_1^*, \dots, e_{l-2}^*\}) - f(X_t \cup \{e_1^*, \dots, e_{l-3}^*\}) + \dots \\ &\le f(X_t \cup \{e_{l-1}^*\}) - f(X) \\ &+ f(X_t \cup \{e_{l-2}^*\}) - f(X) + \dots \\ &\le k(f(X_{t+1}) - f(X_t)) \end{aligned}$$

thus

$$\begin{aligned} f(X^*) - f(X_k) &\le (1 - 1/k)\,(f(X^*) - f(X_{k-1})) \\ &\le e^{-1}(f(X^*) - f(\emptyset)) \\ \Rightarrow\ f(X_k) &\ge (1 - 1/e) f(X^*) \end{aligned}$$



## Constraints Solvable by the Greedy Algorithm

- Cardinality: $1 - 1/e$ approx
- Matroid: $1/2$ approx
- Knapsack: $1 - 1/e$ approx (with partial enumeration)
- ...

Typicaly, it is for constraints suitable for local search



## Continuous Greedy Algorithm for Submodular Maximization

Extend $f\colon 2^E \to \mathbb{R}$ to $F\colon [0,1]^E \to \mathbb{R}$ by

$$F(x) = \mathbb{E}_{X \sim x}[f(X)] = \sum_{Z \subseteq E} \prod_{e \in Z} x_e \prod_{e' \in E \setminus Z} (1 - x_{e'}) f(Z)$$

Then, perform the Frank–Wolfe algorithm

Example: $P$: any convex set

- Start from $x = 0$
- Select $v \in P$ such that $\langle \nabla F(x), v \rangle$ is maximized
- Update $x \leftarrow x + \epsilon v$ and repeat $1/\epsilon$ times

After obtaining the continuous solution, we have to round the solution to obtain a discrete solution

## Analysis

The continuous greedy algorithm solves the following differential equation

$$\frac{dx}{dt} \in \operatorname{argmax}\langle \nabla F(x), v \rangle$$

$$\begin{aligned}
\frac{d}{dt}F(x) &= \max_{v \in P} \langle \nabla F(x), v \rangle \\
&\geq \langle \nabla F(x), x^* \vee x - x \rangle \\
&\geq F(x^* \vee x^*) - F(x) \geq F(x^*) - F(x) \\
\Rightarrow F(x) &\geq (1 - e^{-t}) F(x^*)
\end{aligned}$$

Here, we used (1) $x^* \vee x - x \in P$; and (2) $F$ is concave along the positive direction (even if it is continuously extended)



## Constraints Solvable by Continuous Greedy Algorithm

Continuous submodular maximization on a convex set is always solvable within approx factor of $1 - 1/e$

Rounding part depends on constraints

- Matroid: $1 - 1/e$
- Multiple Knapsack: $1 - 1/e$ with partial enumeration
- ...
- Very generic framework: contention resolution scheme (good approximation if $P$ has a low correlation gap)



## Current Situation on This Area

Basic theory is completed

**Greedy Algorithm**

More efficient algorithm (randomized, etc)
Parallerization
...

**Continuous Greedy Algorithm**

Rounding method better than CR-scheme
Practical implementation
Hardness
...



## Submodular Maximization on Logic

## Setting

We consider the setting that the constraint is represented by a **logic formula** $\phi(X)$ **or** $\phi(x_1, \ldots, x_k)$ **on a graph** $G$ i.e., we consider

$$\text{maximize } f(U) \text{ subject to } G \models \phi(U)$$

**Example 1**: independent set
- $\phi(X) \equiv \forall x, y \in X, \neg e(x, y)$

**Example 2**: $s$-$t$ path
- $\phi(X) \equiv \mathrm{stconn}(X) \wedge \forall Y \subsetneq X, \neg\mathrm{stconn}(Y)$
- $\mathrm{stconn}(X) \equiv s, t \in X, \forall S \subsetneq X, \exists u \in S, \exists v \in X \setminus S, e(u, v)$

**Example 3**: $k$-clique
- $\phi(x_1, \ldots, x_k) \equiv e(x_1, x_2) \wedge \cdots \wedge e(x_{k-1}, x_k)$

The above examples have **large correlation gap**; thus, these cannot be solved by the greedy algorithms



## Class of Logic Formula

Commonly used logic classes

- First Order Logic (FO): predicate logic ($=, \wedge, \vee, \neg, \forall, \exists$) with **vertex variables** $x$ and **adjacency predicate** $e(x, y)$.
  It represents *relative positions of constantly many vertices.*
  - OK: "size $k$ independent set" is in FO
  - NG1: "arbitrary size independent set" is not in FO
  - NG2: "connected" is not in FO
- Monadic Second Order Logic ($\mathrm{MSO}_1$): FO + **vertex subset variables** $X$ and **membership predicate** $x \in X$
  It can represent many combinatorial problems
  - OK: "connected" is in MSO
  - NG1: "cardinality constraint" is not in MSO
  - NG2: "arbitrary many partitions" is not in MSO



## Graph Class

Even the logic classes are restricted, the problem is **still hard** (even just finding a feasible solution)

- **three coloring problem** (in MSO) cannot be solved in polynomial time unless P = NP
- $k$-**clique problem** (in FO) cannot be solved in $O(n^{k-\epsilon})$ time unless W[1] = FPT

$\Rightarrow$ We also need to restrict graph classes

---

A framework to handle problems described by (logic class) $\times$ (graph class) is called algorithmic metatheorem



## Algorithmic Metatheorem (1/2)

Algorithmic Metatheorem is a subfield in combinatorics & model theory. Also, it refers a type of theorems:

> *If a problem is described in a certain logic and the inputs are structured in a certain way, then the problem can be solved with a certain amount of resources [Tantau'16]*

### Classical Results

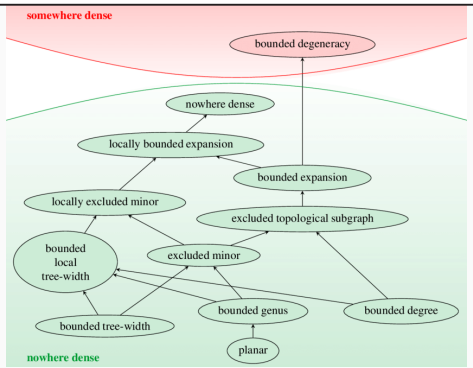
- We can test $G \models \phi$ in linear time if $\phi$ **is a MSO sentence** and $G$ **has bounded treewidth** [Courcelle'90]; it is *best possible* for minor closed graph class. Optimization by [Arnborg–Lagergren–D. Seese'91]
- We can test $G \models \phi$ in linear time if $\phi$ **is a FO sentence** and $G$ **has bounded degree** [Seese'96]

## Algorithmic Metatheorem (2/2)

**Recent Results**: extend graph classes / query problems

- $G \models \phi$ in linear time if $\phi$ **is a FO sentence** and $G$ **has bounded expansion** [Dvorak+, FOCS'10]
- $G \models \phi$ in quasi-linear time if $\phi$ **is a FO sentence** and $G$ **is nowhere dense** [Grohe+, STOC'14]; it is *best possible* for subgraph closed graph class
- **Count** $\{U : G \models \phi(U)\}$ in linear time if $\phi(X)$ **is a MSO formula** and $G$ **has bounded degree** [Kazana-Segoufin'11]
- **Count** $\{(u_1, \ldots, u_k) : G \models \phi(u_1, \ldots, u_k)\}$ in linear time if $\phi(x_1, \ldots, x_k)$ **is a FO formula** and $G$ **has bounded expansion** [Kazana-Segoufin, PODS'13]
- **Count** $|\{X : G \models \phi(X)\}|$ in quasi-linear time if $\phi(x_1, \ldots, x_k)$ **is a FO formula** and $G$ **is nowhere dense** [Schweikardt+, PODS'18]






(Grohe-Kreutzer-Siebert'14)



## Our Results: Algorithmic Metatheorem for Submodular Max.

For nonnegative monotone submodular $f$,

**Theorem (MSO + bounded treewidth)**

$\max f(U)$ s.t. $G \models \phi(U)$ is solved in $n^{O(1)}$ time with approximation factor of $O(\log n)$ if $\phi$ **is a MSO formula** and $G$ **has bounded treewidth** / **+cardinality constraint** is solved in $n^{O(\log n)}$ time.

**Theorem (FO + low degree)**

$\max f(\{u_1, \ldots, u_k\})$ s.t. $G \models \phi(u_1, \ldots, u_k)$ is solved in $O(n^{1+\epsilon})$ time with approximation factor of 2 if $\phi$ **is a FO formula** and $G$ **has low degree** (i.e., maximum degree $\leq \max\{\text{const}, n^\epsilon\}$ for all $\epsilon > 0$

**Theorem (FO + bounded expansion)**

$\max f(\{u_1, \ldots, u_k\})$ s.t. $G \models \phi(u_1, \ldots, u_k)$ is solved in $n^{O(\log k)}$ time with approximation factor of $O(\log k)$ if $\phi$ **is a FO formula** and $G$ **has bounded expansion**; $O$ only suppresses constants indpendent to $k$



## Main Difficulty

**Well-solvable discrete structures**

- **convexity** ($\rightarrow$ greedy algorithm)
- **enumerability** ($\rightarrow$ dynamic programming algorithm)

- **Algorithmic metatheorem** is obtained by **dynamic programming** (recall: DP on tree decomposition)
- However, **submodular maximization** has been solved by **greedy algorithms** — **DP has never been succeeded**
  DP merges "equivalent" solutions to reduce the search space; however, the values of the function depends on the solutions

Resolving this mismatch is the central difficulty

## Proof Strategy

All theorems are proved by the same strategy

- Convert the logic formula into a **tractable normal form**
- Perform **greedy algorithm** multiple times

Intuition: instead of memoizing subsolutions, we perform greedy algorithm multiple times to subproblems

- MSO + bounded treewidth: **tree automaton** (equiv., **decomposable negation normal form**)
- FO + low degree: **Gaifman normal form**
- FO + bounded expansion: **forest normal form** (ours)

---

This idea is later used in deriving $O(\mathrm{poly}(n)B)$ algorithm for automaton-constrained tree knapsack problem (cf: standard DP takes $O(nB^2)$ time) [Kumabe–Maehara–Sin'ya, WALCOM'19])



## MSO + Bounded Treewidth

## Setting: Independent Set Problem

We illustrate the method on the following specific setting:

- $\phi$ represents the **independent set constraint** $\phi(X) = \forall x \in X, \forall y \in X, \neg e(x, y)$
- $G$ is a **balanced binary tree** (i.e., the number of vertices of a subtree is at most $n/2$)

(nb: in this case, we can obtain 2-approximate solution: find a two coloring and output a color class with larger objective value)



## Recall: Linear Maximization

If the **objective function is linear**, we can solve the problem by the following dynamic programming

1. Let $r \in V$ be the root of the tree
2. **Select** $r$, and compute an optimal solution of subtrees recursively (i.e., the roots of subtrees cannot be used)
3. **Does not select** $r$, and compute an optimal solution of subtrees recursively (i.e., the roots of subtrees can be used)
4. Output better solutions in 2 and 3.

Complexity: $O(n)$ by memoizing solutions

Correctness: $f(\mathrm{left} \cup \mathrm{right}) = f(\mathrm{left}) + f(\mathrm{right})$.

## Why this does not work for submodular maximization?

1. Let $r \in V$ be the root of the tree
2. **Select** $r$, and compute an optimal solution of subtrees recursively (i.e., the roots of subtrees cannot be used)
3. **Does not select** $r$, and compute an optimal solution of subtrees recursively (i.e., the roots of subtrees can be used)
4. Output better solutions in 2 and 3.

Correctness fails. Even $f(\text{left})$ and $f(\text{right})$ are large, $f(\text{left} \cup \text{right})$ can be small i.e., the problem is not decomposed into subproblems



## Our Approach: Recursive Greedy Algorithm

1. Let $r \in V$ be the root of the tree
2. **Select** $r$
   - 2.1 Solve the problem on the left subtree, where the function is $f(\cdot \cup \{r\}) - f(\{r\})$. Let $U_1$ be the solution
   - 2.2 Solve the problem on the right subtree, where the function is modified to $f(\cdot \cup \{r\} \cup U_1) - f(\{r\} \cup U_1)$
3. **Does not select** $r$
   - 3.1 Solve the problem on the left subtree. Let $U_1$ be the solution
   - 3.2 Solve the problem on the right subtree, where the function is modified to $f(\cdot \cup U_1) - f(U_1)$
4. Output better solutions in 2 and 3.

*We do not memoize the subsolutions.* Instead, *we use the left-subsolution in a greedy manner.*



## Complexity

The algorithm recursively calls itself for each subtree twice (one passes for each select/do-not-select $r$)

$$T(n) \leq 2(T(n_1) + T(n_2)) + O(n)$$

Since we assumed that the tree is balanced, $n_1 \leq n/2$, $n_2 \leq n/2$,

$$T(n) \lesssim 4T(n/2) + O(n)$$

The solution of this equation is $T(n) = O(n^2)$; polynomial time!



## Approximation Factor

Consider the case that an optimal solution **does not** contain $r$. Let $U_1^*, U_2^*$ be the left and right solutions in an optimal solution, and $U_1, U_2$ be the left and right solutions obtained by the algorithm when $r$ is not selected.

Let $\alpha(n)$ be the approximation factor. Then,

$$\alpha(n/2) f(U_1) \geq f(U_1^*)$$
$$\alpha(n/2)(f(U_1 \cup U_2) - f(U_1)) \geq f(U_1 \cup U_2^*) - f(U_1)$$

Adding these inequalities, we have

$$\begin{aligned}\alpha(n/2) f(U_1 \cup U_2) &\geq f(U_1 \cup U_2^*) - f(U_1) \cup f(U_1^*) \\ &\geq f(U_1 \cup U_1^* \cup U_2^*) - f(U_1) + f(U_1 \cap U_1^*) \\ &\geq f(U_1^* \cup U_2^*) - f(U_1 \cup U_2)\end{aligned}$$

Thus, we have $\alpha(n) \leq \alpha(n/2) + 1 \leq \cdots \leq \log n$.

## Generalization to General Setting (1/3)

Generalization 1: **Balanced** binary tree → **General** binary tree

Use centroid decomposition

**Theorem [Jordan'1869]**
For any binary tree $T$, there exists vertex $u$ such that, each subtee obtained by removing $u$ has at most $2n/3$ vertices

The algorithm tries to select / non-select the centroid and call the algorithm recursively on each subtrees. This only changes the bases of logarithms from $\log_2$ to $\log_{3/2}$ in the complexity & approx factor.



## Generalization to General Setting (2/3)

Generalization 2: Binary tree → Bounded treewidth graph

Use nice tree-decomposition

**Theorem [Klosk'94]**
For any bounded treewidth graph $G$, there exists a tree decomposition whose structure tree is binary (& some nice properties)

Such decomposition is obtained in $O(n)$ time [Bodlaender'93]

Instead of trying "select or not", we try **all the subsets ($2^w$)** in each bag. This changes the recursion of the complexity as

$$T(n) \leq 2^w(T(n_1) + T(n_2)) + O(n) \lesssim 2^{w+1}T(n/2) + O(n)$$

The solution is $T(n) = O(n^{w+1}) = n^{O(1)}$ for bounded treewidth.



## Generalization to General Setting (3/3)

Generalization 1: independent set → MSO formula

Use tree automaton to represent a constraint

**Theorem [Thatcher–Wright'68]**
For any MSO formula $\phi$, there exists a **(bottom-up) tree automaton** $\mathcal{A}$ such that a binary tree $T$ satisfies $T \models \phi$ if and only if $\mathcal{A}$ accepts $T$.

Bottom-up Tree Automaton $\mathcal{A} = (Q, A, \delta, F)$; $\delta\colon Q \times Q \times A \to Q$



## Tree Automaton (1/3)

The transition of a tree automaton is given by

$\delta$(left child's state, right child's state, alphabet) = current state

Example: Independent set:

- $\delta$(arbitrary, arbitrary, not-select) = not-selected
- $\delta$(not-selected, not-selected, select) = selected
- other choices make invalid state

## Tree Automaton (2/3)

[Thatcher–Wright'68] showed that a subset of a binary tree is recognized by a tree automaton iff it is represented by a MSO formula

In the independent set, we tried two states (selected / not-selected). In general case, we try all automaton state. This changes the complexity as

$$T(n) \lesssim 2|\mathcal{A}|^w T(n/2) + O(n)$$

The solution is $T(n) = n^{O(w \log |\mathcal{A}|)}$



## Tree Automaton (3/3)

For a given MSO formula $\phi$ and upperbound of treewidth $w$, we can construct the tree automaton $\mathcal{A}$ (i.e., it is decidable)

The size of the automaton $|\mathcal{A}|$ can be $\Omega(2^{2^{\cdot^{\cdot^{2}}}})$, i.e., the tower of powers of length $|\phi|$

However, this is a constant if we fix $\phi$!

(nb: all results in this study have such galactic constants)



## Conclusion

Monotone Submodular Maximization on Logic Constraint

| Setting | Time | Approx |
|---|---|---|
| MSO + Bounded Treewidth | $n^{O(1)}$ | $O(\log n)$ |
| FO + Low degree | $n^{1+\epsilon}$ | 2 |
| FO + Bounded Expansion | $n^{O(\log k)}$ | $O(\log k)$ |

General strategy:

- Convert the formula into the normal Form (Automaton / Gaifman / Forest)
- Perform greedy algorithm in structured manner

preprint: `https://arxiv.org/abs/1807.04575`

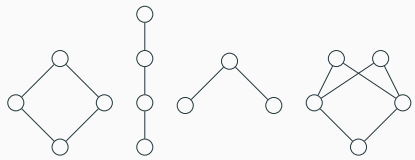


## Submodular Maximization on Lattice

## Lattice

**Lattice**: partially ordered set closed under $\vee$ (join = least upper bound) and $\wedge$ (meet = greatest lower bound)

- Set lattice. $X, Y \subseteq E$. $X \leq Y$ if $X$ is a subset of $Y$
- Integer lattice. $X, Y \in \mathbb{Z}^n$. $X \leq Y$ if $X_i \leq Y_i$ for all $i$
- Vector lattice. $X, Y \subseteq \mathbb{R}^d$. $X \leq Y$ if $X$ is a subspace of $Y$
- Subgroup lattice. $X, Y$ groups. $X \leq Y$ if $X$ is a subgroup of $Y$





## Basics of Lattice

- $e$ is join-irreducible if $e$ cannot be represented as a join of elements
  - set lattice: join irreducible iff singleton
  - vector lattice: join irreducible iff 1-dim subspace
- $e$ is admissible wrt. $X$ if there is no element $e' < e$ such that $X \vee e' \neq X$
  - set lattice: $e$ is admissible wrt $X$ iff $e \notin X$
  - vector lattice: $e$ is admissible wrt $X$ iff $e$ is linearly independent to $X$

We use small letters as join-irreducibles and capital letters as general elements



## Motivating Example: Principal Component Analysis (PCA)

Given $n$ data points $v_1, \ldots, v_N \in \mathbb{R}^d$. Find a subspace of dimension $k$ that approximates the data, i.e., maximize the following function over $k$-dimensional subspace $X$

$$f(X) = \sum_i \|\Pi_X v_i\|^2$$

- Fundamental procedure in data mining
- Exactly solvable by the greedy algorithm

**Question**

Why PCA is solvable by the greedy algorithm?

**I want to say ...**

$f$ is a "submodular function" on a vector lattice



## Difficulty: Definition of Submodularity

How to define submodularity on lattice?

**Lattice Submodularity**

$$f(X) + f(Y) \geq f(X \vee Y) + f(X \wedge Y)$$

but the PCA function does not satisfy this property

**Extended Diminishing Return Property**

$$f(X \vee e) - f(X) \geq f(Y \vee e') - f(Y)$$

for $X \leq Y$, $e \leq e'$ join irreducibles, $X \vee e \succeq X$, $Y \vee e' \succeq Y$

This is weaker than the lattice submodularity on lattice but still PCA function does not satisfy this property

## Counterexample

$N = 1$, $v_1 = (1, 0)$

$X = 0$; $Y = \mathrm{span}(0, y)$; $e = \mathrm{span}(\epsilon, 1)$; $e' = e$

Because $e$ and $v_1$ are almost orthogonal, $f(X \cup e) - f(X) \approx \epsilon$

But $e$ and $Y$ can span the full space: $f(Y' \cup e) - f(Y) = 1$
even though $e$ and $Y$ are *nearly collinear*



## New Definition: Directional DR submodularity

[Nakashima, TM; AAAI'18]

For any $X \leq Y$ and $e_y \in \mathrm{adm}(Y)$, there exists $e'_y \in \mathrm{adm}(Y)$ with $Y \vee e_y = Y \vee e'_y$ such that for all $e_x \in \mathrm{adm}(X)$ with $e_x \leq e'_y$,

$$f(X \vee e_x) - f(X) \geq f(Y \vee e_y) - F(Y)$$

Intuition: it allows us to change the basis

**Proposition**
The PCA function is directional DR submodular

$N = 1$, $v_1 = (1, 0)$; $X = 0$; $Y = \mathrm{span}(0, y)$; $e_y = \mathrm{span}(\epsilon, 1)$

We can choose $e'_y = \mathrm{span}(1, 0)$ since $Y \vee e_y = Y \vee e'_y$

Then, $e_x = e'_y$; thus $f(X \vee e_x) - f(X) = 1$



## Maximization of Directional DR Submod ft.

The greedy algorithm on set lattice can be extended (but need much more complicated analysis)

Example: Rank constraint (rank = shortest distance from the bottom)

- $X = \bot$
- Find $e \in \mathrm{adm}(X)$ such that $f(X \vee e)$ is maximized
- Update $X \leftarrow X \vee e$ and repeat if $|X \vee e| \leq k$

**Theorem**
If the lattice is a modular lattice, the above algorithm gives $1 - 1/e$ approximate solution

modular lattice: $|X \vee e| = |X| + 1$ for all $e \in \mathrm{adm}(X)$

Similar result holds for (a variant of) knapsack constraint



## Continuous Relaxation on Lattice

To obtain better approximation result, we may need a "continuous relaxation"

- Set lattice $2^E \Rightarrow$ Cube $[0, 1]^E$
- General lattice $\Rightarrow$ ???

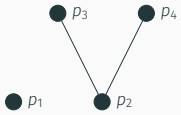
What is the corresponding geometric object?

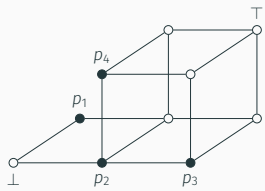
Current Result: Theory on Distributive Lattice

## Cubical Complex

By the Birkhoff representation theorem, the continuous relaxation may be the cubical complex

In each cubical complex, the continuous relaxation $F$ is naturally defined



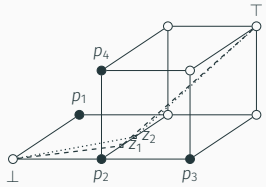



## Difficulty

In the analysis of continuous greedy algorithm, we used "concavity along the positive directions"

What is the "positive direction" if the line passes many cubes?

(Note: the geodesic curve does not satisfy the requirement)






## New "Geodesic" Structure

Requirement: For any $x, y \in P$ with $x \leq y$, there exists a curve between $x$ to $y$ such that $F$ is concave along the curve

**Theorem**

For any $x \leq y$, there **uniquely** exists such curve. The curve is characterized by

- Equilibrium of a cooperative game with infinitely many agents (go from $x$ to $y$ in the same speed)
- Solution to a convex minimum cost flow problem (maximize the time of the slowest agent)

Mathematically, the space with this curve forms a "metric modular" (studied in metric geometry / abstract convexity)



## Result

The continuous greedy algorithm can be extended on the cubical complex with the curve

$\Rightarrow 1 - 1/e$ approximation for multiple knapsack constraint

## Conclusion (lattice)

Greedy Algorithm and Continuous Greedy Algorithm are extended

- Greedy for modular lattices
  - New concept: Directional DR-Submodularity
  - Basics for subspace selection problem (incl. PCA)
  - Rank constraint $1 - 1/e$, Knapsack constraint $(1 - 1/e)/2$
- Continuous greedy for distributive lattice
  - Continuous object for lattice $\Rightarrow$ cubical complex
  - New "curve" structure $\Rightarrow$ metric modular
  - Multiple knapsack constraint $1 - 1/e$



## Conclusion

Basic theory of submodular maximization is completed
$\Rightarrow$ we are seeking more interesting settings

- logic
- lattice
- metric modular
- ...

Topological and combinatorial methods in symmetric motion planning

Kohei Tanaka

Shinshu University

Sep. 1, 2019

ALGI30



## Topological complexity



## Topological complexity

**Problem 1.1**

*A robot moves around in a space $X$. We would assign a path $\gamma_{xy}\colon I = [0,1] \to X$ starting at $x$ and ending at $y$ to each pair $(x, y)$ of points in $X$. This is equivalent to giving a section $X \times X \to X^I$ (not necessarily continuity) of the path-fibration $X^I \to X \times X$, and we call it* motion planning *on $X$.*
*Can we always take a motion planning?*



## Topological complexity

**Problem 1.1**

*A robot moves around in a space $X$. We would assign a path $\gamma_{xy}\colon I = [0,1] \to X$ starting at $x$ and ending at $y$ to each pair $(x, y)$ of points in $X$. This is equivalent to giving a section $X \times X \to X^I$ (not necessarily continuity) of the path-fibration $X^I \to X \times X$, and we call it* motion planning *on $X$.*
*Can we always take a motion planning?*

**Proposition 1.2**

*A space $X$ admits a motion planning if and only if $X$ is path-connected.*

## Topological complexity

**Problem 1.1**

*A robot moves around in a space $X$. We would assign a path $\gamma_{xy} : I = [0,1] \to X$ starting at $x$ and ending at $y$ to each pair $(x, y)$ of points in $X$. This is equivalent to giving a section $X \times X \to X^I$ (not necessarily continuity) of the path-fibration $X^I \to X \times X$, and we call it* motion planning *on $X$.*
*Can we always take a motion planning?*

**Proposition 1.2**

*A space $X$ admits a motion planning if and only if $X$ is path-connected.*

In this study, we focus on *continuous motion planning*, i.e. a section must be continuous (with the compact open topology on $X^I$).



## Topological complexity

**Problem 1.1**

*A robot moves around in a space $X$. We would assign a path $\gamma_{xy} : I = [0,1] \to X$ starting at $x$ and ending at $y$ to each pair $(x, y)$ of points in $X$. This is equivalent to giving a section $X \times X \to X^I$ (not necessarily continuity) of the path-fibration $X^I \to X \times X$, and we call it* motion planning *on $X$.*
*Can we always take a motion planning?*

**Proposition 1.2**

*A space $X$ admits a motion planning if and only if $X$ is path-connected.*

In this study, we focus on *continuous motion planning*, i.e. a section must be continuous (with the compact open topology on $X^I$).

**Example 1**

Let $X$ be a convex set. We can take the shortest path spanning any two points in $X$. This is a continuous motion planning.



**Proposition 1.3**

*A space $X$ admits a continuous motion planning if and only if $X$ is contractible.*



**Proposition 1.3**

*A space $X$ admits a continuous motion planning if and only if $X$ is contractible.*

**Definition 1.4**

*For a continuous map (fibration) $p : X \to B$, the* **Schwartz genus** $\mathrm{SG}(p)$ *is the smallest number of open sets $U_1, \cdots, U_n$ covering $B$ admitting a homotopy local section $s_i : U_i \to X$ of $p$ for each $i$ ($p \circ s_i$ is homotopic to the inclusion on $U_i$).*

**Proposition 1.3**

*A space $X$ admits a continuous motion planning if and only if $X$ is contractible.*

**Definition 1.4**

*For a continuous map (fibration) $p : X \to B$, the* **Schwartz genus** $\mathrm{SG}(p)$ *is the smallest number of open sets $U_1, \cdots, U_n$ covering $B$ admitting a homotopy local section $s_i : U_i \to X$ of $p$ for each $i$ ($p \circ s_i$ is homotopic to the inclusion on $U_i$).*

**Definition 1.5 (Unreduced version of TC)**

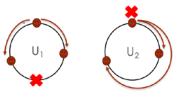
*For a space $X$, the* **topological complexity** $\mathrm{TC}(X)$ *of $X$ is defined as* $\mathrm{SG}(\pi)$ *for the path fibration $\pi : X^I \to X \times X$*



**Example 1.6**

$\mathrm{TC}(S^1) = 2$.

Consider open sets $U_1 = \{(x, y) \mid x \neq -y\}$ and $U_2 = \{(x, y) \mid x \neq y\}$ of $S^1 \times S^1$. We can choose a continuous motion planning on $U_1$ by the shortest path (arc). On the other hand, $U_2$ admits a continuous motion planning by the (anti)clockwise path.

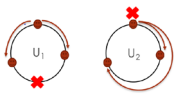





**Example 1.6**

$\mathrm{TC}(S^1) = 2$.

Consider open sets $U_1 = \{(x, y) \mid x \neq -y\}$ and $U_2 = \{(x, y) \mid x \neq y\}$ of $S^1 \times S^1$. We can choose a continuous motion planning on $U_1$ by the shortest path (arc). On the other hand, $U_2$ admits a continuous motion planning by the (anti)clockwise path.




**Fact 1.7 (Fundamental properties on TC (for nice spaces))**

1. $\mathrm{TC}$ *is a homotopy invariant. Moreover,* $\mathrm{TC}(X) = 1$ *iff* $X \simeq *$.
2. *LS-categorical estimate :* $\mathrm{cat}(X) \leq \mathrm{TC}(X) \leq \mathrm{cat}(X \times X)$.
3. *Cohomological lower bound : zero-divisor-cup-length* $\leq \mathrm{TC}$.
4. *Product formula :* $\mathrm{TC}(X \times Y) \leq \mathrm{TC}(X) + \mathrm{TC}(Y) - 1$.



## Grant-Farber's symmetrization

Symmetric (continuous) motion planning (rough idea):

1. $\gamma_{xy} = \overline{\gamma}_{yx} (= \gamma_{xy}(1 - t))$.
2. $\gamma_{xx}$ must be the constant path(loop) on $x$.

## Grant-Farber's symmetrization

Symmetric (continuous) motion planning (rough idea):

1. $\gamma_{xy} = \overline{\gamma}_{yx} (= \gamma_{xy}(1-t))$.
2. $\gamma_{xx}$ must be the constant path(loop) on $x$.

**Definition 1.8**

1. $F_2(X) := X \times X - \Delta_X$ *and* $C_2(X) := F_2(X)/\mathbb{Z}_2$
2. $L(X) := \{\gamma \in X^I \mid \gamma(0) = \gamma(1)\}$ *and* $X^I_L := X^I - L(X)$ *and* $Q(X) := X^I_L/\mathbb{Z}_2$.
3. *The induced fibration* $\rho : Q(X) \to C_2(X)$.
4. *The* **symmetric topological complexity** $\mathrm{TC}^\Sigma(X)$ *is defined as* $\mathrm{SG}(\rho) + 1$.



**Fact 1.9 (Fundamental properties on $\mathrm{TC}^\Sigma$)**

1. $\mathrm{TC}^\Sigma$ *is* **Not** *a homotopy invariant.*
2. $\mathrm{TC}^\Sigma(*) = 1$ *and* $\mathrm{TC}^\Sigma(X) = 2$ *iff* $X \simeq *$ *and* $X \neq *$.
3. $\mathrm{TC}(X) \leq \mathrm{TC}^\Sigma(X)$.
4. $\mathbb{Z}_2$*-cohomological lower bound : cup-length*$(N) + 2 \leq \mathrm{TC}^\Sigma(X)$ *for* $N \subset H^*(X; \mathbb{Z}_2) \otimes H^*(X; \mathbb{Z}_2)$.



**Fact 1.9 (Fundamental properties on $\mathrm{TC}^\Sigma$)**

1. $\mathrm{TC}^\Sigma$ *is* **Not** *a homotopy invariant.*
2. $\mathrm{TC}^\Sigma(*) = 1$ *and* $\mathrm{TC}^\Sigma(X) = 2$ *iff* $X \simeq *$ *and* $X \neq *$.
3. $\mathrm{TC}(X) \leq \mathrm{TC}^\Sigma(X)$.
4. $\mathbb{Z}_2$*-cohomological lower bound : cup-length*$(N) + 2 \leq \mathrm{TC}^\Sigma(X)$ *for* $N \subset H^*(X; \mathbb{Z}_2) \otimes H^*(X; \mathbb{Z}_2)$.

**Example 2**

$\mathrm{TC}^\Sigma(S^1) = 3$. Moreover, $\mathrm{TC}^\Sigma(S^n) = 3$ for any $n \geq 1$.



**Fact 1.9 (Fundamental properties on $\mathrm{TC}^\Sigma$)**

1. $\mathrm{TC}^\Sigma$ *is* **Not** *a homotopy invariant.*
2. $\mathrm{TC}^\Sigma(*) = 1$ *and* $\mathrm{TC}^\Sigma(X) = 2$ *iff* $X \simeq *$ *and* $X \neq *$.
3. $\mathrm{TC}(X) \leq \mathrm{TC}^\Sigma(X)$.
4. $\mathbb{Z}_2$*-cohomological lower bound : cup-length*$(N) + 2 \leq \mathrm{TC}^\Sigma(X)$ *for* $N \subset H^*(X; \mathbb{Z}_2) \otimes H^*(X; \mathbb{Z}_2)$.

**Example 2**

$\mathrm{TC}^\Sigma(S^1) = 3$. Moreover, $\mathrm{TC}^\Sigma(S^n) = 3$ for any $n \geq 1$.

**Theorem 1.10 (Gonzáles and Landweber, 2009)**

*$TC^\Sigma(\mathbb{R}P^r)$ is related to the Euclidean embedding dimension $E(r)$ of the projective space $\mathbb{R}P^r$, as $TC^\Sigma(\mathbb{R}P^r) = E(r) + 1$ for $r \in \{1, 2, 4, 8, 9, 13\}$, $r > 15$.*

**Definition 2.1 (Classifying spaces)**

*Let $P$ be a finite poset. The* **classifying space** *(order complex) $\mathcal{B}P$ is (the realization of) a simplicial complex that consists of totally ordered subsets of $P$. An $n$-simplex of the classifying space corresponds to a totally ordered sequence of $n+1$ distinguished elements in $P$:*

$$p_0 < p_1 < p_2 < \cdots < p_n.$$



**Definition 2.1 (Classifying spaces)**

*Let $P$ be a finite poset. The* **classifying space** *(order complex) $\mathcal{B}P$ is (the realization of) a simplicial complex that consists of totally ordered subsets of $P$. An $n$-simplex of the classifying space corresponds to a totally ordered sequence of $n+1$ distinguished elements in $P$:*

$$p_0 < p_1 < p_2 < \cdots < p_n.$$

**Definition 2.2 (Face poset)**

*Let $X$ be a simplicial complex (or CW-complex). The face poset $\mathcal{F}(X)$ consists of simplices (cells) of $X$ with the inclusion order.*



**Definition 2.1 (Classifying spaces)**

*Let $P$ be a finite poset. The* **classifying space** *(order complex) $\mathcal{B}P$ is (the realization of) a simplicial complex that consists of totally ordered subsets of $P$. An $n$-simplex of the classifying space corresponds to a totally ordered sequence of $n+1$ distinguished elements in $P$:*

$$p_0 < p_1 < p_2 < \cdots < p_n.$$

**Definition 2.2 (Face poset)**

*Let $X$ be a simplicial complex (or CW-complex). The face poset $\mathcal{F}(X)$ consists of simplices (cells) of $X$ with the inclusion order.*

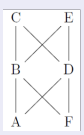
**Fact 2.3**

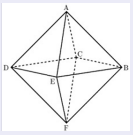
*Every finite CW complex $X$ has the same homotopy type of the classifying space of a finite poset.*



**Example 2.4**

*Let $P$ be a finite poset consisting of 6-points described as the following left-hand Hasse diagram:*




*The classifying space is the hollow octahedron in the above right-hand. We notice that $P$ is the face poset of 2-sphere $S^2$ with the minimal regular cell decomposition. The classifying space $\mathcal{B}(\mathcal{F}(S^2))$ is the barycentric subdivision of the original $S^2$.*

## TC for finite posets and classifying spaces

**Definition 2.5**

*For a finite poset $P$ and $m \geq 0$,*

- *The finite interval $J_m = 0 < 1 > 2 < 3 > \cdots < (>)m$ with length $m$.*
- *The path space $P^{J_m} = \{J_m \to P\} = \{p_0 \leq p_1 \geq p_2 \leq \cdots \leq (\geq)p_m\}$ with length $m$.*
- *$\pi_m : P^{J_m} \to P \times P$ is defined by $\pi_m(\gamma) = (\gamma(0), \gamma(m))$.*
- *$\mathrm{TC}_m(P)$ is the minimal number $n \geq 0$ such that there exist open sets $U_0, \cdots, U_n$ covering $P \times P$ with a local section $U_i \to P^{J_m}$ of $\pi_m$ for each $i$.*



## TC for finite posets and classifying spaces

**Definition 2.5**

*For a finite poset $P$ and $m \geq 0$,*

- *The finite interval $J_m = 0 < 1 > 2 < 3 > \cdots < (>)m$ with length $m$.*
- *The path space $P^{J_m} = \{J_m \to P\} = \{p_0 \leq p_1 \geq p_2 \leq \cdots \leq (\geq)p_m\}$ with length $m$.*
- *$\pi_m : P^{J_m} \to P \times P$ is defined by $\pi_m(\gamma) = (\gamma(0), \gamma(m))$.*
- *$\mathrm{TC}_m(P)$ is the minimal number $n \geq 0$ such that there exist open sets $U_0, \cdots, U_n$ covering $P \times P$ with a local section $U_i \to P^{J_m}$ of $\pi_m$ for each $i$.*

**Lemma 2.6**

$\mathrm{TC}_m(P) \geq \mathrm{TC}_{m+1}(P)$.

Let $\mathrm{TC}(P)$ denote the minimal number of $\mathrm{TC}_m(P)$ (or $\lim_{m\to\infty} \mathrm{TC}_m(P)$).



**Example 2.7**

*Let $P$ be a finite poset which consists of 4-points. The classifying space $\mathcal{B}P$ is a circle $S^1$. $\mathrm{TC}(\mathcal{B}P) = 2$ but $\mathrm{TC}(P) = 4$.*



**Example 2.7**

*Let $P$ be a finite poset which consists of 4-points. The classifying space $\mathcal{B}P$ is a circle $S^1$. $\mathrm{TC}(\mathcal{B}P) = 2$ but $\mathrm{TC}(P) = 4$.*

One of the ideas to fill the gap between $\mathrm{TC}(P)$ and $\mathrm{TC}(\mathcal{B}P)$ is to consider the barycentric subdivisions of finite posets.

**Example 2.7**

*Let $P$ be a finite poset which consists of 4-points. The classifying space $\mathcal{B}P$ is a circle $S^1$. $\mathrm{TC}(\mathcal{B}P) = 2$ but $\mathrm{TC}(P) = 4$.*

One of the ideas to fill the gap between $\mathrm{TC}(P)$ and $\mathrm{TC}(\mathcal{B}P)$ is to consider the barycentric subdivisions of finite posets.

**Definition 2.8**

*For a finite poset $P$, the barycentric subdivision $\mathrm{sd}(P) = \mathcal{F}(\mathcal{B}P)$ is the face poset of the classifying space. We can define inductively the $k$-th iterated barycentric subdivision $\mathrm{sd}^k(P) = \mathrm{sd}^{k-1}(\mathrm{sd}(P))$.*



**Question 2.9**

*Does $\mathrm{TC}(\mathrm{sd}^k(P))$ converge to $\mathrm{TC}(\mathcal{B}P)$?*



**Question 2.9**

*Does $\mathrm{TC}(\mathrm{sd}^k(P))$ converge to $\mathrm{TC}(\mathcal{B}P)$? This is* **not** *true in general!*



**Question 2.9**

*Does $\mathrm{TC}(\mathrm{sd}^k(P))$ converge to $\mathrm{TC}(\mathcal{B}P)$? This is* **not** *true in general!*

**Example 2.10**

*Let $P$ be a finite poset which is weakly contractible but not contractible.*

**Question 2.9**

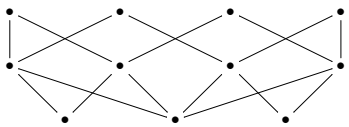
*Does* $\mathrm{TC}(\mathrm{sd}^k(P))$ *converge to* $\mathrm{TC}(\mathcal{B}P)$*? This is* **not** *true in general!*

**Example 2.10**

*Let* $P$ *be a finite poset which is weakly contractible but not contractible. We notice the following two points:*

- $P$ *is weakly contractible* $\Leftrightarrow$ $\mathcal{B}P$ *is contractible.*
- $P$ *is not contractible* $\Leftrightarrow$ $\mathrm{sd}(P)$ *is not contractible.*

$\mathrm{TC}(\mathrm{sd}^k(P)) > 1$ *for any* $k \geq 0$ *but* $\mathrm{TC}(\mathcal{B}P) = 1$.





**Definition 2.11**

$\tau_P : \mathrm{sd}(P) \to P$ *sends a totally ordered subset* $p_0 < \cdots < p_n$ *to the greatest element* $p_n$*. For* $k \geq 0$,

$$\tau_P^k : \mathrm{sd}^k(P) \overset{\tau_{\mathrm{sd}^{k-1}(P)}}{\longrightarrow} \mathrm{sd}^{k-1}(P) \longrightarrow \cdots \longrightarrow \mathrm{sd}(P) \overset{\tau_P}{\longrightarrow} P.$$

*Consider the topological complexity* $\mathrm{TC}(\tau_{P\times P}^k)$ *of the map*

$$\tau_{P\times P}^k : \mathrm{sd}^k(P \times P) \longrightarrow P \times P.$$



**Definition 2.11**

$\tau_P : \mathrm{sd}(P) \to P$ *sends a totally ordered subset* $p_0 < \cdots < p_n$ *to the greatest element* $p_n$*. For* $k \geq 0$,

$$\tau_P^k : \mathrm{sd}^k(P) \overset{\tau_{\mathrm{sd}^{k-1}(P)}}{\longrightarrow} \mathrm{sd}^{k-1}(P) \longrightarrow \cdots \longrightarrow \mathrm{sd}(P) \overset{\tau_P}{\longrightarrow} P.$$

*Consider the topological complexity* $\mathrm{TC}(\tau_{P\times P}^k)$ *of the map*

$$\tau_{P\times P}^k : \mathrm{sd}^k(P \times P) \longrightarrow P \times P.$$

**Theorem 2.12**

*For any finite poset* $P$ *and* $k \geq 0$, $\mathrm{TC}(\mathcal{B}P) \leq \mathrm{TC}(\tau_{P\times P}^k) \leq \mathrm{TC}(\mathrm{sd}^k(P))$. *Moreover for sufficiently large* $k \geq 0$, *we have* $\mathrm{TC}(\tau_{P\times P}^k) = \mathrm{TC}(\mathcal{B}P)$:

$$\lim_{k\to\infty} \mathrm{TC}(\tau_{P\times P}^k) = \mathrm{TC}(\mathcal{B}P).$$

*The simplicial version was shown by J. González (2018).*



## Symmetric TC for finite posets and classifying space

**Question 3.1 (Problems with the combinatorial approximation to** $\mathrm{TC}^{\Sigma}$**)**

1. $\mathbb{Z}_2$*-action on paths with finite length*
2. *The configuration space of a finite posets and the classifying space*
3. *Quotients of finite posets by group actions*

## Symmetric TC for finite posets and classifying space

**Question 3.1 (Problems with the combinatorial approximation to $\mathrm{TC}^{\Sigma}$)**

1. $\mathbb{Z}_2$-action on paths with finite length
2. The configuration space of a finite posets and the classifying space
3. Quotients of finite posets by group actions

**Definition 3.2**

$J_{2m}$ admits the canonical $\mathbb{Z}_2$-action.



## Symmetric TC for finite posets and classifying space

**Question 3.1 (Problems with the combinatorial approximation to $\mathrm{TC}^{\Sigma}$)**

1. $\mathbb{Z}_2$-action on paths with finite length
2. The configuration space of a finite posets and the classifying space
3. Quotients of finite posets by group actions

**Definition 3.2**

$J_{2m}$ admits the canonical $\mathbb{Z}_2$-action. Define an injective $Z_2$-map preserving both ends

$$P^{J_{2m}} \hookrightarrow P^{J_{2m+2}}$$

by



## Symmetric TC for finite posets and classifying space

**Question 3.1 (Problems with the combinatorial approximation to $\mathrm{TC}^{\Sigma}$)**

1. $\mathbb{Z}_2$-action on paths with finite length
2. The configuration space of a finite posets and the classifying space
3. Quotients of finite posets by group actions

**Definition 3.2**

$J_{2m}$ admits the canonical $\mathbb{Z}_2$-action. Define an injective $Z_2$-map preserving both ends

$$P^{J_{2m}} \hookrightarrow P^{J_{2m+2}}$$

by

$$p_0 \leq p_1 \geq \cdots \geq p_m \leq \cdots \geq p_{2m} \mapsto$$
$$p_0 \leq p_1 \geq \cdots \geq p_m \leq p_m \geq p_m \leq \cdots \geq p_{2m}.$$



## Relation between $F_2(\mathcal{B}P)$ and $\mathcal{B}F_2(P)$

## Relation between $F_2(\mathcal{B}P)$ and $\mathcal{B}F_2(P)$

Recall that

$$F_2(\mathcal{B}P) = \mathcal{B}P \times \mathcal{B}P - \Delta_{\mathcal{B}P} \subset \mathcal{B}P \times \mathcal{B}P,$$

and

$$\mathcal{B}(F_2(P)) = \mathcal{B}(P \times P - \Delta_P) \subset \mathcal{B}(P \times P) \cong \mathcal{B}P \times \mathcal{B}P.$$

We have the inclusion $\mathcal{B}(F_2(P)) \hookrightarrow F_2(\mathcal{B}P)$.



## Relation between $F_2(\mathcal{B}P)$ and $\mathcal{B}F_2(P)$

Recall that

$$F_2(\mathcal{B}P) = \mathcal{B}P \times \mathcal{B}P - \Delta_{\mathcal{B}P} \subset \mathcal{B}P \times \mathcal{B}P,$$

and

$$\mathcal{B}(F_2(P)) = \mathcal{B}(P \times P - \Delta_P) \subset \mathcal{B}(P \times P) \cong \mathcal{B}P \times \mathcal{B}P.$$

We have the inclusion $\mathcal{B}(F_2(P)) \hookrightarrow F_2(\mathcal{B}P)$.

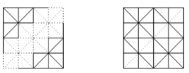


FIGURE 1. $\mathcal{B}(F_2(P))$ and $F_2(\mathcal{B}(P))$ for $P = 0 < 1 > 2 < 3 > 4$

**Proposition 3.3**

*$\mathcal{B}(F_2(P))$ is $\mathbb{Z}_2$-deformation retract of $F_2(\mathcal{B}P)$.*



## Relation between $F_2(\mathcal{B}P)$ and $\mathcal{B}F_2(P)$

Recall that

$$F_2(\mathcal{B}P) = \mathcal{B}P \times \mathcal{B}P - \Delta_{\mathcal{B}P} \subset \mathcal{B}P \times \mathcal{B}P,$$

and

$$\mathcal{B}(F_2(P)) = \mathcal{B}(P \times P - \Delta_P) \subset \mathcal{B}(P \times P) \cong \mathcal{B}P \times \mathcal{B}P.$$

We have the inclusion $\mathcal{B}(F_2(P)) \hookrightarrow F_2(\mathcal{B}P)$.

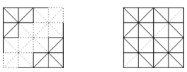


FIGURE 1. $\mathcal{B}(F_2(P))$ and $F_2(\mathcal{B}(P))$ for $P = 0 < 1 > 2 < 3 > 4$

**Proposition 3.3**

*$\mathcal{B}(F_2(P))$ is $\mathbb{Z}_2$-deformation retract of $F_2(\mathcal{B}P)$.*

**Corollary 3.4**

*$C_2(\mathcal{B}P) = F_2(\mathcal{B}P)/\mathbb{Z}_2$ is homotopy equivalent to $\mathcal{B}(F_2(P))/\mathbb{Z}_2$.*



## Quotients of finite posets by $G$-actions

Let $G$ be a (finite) group acting on a finite space $P$. This can be regarded as a functor : $G \to \mathrm{Finite\ spaces}$. We have two options of quotient by the group action:

- $P/G$ : topological quotient $= \mathrm{colim}(G \to \mathrm{Spaces})$.
- $P_G$ : categorical quotient $= \mathrm{colim}(G \to \mathrm{Categories})$

## Quotients of finite posets by $G$-actions

Let $G$ be a (finite) group acting on a finite space $P$. This can be regarded as a functor : $G \to \mathrm{Finite\ spaces}$. We have two options of quotient by the group action:

- $P/G$ : topological quotient $= \mathrm{colim}(G \to \mathrm{Spaces})$.
- $P_G$ : categorical quotient $= \mathrm{colim}(G \to \mathrm{Categories})$

For example, $\mathbb{Z}_2$ naturally acts on the minimal finite model $P$ of a circle in Example 2.7.

The topological quotient $P/\mathbb{Z}_2$ becomes the finite interval in the above left-hand, but the categorical quotient $P_{\mathbb{Z}_2}$ becomes an acyclic category in the above right-hand.



**Remark 3.5 (Barmak and Minian, 2005)**

*$P_G$ is not a finite poset, but an acyclic category. If $G$ acts freely on $P$, we have $\mathcal{B}(P_G) \cong \mathcal{B}P/G$.*

For the configuration space, we use the categorical quotient $A_2(P) = F_2(P)_{\mathbb{Z}_2}$ compatible with the classifying space, instead of the topological quotient $C_2(P)$.



**Remark 3.5 (Barmak and Minian, 2005)**

*$P_G$ is not a finite poset, but an acyclic category. If $G$ acts freely on $P$, we have $\mathcal{B}(P_G) \cong \mathcal{B}P/G$.*

For the configuration space, we use the categorical quotient $A_2(P) = F_2(P)_{\mathbb{Z}_2}$ compatible with the classifying space, instead of the topological quotient $C_2(P)$.

**Theorem 3.6**

*$\mathcal{B}(A_2(P))$ is homotopy equivalent to $C_2(\mathcal{B}P)$.*



**Remark 3.5 (Barmak and Minian, 2005)**

*$P_G$ is not a finite poset, but an acyclic category. If $G$ acts freely on $P$, we have $\mathcal{B}(P_G) \cong \mathcal{B}P/G$.*

For the configuration space, we use the categorical quotient $A_2(P) = F_2(P)_{\mathbb{Z}_2}$ compatible with the classifying space, instead of the topological quotient $C_2(P)$.

**Theorem 3.6**

*$\mathcal{B}(A_2(P))$ is homotopy equivalent to $C_2(\mathcal{B}P)$.*

**Definition 3.7**

*Let denote $A_2^k(P) = \mathrm{sd}^k(F_2(P))_{\mathbb{Z}_2}$ for $k \geq 0$. This is a finite poset for $k \geq 1$. The canonical $\mathbb{Z}_2$-map $\tau^k : \mathrm{sd}^k(F_2(P)) \to F_2(P)$ induces a functor*

$$\widetilde{\tau}^k : A_2^k(P) \longrightarrow A_2(P).$$

**Definition 3.8**

*The $\mathbb{Z}_2$-map $\pi_{2m} : P_L^{J_{2m}} \to F_2(P)$ induces a functor*

$$\rho_{2m} : Q_{2m}(P) = (P_L^{J_{2m}})_{\mathbb{Z}_2} \longrightarrow A_2(P).$$



**Definition 3.8**

*The $\mathbb{Z}_2$-map $\pi_{2m} : P_L^{J_{2m}} \to F_2(P)$ induces a functor*

$$\rho_{2m} : Q_{2m}(P) = (P_L^{J_{2m}})_{\mathbb{Z}_2} \longrightarrow A_2(P).$$

**Definition 3.9**

*For $m \geq 0$ anf $k \geq 1$, let $\mathrm{TC}^{\Sigma}_{2m,k}(P)$ denote the smallest number of open sets $U_1, \cdots, U_n$ covering $A_2^k(P)$ where each $U_i$ admits a functor $s_i : U_i \to Q_{2m}(P)$ with $\rho_{2m} \circ s_i = \widetilde{\tau}^k$ on $U_i$.*



**Definition 3.8**

*The $\mathbb{Z}_2$-map $\pi_{2m} : P_L^{J_{2m}} \to F_2(P)$ induces a functor*

$$\rho_{2m} : Q_{2m}(P) = (P_L^{J_{2m}})_{\mathbb{Z}_2} \longrightarrow A_2(P).$$

**Definition 3.9**

*For $m \geq 0$ anf $k \geq 1$, let $\mathrm{TC}^{\Sigma}_{2m,k}(P)$ denote the smallest number of open sets $U_1, \cdots, U_n$ covering $A_2^k(P)$ where each $U_i$ admits a functor $s_i : U_i \to Q_{2m}(P)$ with $\rho_{2m} \circ s_i = \widetilde{\tau}^k$ on $U_i$.*

We have monotone sequence for any $k$:

$$\mathrm{TC}^{\Sigma}_{0,k}(P) \geq \mathrm{TC}^{\Sigma}_{2,k}(P) \geq \mathrm{TC}^{\Sigma}_{4,k}(P) \geq \cdots \geq 0$$

We denote the stable value by $\mathrm{TC}^{\Sigma}(\widetilde{\tau}^k)$.



Moreover, we have

$$\mathrm{TC}^{\Sigma}(\widetilde{\tau}^1) \geq \mathrm{TC}^{\Sigma}(\widetilde{\tau}^2) \geq \mathrm{TC}^{\Sigma}(\widetilde{\tau}^3) \geq \cdots \geq 0$$

Moreover, we have

$$\mathrm{TC}^{\Sigma}(\tilde{\tau}^1) \geq \mathrm{TC}^{\Sigma}(\tilde{\tau}^2) \geq \mathrm{TC}^{\Sigma}(\tilde{\tau}^3) \geq \cdots \geq 0$$

**Theorem 3.10**

*Let $P$ be a finite poset. $\mathrm{TC}^{\Sigma}(\tilde{\tau}^k) = \mathrm{TC}^{\Sigma}(\mathcal{B}P)$ for sufficiently large $k$:*

$$\lim_{k\to\infty} \mathrm{TC}^{\Sigma}(\tilde{\tau}^k) = \mathrm{TC}^{\Sigma}(\mathcal{B}P).$$



**Thank you !**

# SEPARATION OF BOUNDED ARITHMETIC USING A CONSISTENCY STATEMENT

YORIYUKI YAMAGATA

Abstract. This paper proves Buss's hierarchy of bounded arithmetics $S^1_2 \subseteq S^2_2 \subseteq \cdots \subseteq S^i_2 \subseteq \cdots$ does not entirely collapse. Further, we can allow any finite set of true quantifier-free formulas for the BASIC axioms of $S^1_2, S^2_2, \ldots$. By Takeuti's argument, this implies P $\neq$ NP. Let $\mathbf{Ax}$ be a certain formulation of BASIC axioms. We prove that $S^1_2 \nvdash \mathrm{Con}(\mathrm{PV}^-_1(D) + \mathbf{Ax})$ for sufficiently large $D$, while $S^{D+2}_2 \vdash \mathrm{Con}(\mathrm{PV}^-_1(D) + \mathbf{Ax})$ for a system $\mathrm{PV}^-_1(D)$, a fragment of the system $\mathrm{PV}^-_1$, induction free first order extension of Cook's PV, formulated using sequents with less than $D$ quantifiers and equality symbols. $S^1_2 \nvdash \mathrm{Con}(\mathrm{PV}^-_1(D) + \mathbf{Ax})$ is proved by straightforward adaption of the proof of PV $\nvdash \mathrm{Con}(\mathrm{PV}^-)$ by Buss and Ignjatović. $S^{D+2}_2 \vdash \mathrm{Con}(\mathrm{PV}^-_1(D) + \mathbf{Ax})$ is proved by $S^{D+2}_2 \vdash \mathrm{Con}(\mathrm{PV}^-_q(D) + \mathbf{Ax})$, where $\mathrm{PV}^-_q$ is an extension of $\mathrm{PV}^-$ by adding Henkin quantifiers but without propositional connectives. The latter statement is proved by an extension of a technique used for Yamagata's proof of $S^2_2 \vdash \mathrm{Con}(\mathrm{PV}^-)$, in which a kind of satisfaction relation Sat is defined. By extending Sat to formulas with less than $D$-quantifiers, $S_2 \vdash \mathrm{Con}(\mathrm{PV}^-_q(D) + \mathbf{Ax})$ is obtained in a straightforward way.

## 1. Introduction

[1] This paper proves Buss's hierarchy of bounded arithmetics [3] $S^1_2 \subseteq S^2_2 \subseteq \cdots \subseteq S^i_2 \subseteq \cdots$ does not entirely collapse. More precisely, we prove that, for a certain $D$, $S^1_2 \subsetneq S_2$ holds. We can allow any finite set of true quantifier-free formulas for the BASIC axioms of $S^1_2, S^2_2, \ldots$. By Takeuti's argument [7], this implies P $\neq$ NP.

Our proof is based on the consistency statement of the system $\mathrm{PV}^-_1(D) + \mathbf{Ax}$ where $\mathbf{Ax}$ is a certain formulation of BASIC axioms. $\mathrm{PV}^-_1(D)$ is a fragment of the system $\mathrm{PV}^-_1$ of which proofs contain only formulas with less than $D$ connectives. $\mathrm{PV}^-_1$ is obtained from Cook's PV [5] by removing induction while extending to contain first-order predicate logic. We prove that $S^1_2 \nvdash \mathrm{Con}(\mathrm{PV}^-_1(D) + \mathbf{Ax})$ for sufficiently large $D$, while $S_2 \vdash \mathrm{Con}(\mathrm{PV}^-_1(D) + \mathbf{Ax})$.

The first statement, $S^1_2 \nvdash \mathrm{Con}(\mathrm{PV}^-_1(D) + \mathbf{Ax})$, is proved by straightforward adaption of the proof of PV $\nvdash \mathrm{Con}(\mathrm{PV}^-)$ by Buss and Ignjatović [2]. What we need to do is just checking that the number of connectives which appears in the formulas of constructed proofs is bounded by a fixed constant $D$.

The second statement $S_2 \vdash \mathrm{Con}(\mathrm{PV}^-_1(D) + \mathbf{Ax})$ is much harder to prove. To perform this task, first we define a system $\mathrm{PV}^-_q$, which is obtained from PV from removing induction and adding Henkin-like quantifiers [6] and Boolean terms, but not propositional connectives, to the system. Then, we prove that $\mathrm{PV}^-_q(D)$, which is obtained from $\mathrm{PV}^-_q$ by restricting the number of quantifiers to less than $D$, is consistent by only means available inside $S_2$. Actually, we prove a slightly stronger result

[1] September 27, 2019

that for a kind of "satisfaction" relation Sat, $S_2$ prove that the system $\mathrm{PV}_q^-(D)$ is sound. For this purpose, we extend the method of Yamagata [8] to $\mathrm{PV}_q^-$. In [8], the consistency of $\mathrm{PV}^-$, which is obtained from PV from removing induction, is proved inside $S_2^2$ by defining a kind of satisfaction relation. We extend this satisfaction relation Sat to $\mathrm{PV}_q^-(D)$ formulas, by carefully controlling the bounds of quantifiers appearing in Sat. Once Sat is defined, the soundness proof inside $S_2$ is straightforward.

Next, we encode $\mathrm{PV}_1^-(D)$ into $\mathrm{PV}_q^-$. $t \neq u$ is defined by $\exists i.\, \mathrm{bit}(t,i) \leftrightarrow \mathrm{bit}(u,i) = \top$ using the bit extraction function bit. $t = u \wedge r = s$ is encoded by & where & is the Boolean and. Using this encoding, we can encode all propositional connectives. Thus $\mathrm{PV}_1^-(D)$ into $\mathrm{PV}_q^-(D+1)$. Using this embedding, we prove that for any finite number of sound axioms $\mathbf{Ax}$, we can prove that $\mathrm{PV}^-\,1(D) + \mathbf{Ax}$ is consistent.

Our formulation of $S_2^1$ and $S_2$ is general enough so that they can include any finite quantifier-free true axioms. Therefore our result implies P $\neq$ NP by Takeuti's remark [7].

This paper is organized as follows. Section 2 defines basic notations and defines our formulation of Buss's bounded arithmetic $S_2^i, i = 1, 2, \ldots$. Section 3 introduces PV and $\mathrm{PV}^-$. Section 4 introduces $\mathrm{PV}_q^-$. Section 5 proves $S_2 \vdash \mathrm{Con}(\mathrm{PV}_1^-(D))$. Section 6 introduces $\mathrm{PV}_1^-$ and its encoding to $\mathrm{PV}_q^-$. Section 7 proves $S_2 \vdash \mathrm{Con}(\mathrm{PV}_1^-(D)) + \mathbf{Ax}$. Section 8 proves $S_2^1 \nvdash \mathrm{Con}(\mathrm{PV}_1^-(D) + \mathbf{Ax})$. Finally, Section 9 proves separation of bounded arithmetics $S_2^1 \subsetneq S_2$ and P $\neq$ NP.

## 2. Preliminaries

The sequence $a_1, a_2, \ldots, a_n$ is often abbreviated as $\vec{a}$. If we treat the sequence $a_1, a_2, \ldots, a_n$ as a single object, we write $[a_1, a_2, \ldots, a_n]$. For each sequence $a$, $(a)_i$ is its $i$-th element. We denote an empty sequence by [ ] in the meta-language. For integer $n$, $|n|$ denotes its length of binary representation. For any set $A$, $\#A$ denote its number of elements.

Many types of objects are considered such as proofs of PV, terms of PV, or the *computation* of these terms, all of which require the assignment of Gödel numbers to them. As all of the objects under consideration can be coded as finite sequences of primitive symbols, it will suffice to encode these sequences of symbols. Variables $x_1, x_2, \ldots$ are encoded by variable names $x$ and natural numbers $1, 2, \ldots$ which can be represented by binary strings. Function symbols for all polynomial-time functions are encoded using trees of the primitive functions and labels that show how the function is derived using Cobham's inductive definition of polynomial-time functions. Thus, the symbols that are used in our systems are finite, which enables us to use the finite set of numbers $\{0, \ldots, N-1\}$ to code these symbols. Therefore, the sequence of symbols is coded as $N$-adic numbers.

For each object $a$ consisting of symbols, $\mathrm{size}(a)$ denotes the number of primitive symbols in $a$, that is, the number of the $N$-adic numbers in its Gödel number. If $a$ is a sequence or tree in an object language, $\mathrm{nodes}(a)$ denotes the number of nodes in $a$.

We use the notation $a \equiv b$ when $a$ and $b$ are syntactically equivalent.

For a given term $t$, the notion of subterm $u$ is defined in the usual way. Further, $u$ may be identical to $t$. If $t \not\equiv u$, we call $u$ a proper subterm.

If a statement is labeled by $S_2^i$, it means that the statement is provable within $S_2^i$.

We conservatively extend Buss's $S_2^i, i = 1, 2, \ldots$ by adding function symbols for all polynomial-time computable functions and their defining axioms. The predicate symbol of $S_2$ is $=$ . We formulate $x \leq y$ by $x \dot{-} y = 0$ using positive subtraction $\dot{-}$. Additionally, $S_2^i$ has axioms $\mathbf{Ax}$. The exact contents of $\mathbf{Ax}$ are not important, but $\mathbf{Ax}$ must satisfy

(1) $\mathbf{Ax}$ is finite,
(2) all formulas in $\mathbf{Ax}$ are true in the standard interpretation,
(3) all formulas in $\mathbf{Ax}$ is quantifier free,
(4) $\mathbf{Ax}$ implies all BASIC$^e$ axioms in [2]
(5) $\mathbf{Ax}$ implies $x + y = 0 \to x = 0$, $x + y = 0 \to y = 0$ and $(x = 0 \wedge y = 0) \to x + y = 0$,
(6) $\mathbf{Ax}$ implies $\mathrm{Eq}(x, y) = 0 \leftrightarrow x = y$ for a function Eq,
(7) $\mathbf{Ax}$ implies formulas corresponding P = NP in [7] if P = NP.

## 3. PV and related systems

In this section, we introduce our version of PV and PV$^-$.

PV is formulated as a theory of binary strings rather than integers. We identify binary strings that begin with 1 as a binary representation of integers and binary strings which consists of all 0, including the empty sequence $\varepsilon$, is 0.

PV provides the symbols for the empty sequence $\varepsilon$ and its binary successors $0, 1$ denoted by $b, b_1, \ldots$, which add 0 or 1 to leftmost positions of strings. If a term is solely constructed by $\varepsilon, 0, 1$, it is referred to as a *numeral*. Although binary successors are functions, the notation we use for them employs a special convention to omit the parentheses after the function symbol. Thus, we write $01x$ instead of $0(1(x))$.

The language of PV contains function symbols for all polynomial-time functions. In particular, it contains the constant $\varepsilon$, binary successors $0, 1$ and $\varepsilon^n$, $\mathrm{proj}_n^i$. The intuitive meaning is that $\varepsilon^n$ is the $n$-ary constant function whose value is $\varepsilon$, and $\mathrm{proj}_n^k$ is the projection function. From here, a function symbol for a polynomial-time function $f$ and $f$ itself are often identified. The terms are denoted by $t, t_1, \ldots, u, r, s, \ldots$

For each function symbol $f$ of a polynomial-time function, let $\mathrm{Base}(f)$ be the set of function symbols that are used in Cobham's recursive definition of $f$. We assume that $\mathrm{Base}(f)$ always contains $\varepsilon$, 0, and 1 regardless of $f$. For a set of function symbols $S$, we define $\mathrm{Base}(S) = \bigcup_{f \in S} \mathrm{Base}(f)$. If $\alpha$ represents any sequence of symbols, $\mathrm{Base}(\alpha)$ is defined by the union of $\mathrm{Base}(f)$ for the function symbols $f$ that appear in $\alpha$. $\mathrm{Base}(f)$ is computable by a polynomial time function. For each function symbol $f$, $\mathrm{ar}(f)$ is the arity of $f$. We encode $f \equiv \varepsilon, 0, 1, \varepsilon^n, \mathrm{proj}_n^i$ by

$$\lceil \varepsilon \rceil = \lceil [\mathrm{Fun}, \varepsilon] \rceil \tag{1}$$

$$\lceil 0 \rceil = \lceil [\mathrm{Fun}, 0] \rceil \tag{2}$$

$$\lceil 1 \rceil = \lceil [\mathrm{Fun}, 1] \rceil \tag{3}$$

$$\lceil \varepsilon^n \rceil = \lceil [\mathrm{Fun}, \varepsilon, \overbrace{\#\cdots\#}^{n}] \rceil \tag{4}$$

$$\lceil \mathrm{proj}_n^i \rceil = \lceil [\mathrm{Fun}, \mathrm{proj}, i, \overbrace{\#\cdots\#}^{n}] \rceil \tag{5}$$

where # is a "filler" symbol. Then, a function defined by composition of $g$ and $h_1, \ldots, h_m$ is encoded as $[\mathrm{Fun}, \mathrm{comp}, g, h_1, \ldots, h_n]$. A function defined by recurrence of $g_\varepsilon, g_0, g_1$ is encoded as $[\mathrm{Fun}, \mathrm{rec}, g_\varepsilon, g_0, g_1]$. Then, for any function symbols $f$ which are defined by Cobham's inductive definition, $\mathrm{ar}(f) \leq \mathrm{size}(f)$ is satisfied.

The only predicate in the vocabulary of PV is the equality $=$. Our PV does not have inequalities $\leq, \geq, \ldots$. The formulas $t_1 = t_2$ of PV are formed by connecting two terms $t_1, t_2$ by the equality $=$. We consider PV to be purely equational; hence, the formulas do not contain propositional connectives and quantifiers.

There are three types of axioms and inferences in PV: defining axioms, equality axioms, and induction. We consider that the proofs in PV are all tree-like, not DAG-like. This restriction to the representation of proofs is essential to our consistency proof.

3.1. **Defining axioms.** For all of Cobham's defining equations of polynomial-time functions, there are corresponding defining axioms in PV. For the constant function $\varepsilon^n$, the defining axiom is

$$\varepsilon^n(x_1, \ldots, x_n) = \varepsilon \tag{6}$$

for a positive integer $n$. For the projection function, the defining axiom is

$$\mathrm{proj}^i_n(x_1, \ldots, x_n) = x_i \tag{7}$$

for a positive integer $n$ and an integer $i, 1 \leq i \leq n$. For the binary successor functions 0 and 1, there is no defining axiom. If the function $f(x_1, \ldots, x_n)$ is defined by the composition of $g$ and $h_1, \ldots, h_m$, the defining axiom is

$$f(x_1, \ldots, x_n) = g(h_1(x_1, \ldots, x_n), \ldots, h_m(x_1, \ldots, x_n)). \tag{8}$$

For the function defined by recursion of binary strings, there are three defining axioms:

$$f(\varepsilon, x_1, \cdots, x_n) = g_\varepsilon(x_1, \cdots, x_n) \tag{9}$$

$$f(0x, x_1, \cdots, x_n) = g_0(x, f(x, \vec{x}), x_1, \cdots, x_n) \tag{10}$$

$$f(1x, x_1, \cdots, x_n) = g_1(x, f(x, \vec{x}), x_1, \cdots, x_n). \tag{11}$$

Using Cobham's recursive definition of polynomial time functions, it is easy to see that all polynomial time functions can be defined using these defining axioms. Even though Cook and Urquhart's PV [5] requires all recursion schema to be bounded by a function with a polynomial growth rate, we do not impose this restriction. Thus, our theory can be extended beyond polynomial-time functions. However, this paper focuses on the theory based on polynomial-time functions.

We present defining axioms of which forms $f(\vec{x}) = t$ but we also introduce defining axioms of which forms $t = f(\vec{x})$.

3.2. **Equality axioms.** The identity axiom is formulated as

$$t = t \tag{12}$$

The remaining equality axioms are formulated as inference rules rather than axioms.

$$\frac{u = t}{t = u} \tag{13}$$

$$\frac{t = u \quad u = r}{t = r} \tag{14}$$

$$\frac{t_1 = u_1 \quad \cdots \quad t_n = u_n}{f(t_1, \ldots, t_n) = f(u_1, \ldots, u_n)} \tag{15}$$

$$\frac{t(x) = u(x)}{t(r) = u(r)} \tag{16}$$

for any term $r$.

3.3. **Induction.**

$$\frac{t_1(\varepsilon) = t_2(\varepsilon) \quad t_1(s_i x) = v_i(t_1(x)) \quad t_2(s_i x) = v_i(t_2(x)) \quad (i = 0, 1)}{t_1(x) = t_2(x)} \tag{17}$$

The system PV contains defining axioms, equality axioms, and induction as axioms and inference rules. By contrast, the system $\mathrm{PV}^-$ contains only defining axioms and equality axioms as axioms and inference rules.

## 4. Equational PV with quantifiers

This section introduces $\mathrm{PV}^-_q$, an extension of $\mathrm{PV}^-$ by Henkin-like quantifiers and two types, bit strings $\mathbf{W}$ and Boolean $\mathbf{B}$.

4.1. **Language.** $\mathrm{PV}^-_q$ equips two types $\mathbf{W}$ and $\mathbf{B}$. Types are denoted by $\mathbf{T}$.

$\mathrm{PV}^-_q$ has three constants $\varepsilon, \bot, \top$. $\varepsilon$ is the empty sequence, and $\bot$ and $\top$ signify false and true of truth values respectively.

Function symbols of $\mathrm{PV}^-_q$ are those of $\mathrm{PV}^-$ which are interpreted as functions from $\mathbf{W}$ arguments to $\mathbf{W}$ and new functions $!, \&$ and bit. $!, \&$ are Boolean functions. $!$ is the Boolean negation, and $\&$ is the Boolean conjunction. We also use notations $\|$ for the disjunction, $\to$ the implication and $\leftrightarrow$ as abbreviations. $p(z, x, y)$ is defined as a function which gives $x$ if $z$ is false and $y$ if $z$ is true. If $x, y$ are Boolean, we can define $p(z, x, y) \equiv (!z \to x)\&(z \to y)$. bit denotes a function from two arguments of type $\mathbf{W}$ to $\mathbf{B}$ such that $\mathrm{bit}(x, a)$ gives the $|a|$-th bit of $x$ in Boolean. Composition and recursive definitions are not applied to functions of which values are Boolean. Functions are called *numerical* if they are function symbols of $\mathrm{PV}^-$. Variables of $\mathrm{PV}^-_q$ have types. Variables of type $\mathbf{T}$ are denoted by $x^{\mathbf{T}}, y^{\mathbf{T}}, x_1^{\mathbf{T}}, \ldots$. We omit the type annotation if it is obvious or does not matter in the context. Only well-typed terms are allowed in $\mathrm{PV}^-_q$. A term $t$ with type $\mathbf{T}$ is denoted by $t^{\mathbf{T}}$.

A dependency matrix is a sequence $\mathbf{Q}$ of a form either $x_i(\vec{y}_i)$ or $\{z_i\}(\vec{w}_i)$. All $x_i$ and $z_i$ must be different. We call $x_i, i = 1, \ldots, n$ a defined variable and $\vec{y}_i$ dependent variables of $x_i$. Further, we call $z_i$ a universally quantified variable and $\vec{w}_i$ dependent variables of $z_i$. Defined variables and universally quantified variables of a dependency matrix $\mathbf{Q}$ are called bound variables of $\mathbf{Q}$. A formula of $\mathrm{PV}^-_q$ is a form $\mathbf{Q} \implies t = u$ for a dependency matrix $\mathbf{Q}$. $t, u$ should have the same type. Formulas of $\mathrm{PV}^-_q(D)$ are formulas of $\mathrm{PV}^-_q$ of which clauses appearing in $\mathbf{Q}$ are at most $D$.

All these syntactic objects of $\mathrm{PV}^-_q$, namely, function symbols, variables, terms, and formulas can be encoded to natural numbers following the general idea of Section 2. Further, the following the idea of Section 3, we can guarantee that each function symbol $f$ satisfies $\mathrm{ar}(f) \leq \mathrm{size}(f)$. Base is defined in a similar way to PV.

4.2. **Axiom.** $\mathrm{PV}_q^-$ has two kinds of axioms: the identity axioms and defining axioms. An identity axiom for type $\mathbf{T}$ is $x^{\mathbf{T}} = x^{\mathbf{T}}$. The defining axioms are defining equations of $\mathrm{PV}^-$, bit, $p$ and all valid Boolean equations.

$$\mathrm{bit}(\varepsilon, x^{\mathbf{W}}) = \bot \qquad \mathrm{bit}(0x^{\mathbf{W}}, \varepsilon) = \bot \qquad \mathrm{bit}(1x^{\mathbf{W}}, \varepsilon) = \top \tag{18}$$

$$\mathrm{bit}(bx, b'y) = \mathrm{bit}(x, y) \tag{19}$$

$$p(\bot, x, y) = x \qquad p(\top, x, y) = y \tag{20}$$

Finite many axioms of Boolean algebra

$$t(x_1, \ldots, x_n) = u(x_1, \ldots, x_n) \tag{21}$$

is axiom, where the variables in the last equations are all Boolean.

4.3. **Inferences.** The inferences of $\mathrm{PV}_q^-$ are trees of judgments. Assume that $\mathbf{Q}, \mathbf{Q}_1, \ldots, \mathbf{Q}_n$ are dependency matrices. First, we have the structural rules.

$$\frac{\mathbf{Q}_2, x(\vec{x}), y([x,]\vec{y}), \mathbf{Q}_2 \implies \phi}{\mathbf{Q}_1, y([\vec{x},]\vec{y}), x(\vec{x}), \mathbf{Q}_2 \implies \phi} \tag{22}$$

$$\frac{\mathbf{Q}_1, \{x\}(\vec{x}), \{y\}([x,]\vec{y}), \mathbf{Q}_2 \implies \phi}{\mathbf{Q}_1, \{y\}([\vec{x},]\vec{y}), \{x\}(\vec{x}), \mathbf{Q}_2 \implies \phi} \tag{23}$$

$$\frac{\mathbf{Q}_1, x(\vec{x}), \{y\}(\vec{y}), \mathbf{Q}_2 \implies \phi}{\mathbf{Q}_1, \{y\}(\vec{y}), x(\vec{x}), \mathbf{Q}_2 \implies \phi} \quad \frac{\mathbf{Q}_1, \{x\}(\vec{x}), y(\vec{y}), \mathbf{Q}_2 \implies \phi}{\mathbf{Q}_1, y(\vec{y}), \{x\}(\vec{x}), \mathbf{Q}_2 \implies \phi} \tag{24}$$

where $y \notin \{\vec{x}\}$ and $x \notin \{\vec{y}\}$. In (22), (23), (24), $[\cdots]$ means $\cdots$ are optional.

For each axiom $\phi$ of $\mathrm{PV}_q^-$, a formula

$$\mathbf{Q} \implies \phi \tag{25}$$

can be derived. The symmetry rule.

$$\frac{\mathbf{Q} \implies u = t}{\mathbf{Q} \implies t = u} \tag{26}$$

The transitivity rule.

$$\frac{\mathbf{Q}_1 \implies t = u \quad \mathbf{Q}_2 \implies u = r}{\mathbf{Q}_1, \mathbf{Q}_2 \implies t = r}\ u \tag{27}$$

where $\mathbf{Q}, \mathbf{Q}_2$ must be a well-formed dependency matrix. The compatibility rule.

$$\frac{\mathbf{Q}_1 \implies t_1 = u_1 \quad \cdots \quad \mathbf{Q}_n \implies t_n = u_n}{\mathbf{Q}_1, \ldots, \mathbf{Q}_n \implies f(t_1, \ldots, t_n) = f(u_1, \ldots, u_n)}\ f(x_1, \ldots, x_n) \tag{28}$$

if $\mathbf{Q}_1, \ldots, \mathbf{Q}_n$ must be a well-formed dependency matrix. The quantification and substitution.

$$\frac{\mathbf{Q}, \{x\}(\vec{x}), \implies \phi(x)}{\mathbf{Q} \implies \phi(t)} \qquad \frac{\mathbf{Q} \implies \phi(t)}{\mathbf{Q}, y(\mathrm{FV}(\phi(t)) \implies \phi(y)} \tag{29}$$

Removing a variable from the matrix

$$\frac{\mathbf{Q}_1, x(\vec{y}), \mathbf{Q}_2 \implies \phi}{\mathbf{Q}_1, \mathbf{Q}_2 \implies \phi} \quad \frac{\mathbf{Q}_1, \{x\}(\vec{y}), \mathbf{Q}_2 \implies \phi}{\mathbf{Q}_1, \mathbf{Q}_2 \implies \phi} \tag{30}$$

if $x$ is not a free variable of $\mathbf{Q}_2$ and $\phi$. Adding a variable to the matrix.

$$\frac{\mathbf{Q} \implies \phi(y)}{\{x\}(\vec{x}), \mathbf{Q} \implies \phi(x)} \quad \frac{\mathbf{Q} \implies \phi(y)}{x(\vec{x}), \mathbf{Q} \implies \phi(x)} \tag{31}$$

if $x$ is not a bound variable in $\mathbf{Q}_1, \mathbf{Q}_2$ and $\vec{x}$ are contained in free variables of $\mathbf{Q} \implies \phi$.

$$\frac{\mathbf{Q}_1 \implies t_1 = u_1 \quad \mathbf{Q}_2 \implies t_2 = u_2}{\mathbf{Q_1}, \mathbf{Q_2} \implies p(b, t_1, t_2) = p(b, u_1, u_2)} \tag{32}$$

The trees which are formed using these rules are called $\mathrm{PV}_q^-$*-proofs*. If each formula is in $\mathrm{PV}_q(D)$, the proof is called a $\mathrm{PV}_q^-(D)$-proof.

## 5. Consistency proof of bounded-depth quantified equational PV

This section proves a kind of soundness of tree-like $\mathrm{PV}_q^-(D)$-proofs by $S_2$. We define the notion of *approximate computations* (Section 5.1). Further, we prove the basic properties of computations (Section 5.2) which are used in the soundness proof. Finally, we prove soundness of tree-like $\mathrm{PV}_q^-(D)$-proofs by $S_2^{D+3}$ (Section 5.3).

### 5.1. Approximate computation.

In this section, we define the notion of *approximate computations* as being representations of the evaluations of the terms of $\mathrm{PV}_q^-$. The idea that the values of computation can be approximated using $*$ symbol comes from Beckmann [1], but we only allow $*$ contained in numerals.

**Definition 1** (Approximate values for $\mathbf{W}$)**.** Let $\mathcal{W}$ be the set of terms which are created by $0, 1$ from constants $\varepsilon$, $*^{\mathbf{W}}$. $*$ stands for the *unknown* value. $\mathcal{W}$ has an order structure. For any $v, w, \in \mathcal{W}$, let $\preceq^{\mathbf{W}}$ be the relation which recursively defined by

(1) $\varepsilon \preceq \varepsilon$,
(2) $v \preceq *$,
(3) $v \preceq w \implies bv \preceq bw$ for $b = 0, 1$,

If $v \preceq w$, $w$ is often written as $v^*$. If $v$ does not contain $*$, we say that $v$ is exact. As a convention, $v^*$ denotes an approximation of $v$ and $v^+$ a refinement of $v$. For $v \in \mathcal{W}$, nodes$(v)$ is defined as the number of nodes when we see $v$ as a tree. We call elements of $\mathcal{W}$ *g-numerals*.

**Lemma 1** ($S_2^1$)**.** $\preceq$ *is order relation.*

*Proof. Transitivity law* : we prove that if $v \preceq w$ and $w \preceq z$ hold, $v \preceq z$ holds by induction on nodes$(v)$ + nodes$(w)$ + nodes$(z)$. If $z \equiv *$ then the conclusion follows. If $v \equiv *$ then $w \equiv *$ and $z \equiv *$ must hold. Therefore, $v \preceq z$. Next, if $z \equiv \varepsilon$, then $v \equiv w \equiv \varepsilon$. Thus $v \preceq z$. If $v \equiv \varepsilon$ then $z$ must be either $*$ or $\varepsilon$. For both cases, $v \preceq z$. If $v \equiv b_1 v'$, $w \equiv b_2 w'$ and $z \equiv b_3 z'$, then $b_1 \equiv b_2 \equiv b_3$, $v' \preceq w'$ and $z \preceq z'$ hold. By induction hypothesis, $v' \preceq z'$ therefore $v \preceq z$. Finally, if $v \equiv v_1, v_2$, $w \equiv w_1, w_2$ and $z \equiv z_1, z_2$, then $v_1 \preceq w_1$, $w_1 \preceq z_1$, $v_2 \preceq w_2$ and $w_2 \preceq z_2$. By induction hypothesis, $v_1 \preceq z_1$ and $v_2 \preceq z_2$. Therefore, $v \preceq z$.

*Anti-symmetry law* : we prove that f $v \preceq w$ and $w \preceq v$, $v \equiv w$ by induction on nodes$(v)$ + nodes$(w)$. If $v \equiv *$ then $w$ must be $*$ therefore $v \equiv w$. Similarly if $v \equiv \varepsilon$ then $w$ must be $\varepsilon$ therefore $v \equiv w$. By a symmetric argument we can assume

that $v$ and $w$ are not $*$ nor $\varepsilon$. Then, either $v \equiv bv'$ and $w \equiv bw'$ or $v \equiv v_1, v_2$ and $w \equiv w_1, w_2$. For the former case $v' \trianglelefteq w'$ and $w' \trianglelefteq v'$ must hold. Therefore $v' \equiv w'$. Thus $v \equiv w$. For the later case $v_1 \trianglelefteq w_1$, $w_1 \trianglelefteq v_1$, $v_2 \trianglelefteq w_2$ and $w_2 \trianglelefteq v_1$ must hold. By induction hypothesis $v_1 \equiv w_1$ and $v_2 \equiv w_2$. Therefore $v \equiv w$. □

**Definition 2.** Let $t$ be a term of $\mathrm{PV}^-$ with free variables $x_1, \ldots, x_n$. An *environment* $\rho$ of $t$ is a map assigning subformulas of $t$ to either g-numeral or a Boolean value depending on its type. $\mathrm{dom}(\rho)$ denotes terms $t$ of which $\rho(t)$ is defined. The environment in which $\rho(x)$ is undefined for any variable term, is called *empty environment* and denoted by $[\,]$.

For environments $\rho_1, \rho_2$, $\rho_1 \trianglelefteq \rho_2$ if $\mathrm{dom}(\rho_1) \supseteq \mathrm{dom}(\rho_2)$ and for any $t \in \mathrm{dom}(\rho_2)$, $\rho_1(t) \trianglelefteq \rho_2(t)$. We say that $\rho_2$ is refined by $\rho_1$. If all values of $\rho(t), t \in \mathrm{dom}(\rho)$ are exact, we say that $\rho$ is exact.

When $\rho_1 \trianglelefteq \rho_2$ holds, $\rho_1$ is denoted by $\rho^+$ while $\rho_2$ is denoted by $\rho^*$.

**Definition 3.** Let $v$ be an element of either $\mathcal{W}$ or $\mathcal{B}$, $t$ be a term of $\mathrm{PV}^-$, and $\rho$ be an environment such that $\mathrm{dom}(\rho) \supseteq \mathrm{sbt}(t)$ where $\mathrm{sbt}(t)$ is the set of strict subterms in $t$. The form $\langle t, \rho\rangle \downarrow v$ is referred to as a *(computation) judgement*, $t$ as the main term, $\rho$ as the environment, and $v$ as the value (of $t$). We identify all computational judgments of which environments $\rho$ have the same values on $\mathrm{FV}(t)$. When we assign a computational judgment to the Gödel number, we use the shortest environment. Because we allow approximate computations, a term $t$ may have several g-values as values under the same environment. If $\rho$ is exact, a judgment or a bracket is called *exact*. If $t \equiv f(x_1, \ldots, x_n)$ for a function $f$, the judgement is called *simple*.

If a computational judgement $\langle t, \rho\rangle \downarrow v$ has a form $\langle f(x_1, \ldots, x_n), \rho\rangle \downarrow w$ or $\langle v, \rho\rangle \downarrow w$ where $v$ is an exact value, then it is called *simple*. An inference of a computational judgment is also called simple if its conclusion and premises are simple.

A computation judgment can be derived using the following rules. Each rule is attached by a symbol such as $*$ called a *label*.

In the following rules, on the contrary to the case of terms $t(t_1, \ldots, t_n)$, $f(t_1, \ldots, t_n)$ for any function symbol $f$ means that $t_1, \ldots, t_n$ really appears in $f(t_1, \ldots, t_n)$.

$$\frac{}{\langle t^{\mathbf{W}}, \rho\rangle \downarrow *}\ * \tag{33}$$

$$\frac{}{\langle x, \rho\rangle \downarrow v^*}\ \mathrm{Env} \tag{34}$$

where $v = \rho(x)$ and $v \trianglelefteq v^*$.

$$\frac{}{\langle v, \rho\rangle \downarrow v^*}\ v \tag{35}$$

where $v$ is a numeral and $v^*$ is an approximation of $v$.

$$\frac{\langle t, \rho\rangle \downarrow v}{\langle bt, \rho^+\rangle \downarrow bv^*}\ b \tag{36}$$

where $b$ is either 0 or 1, $v^*$ is an approximation of $v$, $v$ is a g-numeral and $t$ is not a numeral.

$$\frac{(\langle t_i, \rho\rangle \downarrow v_i)_{i=1,\ldots,m}}{\langle \varepsilon^m(t_1,\ldots,t_m), \rho^+\rangle \downarrow \varepsilon}\ \varepsilon^m \tag{37}$$

where $\varepsilon^m$ is the $m$-ary constant function of which the value is always $\varepsilon$. $(\langle t_i, \rho\rangle \downarrow v_i)_{i=1,\ldots,m}$ is the sequence $\langle t_{i_1}, \rho\rangle \downarrow v_1, \ldots, \langle t_{i_k}, \rho\rangle \downarrow v_m$ of judgements. We use similar notation from here.

$$\frac{(\langle t_j, \rho\rangle \downarrow v_j)_{j=1,\ldots,m}}{\langle \mathrm{proj}^i_m(t_1,\ldots,t_m), \rho^+\rangle \downarrow v^*_i}\ \mathrm{proj}^i_m \tag{38}$$

for $i = 1, \cdots, m$. $v^*_i$ is an approximation of $v_i$.

For $p$, the computation is defined as follows.

$$\frac{\langle t, \rho\rangle \downarrow b \quad \langle u_1, \rho\rangle \downarrow v_1 \quad \langle u_2, \rho\rangle \downarrow v_2}{\langle p(t, u_1, u_2), \rho^+\rangle \downarrow p(b, v^*_1, v^*_2)}\ p \tag{39}$$

For bit, the computation is defined as follows.

$$\frac{\langle t, \rho\rangle \downarrow v \quad \langle u, \rho\rangle \downarrow w}{\langle \mathrm{bit}(t, u), \rho\rangle \downarrow b}\ \mathrm{bit} \tag{40}$$

where $b$ is decided by using bit function on g-numerals.

$$\mathrm{bit}(\varepsilon, w) = \bot \text{ if } w \neq *\tag{41}$$

$$\mathrm{bit}(bv, \varepsilon) = b \qquad\qquad \mathrm{bit}(bv, b'w) = \mathrm{bit}(v, w). \tag{42}$$

If $\mathrm{bit}(v, w)$ function is undefined, we often write $\mathrm{bit}(v, w) = *$. If $\mathrm{bit}(v, w) = *$, we put $b = \rho(\mathrm{bit}(t, u))$.

For Boolean formula $P(b_1, \ldots, b_n, \mathrm{bit}(t_1, u_1), \ldots, \mathrm{bit}(t_m, u_m))$, the computation is defind as follows.

$$\frac{(\langle \mathrm{bit}(t_i, u_i), \rho^*\rangle \downarrow b'_i)_{i\in I}}{\langle P(b_1, \ldots, b_n, \mathrm{bit}(t_1, u_1), \ldots, \mathrm{bit}(t_m, u_m)), \rho\rangle \downarrow b}\ \mathrm{Boole} \tag{43}$$

where $I \subseteq \{1, \ldots, m\}$. To compute $b$, first $b'_i, i = 1, \ldots, m$ are to be determined. If $i \in I$, $b'_i$ in the premise is used. Otherwise, let $b'_i = \rho(\mathrm{bit}(t_i, u_i))$. Then, $b$ is determined by computing $P(b_1, \ldots, b_n, b'_1, \ldots, b'_m)$.

For $p$, the computation is defined as follows.

$$\frac{\langle t_1, \rho\rangle \downarrow \bot \quad \langle u_1, \rho\rangle \downarrow v_1 \quad \langle u_2, \rho\rangle \downarrow v_2}{\langle p(t_1, u_1, u_2), \rho^+\rangle \downarrow v^*_1} \tag{44}$$

$$\frac{\langle t_1, \rho\rangle \downarrow \top \quad \langle u_1, \rho\rangle \downarrow v_1 \quad \langle u_2, \rho\rangle \downarrow v_2}{\langle p(t_1, u_1, u_2), \rho^+\rangle \downarrow v^*_2} \tag{45}$$

If $f$ is defined by composition, we have the following rule.

$$\frac{\langle g(\vec{y}, \xi\rangle \downarrow z \quad \langle h_1(\vec{x}), \nu\rangle \downarrow w_1 \cdots \langle h_m(\vec{x}), \nu\rangle \downarrow w_m \quad (\langle t_i, \rho\rangle \downarrow v_i)_{i=1,\ldots,n}}{\langle f(t_1, \ldots, t_n), \rho^+\rangle \downarrow z^*}\ \mathrm{comp} \tag{46}$$

where $\vec{y} = y_1, \ldots, y_m$, $\vec{x} = x_1, \ldots, x_n$, $\nu(x_i) = v_i, i = 1, \ldots, n$ and $\xi(y_j) = w_j, j = 1, \ldots, m$.

If $f$ is defined by recursion, we have the following rules.

$$\frac{\langle g_\varepsilon(x_1, \ldots, x_n), \xi\rangle \downarrow z \quad \langle t, \rho\rangle \downarrow \varepsilon \quad (\langle t_i, \rho\rangle \downarrow v_i)_{i=1,\ldots,n}}{\langle f(t, t_1, \ldots, t_n), \rho^+\rangle \downarrow z^*}\ \text{rec-}\varepsilon \tag{47}$$

where $\xi(x_i) = v_i$ for $i = 1, \ldots, n$.

(48)
$$\frac{\langle g_b(x_0, y, \vec{x}), \xi\rangle \downarrow z \quad \langle t, \rho\rangle \downarrow bv_0 \quad \langle f(x_0, \vec{x}), \nu\rangle \downarrow w \quad (\langle t_j, \rho\rangle \downarrow v_j)_{j=1,\ldots,n}}{\langle f(t, t_1, \ldots, t_n), \rho^+\rangle \downarrow z^*}\ \text{rec-}b$$

where $b = 0, 1$ and $\vec{x} = x_1, \ldots, x_n$. The environment $\nu$ is defined by $\nu(x_j) = v_j$ for $j = 1, \ldots, n$ and $\nu(x_0) = v_0$ while $\xi$ is defined by $\xi(x_j) = v_j, \xi(x_0) = v_0, \xi(y) = w$.

**Definition 4** (Computation Sequence)**.** A *computation sequence $\sigma$ bounded by $U$* is a sequence $\sigma_1, \ldots \sigma_L, L \leq U$ where each $\sigma_i$ has a sequence

$$[R, \langle t_i, \rho_i\rangle \downarrow v_i, n_{1i}, \ldots, n_{l_i i}] \tag{49}$$

which satisfies $n_{ji} < i, j = 1, \ldots, l_i$ and $\mathrm{size}(t_i), \mathrm{nodes}(v_i) \leq U$. Each inference

$$\frac{(\sigma_{n_{1i}})_2 \quad \cdots \quad (\sigma_{n_{l_i i}})_2}{(\sigma_i)_2}\ (\sigma_i)_1 \tag{50}$$

must be a valid computation rule. Here, $(a)_i$ is a projection of $[a_1, \ldots, a_n]$ to $a_i$. $\mathcal{C}(B)$ denotes the set of all computation sequences bounded by $B$. Those computation judgments that are not used as assumptions of some inference rule are referred to as *conclusions* of $\sigma$. If $\langle t, \rho\rangle \downarrow v$ is the only conclusion of $\sigma$, it is written as $\sigma \vdash \langle t, \rho\rangle \downarrow v$; however, if $\sigma$ has multiple conclusions $\vec{\alpha}$, it is written as $\sigma \vdash \vec{\alpha}$. If $\sigma \vdash \langle t, \rho\rangle \downarrow v, \vec{\alpha}$, $\sigma$ is often considered to be a computation of $\langle t, \rho\rangle \downarrow v$. Although a computation sequence $\sigma$ is a sequence, $\sigma$ is often considered to be a DAG, of which the conclusions form the lowest elements.

If there is a computation sequence $\sigma$ with conclusions $\langle t, \rho\rangle \downarrow v, \vec{\alpha}$ such that $\sigma \in \mathcal{C}(L)$, we write $\vdash_L \langle t, \rho\rangle \downarrow v, \vec{\alpha}$. It is easy to see that the relation $\vdash_{|B|}$ is a $\Sigma^b_1$ relation for a given $\rho$.

5.2. **Basic properties of computations.** In this subsection, the basic properties of computations are proved. After proving technical lemmas (Lemma 2, 3 and 4), we prove the lemmas concerning the forms of values of computations of $\varepsilon, 0t, 1t$ and numerals (Lemma 5, 6). Lemma 6 is crucial for our consistency proof because it shows that the numerals are only computed to the same numerals. Next, we prove Lemma 10, which states that the values $v_1$ and $v_2$ obtained by computations of the same term $t^{\mathbf{W}}$ are always compatible, that is, there is the infimum. This enables us to extract the most "accurate" value $v(t, \rho, \sigma)$ of a term $t$ from a computation $\sigma$ of $t$ under an environment $\rho$ (Definition 6). Substitution lemmas (Lemma 13, 14) establish the relationship between substitution into a term which is evaluated by a computation, and assignment in the environment in which the term is evaluated. These lemmas enable a consistency proof of quantification rules. Next, we prove Lemmas 15, 16 and 17 which are used to prove "soundness" of defining axioms in Section 5. All lemmas listed above are proved in $S_2$.

**Lemma 2** ($S^1_2$)**.** *If a computation $\sigma$ derives a conclusion $\langle t, \rho\rangle \downarrow v$, there is a computation $\sigma_1$ which derives a conclusion $\langle t, \rho^+\rangle \downarrow v^*$ for any $\rho^+ \trianglelefteq \rho$ and $v \trianglelefteq v^*$ such that all judgements and rules of $\sigma_1$ are same to $\sigma$ except $\langle t, \rho^+\rangle \downarrow v^*$.*

**Lemma 3** ($S^1_2$)**.** *If $\langle t, \rho\rangle \downarrow v$ appears as a node in the computation sequence $\sigma \in \mathcal{C}(B), \sigma \vdash \vec{\alpha}$ (as a DAG), there is a computation sequence $\tau$ such that $\tau \vdash \langle t, \rho\rangle \downarrow v, \vec{\alpha}$, $\tau \in \mathcal{C}(B + 1)$.*

*Proof.* If $\langle t, \rho\rangle \downarrow v$ is derived according to the inference $R$, another instance of $R$ is added to $\sigma$, which uses the same assumptions as $R$, in which case $\tau$ is obtained. □

**Lemma 4** ($S^1_2$)**.** *If $\sigma \in \mathcal{C}(U), \sigma \vdash \alpha, \vec{\alpha}$, then there exists $\tau \in \mathcal{C}(U)$ such that $\tau \vdash \vec{\alpha}$.*

**Lemma 5** ($S^1_2$)**.** *If $\langle \varepsilon, \rho\rangle \downarrow v$ is contained in a computation $\sigma$, then either $v \equiv \varepsilon$ or $v \equiv *$. If $\langle bt, \rho\rangle \downarrow v$ where $t$ is not a numeral, is contained in $\sigma$, then either $v \equiv bv_0$ for some g-value $v_0$ or $v \equiv *$. If $v \equiv bv_0$, then $\sigma$ contains $\langle t, \rho^*\rangle \downarrow v'_0$ and $v'_0 \trianglelefteq v_0$.*

*Proof.* By induction on $C(\sigma)$. The only rule which can derive $\langle \varepsilon, \rho\rangle \downarrow v$ is either $v$ or $*$-rule. Thus, $v$ is either $\varepsilon$ or $*$. Similarly, if $\sigma$ derives $\langle bt, \rho\rangle \downarrow v$ , the only rule which can derive this is $*$ or $b$-rule. If $v \equiv bv_0$, $\langle bt, \rho\rangle \downarrow bv_0$ can only be derived by $b$-rule. Thus, the assumption contains $\langle t, \rho^*\rangle \downarrow v_0^+$ where $v_0^+ \trianglelefteq v_0$. □

**Lemma 6** ($S^1_2$)**.** *If $\langle v, \rho\rangle \downarrow w$, in which $v$ are a numeral, is contained in a computation $\sigma$, then $v \trianglelefteq w$.*

*Proof.* Only rules which can derive $\langle v, \rho\rangle \downarrow w$ are $*$ and $v$-rules. □

Next, we prove *compatibility* of the results of computations. However, this only holds for *maximal* environments.

**Definition 5** (maximal environment)**.** For an environment $\rho$ and a term of type $\mathbf{W}$, we define

$$V(t, \rho) = \{v \mid \sigma \vdash \langle t, \rho\rangle \downarrow v, C(\sigma) \leq \mathrm{nodes}(t) \cdot U\} \tag{51}$$

$$\mathrm{vals}(t, \rho) = \{v \in V(t, \rho) \mid v' \in V(t, \rho), v' \trianglelefteq v \implies v' = v\} \tag{52}$$

We say that $\rho$ is maximal if $\rho(t) \in \mathrm{vals}(t, \xi)$ and $\rho(\mathrm{bit}(a, b)) = \mathrm{bit}(\rho(a), \rho(b))$ if $\mathrm{bit}(\rho(a), \rho(b)) \neq *$. The property that $\rho$ is a maximal environment is denoted by $\mathrm{Env}(\rho, t, U)$. $\mathrm{Env}(\rho, t, U)$ is a $\Pi^b_2$-formula.

**Lemma 7** ($S^2_2$)**.** *Assume that $\rho_x$ is a map from a finite set of variables to g-values and $\mathrm{bit}(a, b)$ to $\mathcal{B}$ if $\mathrm{bit}(a, b)$ is a subterm of $t$. Then, we can extend $\rho_x$ to a maximal environment $\rho$ such that $\rho(x) = \rho_x(x)$ if $x \in \mathrm{dom}(\rho_x)$ and $\rho(\mathrm{bit}(a, b)) = \rho_x(\mathrm{bit}(a, b))$ if $\mathrm{bit}(\rho(a), \rho(b)) = *$. Further, if $\rho^+_x(x) \trianglelefteq \rho_x(x)$, then we can extend $\rho^+$ to a maximal environment satisfying $\rho^+(t) \trianglelefteq \rho(t)$.*

*Proof.* $\rho$ forms a partial order by $\trianglelefteq$. By MAXLEN principle, we can argue that there is a maximal $\rho$ respect to $\trianglelefteq$. Because $V(t, \rho) \subseteq V(t, \rho^+)$, there is a maximal $\rho^+$ such that $\rho^+ \trianglelefteq \rho$. □

**Lemma 8** ($S^2_2$)**.** *Let $\rho$ be a maximal environment. Then, for any computation $\sigma$ of $\langle t, \rho\rangle \downarrow v$ such that $C(\sigma) \leq \mathrm{nodes}(t) \cdot U$, $\rho(t) \trianglelefteq v$ holds.*

*Proof.* By induction on $\sigma$. We only show the case in which the last rule is bit.

$$\frac{\langle t, \rho\rangle \downarrow v \quad \langle u, \rho\rangle \downarrow w}{\langle \mathrm{bit}(t, u), \rho\rangle \downarrow b} \tag{53}$$

Because $\rho(t) \trianglelefteq v$ and $\rho(u) \trianglelefteq w$, if $\mathrm{bit}(v, w) \neq *$, $\rho(\mathrm{bit}(t, u)) \trianglelefteq \mathrm{bit}(v, w) = b$. If $\mathrm{bit}(v, w) = *$, by definition $\rho(\mathrm{bit}(t, u)) = b$. □

**Lemma 9** ($S^2_2$)**.** *Let $t$ and $u_1, \ldots, u_n$ be terms. For a maximal environment $\rho$, there is a maximal environment $\rho'$ such that $\rho'(x_i) = \rho(u_i)$ and $\rho(t(u_1, \ldots, u_n)) \trianglelefteq \rho'(t(x_1, \ldots, x_n))$.*

Define $\rho_1 \bigwedge \rho_2$ if for any $t$, either $\rho_1(t) \trianglelefteq \rho_2(t)$ or $\rho_2(t) \trianglelefteq \rho_1(t)$ holds.

**Lemma 10** ($S_2^2$)**.** *Let $\rho_1$ and $\rho_2$ be maximal environments which satisfy $\rho_1 \bigwedge \rho_2$, $t$ be the term. If both of $\langle t, \rho_1\rangle \downarrow v$ and $\langle t, \rho_2\rangle \downarrow w$ are contained in computations $\sigma$ and $\tau$ respectively, then $v \bigwedge w$.*

To prove this theorem, we need the next lemma similar to Lemma 4.4. in [1].

**Lemma 11** ($S_2^1$)**.** *If g-numeral $w, u, v$ have relations $w \trianglelefteq u, w \trianglelefteq v$, either $u \trianglelefteq v$ or $v \trianglelefteq u$ holds.*

*Proof.* By induction on $\mathrm{nodes}(w) + \mathrm{nodes}(u) + \mathrm{nodes}(v)$.

If $w = *$, $u = v = *$ therefore $u \trianglelefteq v$. If $w = \varepsilon$, $u, v$ are either $*$ or $\varepsilon$. Therefore $u \trianglelefteq v$ or $v \trianglelefteq u$. If $w = bw'$ for some $b = 0, 1$ and $u'$, there are two possibilities on $u$.

(1) $u$ is $*$
(2) $u$ is $bu'$

If $u$ is $*$, $v \trianglelefteq u$. If $u$ is $bu'$ , $w' \trianglelefteq u'$. There are also two possibilities for $v$. The only non-trivial case is the case in which $v = bv'$. The other case is symmetric. By induction hypothesis, $u' \trianglelefteq v'$ or $v' \trianglelefteq u'$. Therefore, $u \bigwedge v$. □

**Corollary 1** ($S_2$)**.** *If $u \bigwedge v$ for g-numerals $u, v$, $u \trianglelefteq u'$ and $v \trianglelefteq v'$, then $u' \bigwedge v'$.*

*Proof.* We can assume $u \trianglelefteq v$. Then, $u \trianglelefteq v \trianglelefteq v'$. Thus, $u \trianglelefteq u'$ and $u \trianglelefteq v'$. Using Lemma 11, $u' \bigwedge v'$. □

By Corollary 1 and Lemma 8, the lemma is immediate. This fact enables us the following definition and lemma.

**Definition 6.** Let $t$ be a $\mathrm{PV}_q^-$-term, $\sigma$ a computation and $\rho$ a maximal environment. Assume $\sigma$ contains computational judgements $\langle t, \rho_1\rangle \downarrow v_1, \ldots, \langle t, \rho_n\rangle \downarrow v_n$ where maximal environments $\rho_1, \ldots, \rho_n$ satisfy $\rho \trianglelefteq \rho_i, i = 1, \ldots, n$. By Lemma 10, $v_1, \ldots, v_n$ are all pairwise compatible. Then, $v(t, \rho, \sigma)$ is defined as infimum of $v_1, \ldots, v_n$. If there is no such computational judgement, $v(t, \rho, \sigma) = *$ if $t$ has type **W** and $v(t, \rho, \sigma) = \rho(t)$ if $t$ has type **B**. As a special case, if $v$ is an exact value, we put $v(v, \rho, \sigma) = v$.

**Lemma 12** ($S_2^1$)**.** *Let $\sigma$ be a computation, $t$ be a term and $\rho$ be an environment. Let $v = v(t, \rho, \sigma)$. Then, by adding at most one inference to $\sigma$, we obtain $\tau$ which derives $\langle t, \rho\rangle \downarrow v$.*

**Lemma 13** ($S_2^2$, Substitution Lemma I)**.** *Let $u_1, \ldots, u_n$, $t_1, \ldots, t_m$ and $\vec{a} = a_1, \ldots, a_l$ be terms. Let $\sigma$ be a computation which contains occurrences of judgements*

$$\langle t_1(u_1, \ldots, u_n), \rho_1\rangle \downarrow v_1, \ldots, \langle t_m(u_1, \ldots, u_n), \rho_m\rangle \downarrow v_m. \tag{54}$$

*as conclusions. Assume that $\rho \trianglelefteq \rho_1, \ldots, \rho_n$ such that $\rho$ is maximal. Let $\rho'$ be the maximal environment which extends $\rho$ by $\rho'(x_i) = v(u_i, \rho, \sigma)$ and $\rho'(\mathrm{bit}(a(\vec{x}), b(\vec{x}))) = \rho(\mathrm{bit}(a(\vec{u}), b(\vec{u})))$. Then, there is a computation $\tau$ such that*

$$C(\tau) \in C(\sigma) + \sum_{j=1}^{m} \mathrm{size}(t_j(\varepsilon, \ldots, \varepsilon)) \tag{55}$$

*and $\tau$ has conclusions*

$$\langle t_1(\vec{x}), \rho'\rangle \downarrow v_1, \ldots, \langle t_m(\vec{x}), \rho'\rangle \downarrow v_m. \tag{56}$$

*Further, $\tau$ contains all judgements in $\sigma$ as judgements and all conclusions in $\sigma$ as conclusions. $\tau$ derives these conclusions from the same assumptions using the same rule to $\sigma$.*

Each $\langle t_j(u_1,\ldots,u_n),\rho_j\rangle\downarrow v_j, j=1,\ldots,m$ is an occurrence of a judgement but denoted as if it is a judgement by abusing notations. Similarly, $u_1,\ldots,u_n$ in each $\langle t_j(u_1,\ldots,u_n),\rho_j\rangle\downarrow v_j, j=1,\ldots,m$ are occurrences of terms but denoted as if they are terms.

*Proof.* Let $U$ be a fixed integer larger then $C(\sigma)+\sum_{j=1}^{m}\mathrm{size}(t_j(\varepsilon,\ldots,\varepsilon))$ and bound $U$ in the definition of maximal environments. We prove the lemma through the following claims.

**Claim 1.** *Let $\rho,\rho'$ are the same $\rho,\rho'$ in Lemma 13. Let $\kappa$ be a computation with distinguished occurrences of judgments*

$$A\equiv\langle s_1(u_1,\ldots,u_n),\rho_1\rangle\downarrow z_1,\ldots,\langle s_k(u_1,\ldots,u_n),\rho_k\rangle\downarrow z_k \tag{57}$$

*among conclusions and satisfies*

$$C(\kappa)\leq U-\sum_{d=1}^{k}\mathrm{size}(s_d(\varepsilon,\ldots,\varepsilon)) \tag{58}$$

*Further, $\kappa$ contains all judgements in $\sigma$ as judgements and all conclusions in $\sigma$ as conclusions. We assume that $\kappa$ derives these conclusions from the same assumptions using the same rule to $\sigma$. If $\kappa$ contains a judgement in the form of $\langle u_i,\rho^*\rangle\downarrow v$ where $\rho\trianglelefteq\rho^*$, then $\langle u_i,\rho^*\rangle\downarrow v$ is already contained in $\sigma$. Then, there is a computation $\lambda$ which has all conclusions of $\kappa$ plus $\langle s_1(\vec{x}),\rho'\rangle\downarrow z_1,\ldots,\langle s_k(\vec{x}),\rho'\rangle\downarrow z_k$ as conclusions. $\lambda$ satisfies*

$$C(\lambda)\leq C(\kappa)+\sum_{d=1}^{k}\mathrm{size}(s_d(\varepsilon,\ldots,\varepsilon)) \tag{59}$$

*Further, $\lambda$ contains all judgements in $\sigma$ as judgements and all conclusions in $\sigma$ as conclusions. $\lambda$ derives these conclusions from the same assumptions using the same rule to $\sigma$. If $\lambda$ contains a judgement in the form of $\langle u_i,\rho^*\rangle\downarrow v$ where $\rho\trianglelefteq\rho^*$, then $\langle u_i,\rho^*\rangle\downarrow v$ is already contained in $\sigma$.*

We prove the claim by induction on $\Sigma_{d=1}^{k}\mathrm{size}(s_d)$. The claim is $\Pi_2^b$-formula because first $\kappa$ is universally quantified and $\lambda$ is existentially quantified next. Because $\kappa$ changes through the induction steps, quantification over $\kappa$ is necessary. The quantification on $\kappa$ is polynomially bounded, because of the conditions of (58). The quantification of $\lambda$ is also polynomially bounded because of (59). Therefore, the claim can be formulated using bounded quantifiers. Thus, the proof can be formalized by $\Pi_2^b$-LIND. Hence, our proof can be formalized in $S_2^2$. We can safely assume that the last judgement of $\kappa$ is $\langle s_1(u_1,\ldots,u_n),\rho_1\rangle\downarrow z_1$. We use the case analysis of the rule which derives the last judgment and constructs $\lambda$, checking the construction does not introduce a new judgment which has a form of $\langle u_i,\rho^*\rangle\downarrow v$.

First, we drop $\langle s_1(u_1,\ldots,u_n),\rho_1\rangle\downarrow z_1$ from $A$ if $s_1$ does not contain any $x_1,\ldots,x_n$. From here, we assume that $s_1$ contains one of $x_i$.

The case in which the last rule of $\kappa$ is the $*$-rule, is trivial.

If $s_1(u_1,\ldots,u_n)=u_i$ for some $i$, we transform the rule which derives $\langle u_1,\rho_1\rangle\downarrow z_1$ to

$$\frac{}{\langle x_1,\rho'\rangle\downarrow z_1}\ \mathrm{Env} \tag{60}$$

and remove $\langle u_1,\rho_1\rangle\downarrow z_1$ from $A$. The inference obtained is valid because $\rho(u_1)\trianglelefteq z_1$. Because $u_i$ does not contain $x_1$, a statement which contains $u_i$ is not created in $\lambda$.

If the last rule of $\kappa$ is $v$-rule and $s_1(u_1,\ldots,u_n)$ has type $\mathbf{W}$, $s_1(u_1,\ldots,u_n)\equiv b_1\cdots b_l(u_1)$ and $u_1$ is a numeral. First we remove $\langle s_1(u_1,\ldots,u_n),\rho_1\rangle\downarrow v_1$ from $A$ and use induction hypothesis. We obtain the computation $\tau_1$. We add the rules

$$\begin{array}{c}\dfrac{}{\langle x,\rho'\rangle\downarrow z_1}\ \mathrm{Env}\\ \vdots\\ \langle b_1\cdots b_l(x),\rho'\rangle\downarrow b_1\cdots b_l z_1\end{array} \tag{61}$$

to $\tau_1$. Let this computation $\tau$.

$$C(\tau)\leq C(\sigma)+l+\sum_{d=2}^{k}\mathrm{size}(s_d(\varepsilon,\ldots,\varepsilon)) \tag{62}$$

Therefore, we prove the lemma. Because $u_i$ does not contain $x_1$, a statement which contains $u_i$ is not created in $\lambda$.

The case in which the last rule of $\kappa$ is either $0,1$, $\varepsilon^m$, proj, comp or rec-rule. Then, the computation rule has a form

$$\frac{\vec{\beta}\quad\vec{\gamma}\quad(\langle r_q(u_1,\ldots,u_n),\rho_1^*\rangle\downarrow p_q)_{q=1,\ldots,l}}{\langle f(r_1(u_1,\ldots,u_n),\ldots,r_l(u_1,\ldots,u_n)),\rho_1\rangle\downarrow w_1} \tag{63}$$

where $\vec{\beta}$ are simple, which can be empty and $\vec{\gamma}$ are statements which no longer contain $u_k$. Let $\kappa_1$ be a computation obtained by making $(\langle r_q(u_1,\ldots,u_n),\rho_1^*\rangle\downarrow z_q)_{q=1,\ldots,l}$ conclusions and dropping $\langle s_1(u_1,\ldots,u_n),\rho_1\rangle\downarrow w_1$ from $A$. This increases $C(\kappa_1)$ from $C(\kappa)$ at most $l$. By explicitly counting parentheses and a comma, $\sum_{q=1}^{l}\mathrm{size}(r_q(\varepsilon,\ldots,\varepsilon))+l+2\leq\mathrm{size}(s_1(\varepsilon,\ldots,\varepsilon))$. Because

$$C(\kappa_1)\leq C(\kappa)+l \tag{64}$$

$$\leq U-\sum_{d=1}^{k}\mathrm{size}(s_d(\varepsilon,\ldots,\varepsilon))+l \tag{65}$$

$$\leq U-\sum_{q=1}^{l}\mathrm{size}(r_q(\varepsilon,\ldots,\varepsilon))-l-2-\sum_{d=2}^{k}\mathrm{size}(s_d(\varepsilon,\ldots,\varepsilon))+l \tag{66}$$

$$\leq U-\sum_{q=1}^{l}\mathrm{size}(r_q(\varepsilon,\ldots,\varepsilon))-\sum_{d=2}^{k}\mathrm{size}(s_d(\varepsilon,\ldots,\varepsilon))-2 \tag{67}$$

$\kappa$ satisfies (58). Thus we can apply induction hypothesis to $\kappa_1$. Therefore, we obtain $\lambda_1$ which contains all conclusions of $\kappa$ plus $(\langle r_q,\rho'\rangle\downarrow p_q)_{q=1,\ldots,l},(\langle s_d,\rho'\rangle\downarrow z_d)_{d=1,\ldots,m}$

as conclusions. By adding one rule to $\lambda_1$, we obtain $\lambda$.

$$C(\lambda) \leq C(\lambda_1) + 1 \tag{68}$$

$$\leq C(\kappa_1) + \sum_{q=1}^{l} \mathrm{size}(r_q(\varepsilon, \ldots, \varepsilon)) + \sum_{i=2}^{k} \mathrm{size}(s_d(\varepsilon, \ldots, \varepsilon)) + 1 \tag{69}$$

$$\leq C(\kappa) + l + \sum_{q=1}^{l} \mathrm{size}(r_q(\varepsilon, \ldots, \varepsilon)) + \sum_{d=2}^{k} \mathrm{size}(s_d(\varepsilon, \ldots, \varepsilon)) + 1 \tag{70}$$

$$\leq C(\kappa) + \sum_{d=1}^{k} \mathrm{size}(s_d(\varepsilon, \ldots, \varepsilon)) - 1 \tag{71}$$

The case in which the last rule $\kappa$ is bit.

$$\frac{\begin{array}{cc}\vdots & \vdots \\ \langle r(\vec{u}), \rho\rangle \downarrow v & \langle s(\vec{u}), \rho\rangle \downarrow w\end{array}}{\langle \mathrm{bit}(r(\vec{u}), s(\vec{u})), \rho\rangle \downarrow b} \tag{72}$$

where $b = \mathrm{bit}(v, w)$ if $\mathrm{bit}(v, w) \neq *$ and $b = \rho(\mathrm{bit}(r(\vec{u}), s(\vec{u}))$ otherwise. By induction hypothesis, we have a computation $\delta$ of $\langle r(\vec{x}), \rho'\rangle \downarrow v, \langle s(\vec{x}), \rho'\rangle \downarrow w$. If $\mathrm{bit}(v, w) = b \neq *$,

$$\frac{\begin{array}{cc}\vdots & \vdots \\ \langle r(\vec{x}), \rho\rangle \downarrow v & \langle s(\vec{x}), \rho\rangle \downarrow w\end{array}}{\langle \mathrm{bit}(r(\vec{x}), s(\vec{x})), \rho\rangle \downarrow b} \tag{73}$$

is a valid computation. Assume that $\mathrm{bit}(v, w) = *$. Then, $b' = \rho'(r(\vec{x}), s(\vec{x})) = \rho(r(\vec{u}), s(\vec{u})) = b$.

By the construction, $\lambda$ contains all judgments in $\kappa$. By induction hypothesis, judgements which has a form of $\langle u_i, \rho^*\rangle \downarrow v$ in $\lambda_1$ are contained in $\sigma$. A Judgment introduced in $\lambda$ has a form $\langle s_1(x_1, \ldots, x_n), \rho_1\rangle \downarrow v_1$. Because $s_1$ contains one of $x_1, \ldots, x_n$, $s_1(x_1, \ldots, x_n)$ is not same to $u_1, \ldots, u_n$.

□

**Lemma 14** ($S^2_2$, Substitution Lemma II)**.** *Let $u_1, \ldots, u_n$ and $t_1, \ldots, t_m$ be terms. Assume that $u_1, \ldots, u_n$ does not contain $x_1, \ldots, x_n$ but $t_1, \ldots, t_m$ may. Let $\rho$ be a maximal environment and $\rho'$ is the maximal extension of $\rho$ such that $\rho'(x_i) = w_i$ and $\rho'(\mathrm{bit}(r(\vec{x}), s(\vec{x}))) = \rho(\mathrm{bit}(r(\vec{u}), s(\vec{u})))$. Assume that $\mathrm{size}(t_j(u_1, \ldots, u_n)) \leq C, j = 1, \ldots, m$. Then for any computation $\sigma$ with conclusions $\langle t_j(x_1, \ldots, x_n), \rho'\rangle \downarrow v_j$ for $j = 1, \ldots, m$ such that $v(u_i, \rho, \sigma) \trianglelefteq \rho(x_i)$, there is a computation $\tau$ such that $\tau$ has all conclusions of $\sigma$ plus $\langle t_j(u_1, \ldots, u_n), \rho\rangle \downarrow v_j$ for $j = 1, \ldots, m$ as conclusions and*

$$C(\tau) \leq \max(C(\sigma) + \sum_{j=1}^{m} \mathrm{size}(t_j(\varepsilon, \ldots, \varepsilon)), C). \tag{74}$$

*$\tau$ contains all judgements in $\sigma$. $\tau$ contains all conclusions of $\sigma$ and derives them from the same assumptions using the same rule to $\sigma$.*

*Proof.* Let $U$ be an integer larger than $C(\sigma) + \sum_{j=1}^{m} \mathrm{size}(t_j(\varepsilon, \ldots, \varepsilon))$ and $C$. We prove the lemma using the following claims.

**Claim 2.** *Let $\rho, \rho'$ be the same as in the lemma. Let $\kappa$ be a computation with distinguished conclusions*

$$A \equiv \langle s_1(x_1, \ldots, x_n), \rho' \rangle \downarrow z_1, \ldots, \langle s_k(x_1, \ldots, x_n), \rho' \rangle \downarrow z_k \tag{75}$$

*which satisfies*

$$\mathrm{size}(s_d(u_1, \ldots, u_n)) \leq D, d = 1, \ldots, k.. \tag{76}$$

*Assume that $\kappa$ contains all judgements of $\sigma$ and derives them from the same assumptions using the same rule to $\sigma$. We assume that $\kappa$ satisfies*

$$C(\kappa) \leq \max(U - \sum_{d=1}^{k} s_d(\varepsilon, \ldots, \varepsilon), C) \tag{77}$$

*Then, there are a computation $\lambda$ such that the conclusions of $\lambda$ are $\langle s_d(u_1, \ldots, u_n), \rho \rangle \downarrow z_d$ for $d = 1, \ldots, k$ and*

$$C(\lambda) \leq \max(C(\kappa) + \sum_{d=1}^{k} \mathrm{size}(s_d(\varepsilon, \ldots, \varepsilon)), C). \tag{78}$$

*Further, $\lambda$ contains all judgements in $\kappa$. $\lambda$ contains all conclusions of $\sigma$ and derives them from the same assumptions using the same rule of $\sigma$ and $\lambda$ does not contain new inference of $\langle x_i, \rho \rangle \downarrow v_i$.*

Because $\kappa$ changes through the induction steps, quantification over $\kappa$ is necessary. The quantification on $\kappa$ is polynomially bounded, because of the conditions of (77) and (76). The quantification of $\lambda$ is also polynomially bounded by (76). Therefore, Claim 2 can be formalized by $\Pi^b_2$-formula. Therefore, our proof can be formalized in $S^2_2$. From the claim, our lemma is readily proven. We can safely assume that the last judgement is $\langle s_1(x_1, \ldots, x_n), \rho' \rangle \downarrow z_1$. We use a case analysis on the last rule of $\kappa$.

If the last rule of $\kappa$ is either $*$ or $v$-rule, the proof is trivial.

The case in which the last rule of $\kappa$ is Env-rule for $x_i$. Because $\kappa$ contains all judgements of $\sigma$, $\kappa$ contains $\langle u_i, \rho^* \rangle \downarrow w_i^+$.

The case in which the last rule of $\kappa$ is either $\varepsilon^m$, $0, 1$, proj, comp, rec-rule, $p$ or Boolean rules. Then, the computation rule has a form

$$\frac{\vec{\beta} \quad (\langle r_q, \rho^* \rangle \downarrow p_q)_{q=1,\ldots,l}}{\langle f(r_1, \ldots, r_l), \rho \rangle \downarrow z_1} \tag{79}$$

where $\vec{\beta}$ are simple, which can be empty. Let $\kappa_1$ be the computation obtained by making $(\langle r_q, \rho \rangle \downarrow p_q)_{q=1,\ldots,l}$ conclusions by increasing $C(\kappa)$ at most $l$. We add $(\langle r_q, \rho \rangle \downarrow p_q)_{q=1,\ldots,l}$ to $A$ while remove the occurrence of $\langle s_1(x_1, \ldots, x_n), \rho' \rangle \downarrow z_1$ from $A$. By explicitly counting parentheses and a comma,

$$\sum_{d=1}^{l} \mathrm{size}(r_q(\varepsilon, \ldots, \varepsilon)) + l + 2 \leq \mathrm{size}(s_1(\varepsilon, \ldots, \varepsilon)). \tag{80}$$

Because

$$C(\kappa_1) \leq C(\kappa) + l \tag{81}$$

$$\leq U - \sum_{d=1}^{k} \mathrm{size}(s_d(\varepsilon, \ldots, \varepsilon)) + l \tag{82}$$

$$\leq U - \sum_{q=1}^{l} \mathrm{size}(r_q(\varepsilon, \ldots, \varepsilon)) - l - 2 - \sum_{d=2}^{k} \mathrm{size}(s_d(\varepsilon, \ldots, \varepsilon)) + l \tag{83}$$

$$\leq U - \sum_{q=1}^{l} \mathrm{size}(r_q(\varepsilon, \ldots, \varepsilon)) - \sum_{d=2}^{k} \mathrm{size}(s_d(\varepsilon, \ldots, \varepsilon)) - 2 \tag{84}$$

Thus we can apply induction hypothesis to $\kappa_1$. Therefore, we obtain $\lambda_1$ of which conclusions are

$$(\langle r_q(u_1, \ldots, u_n), \rho\rangle \downarrow p_q)_{q=1,\ldots,l}, (\langle s_d(u_1, \ldots, u_n), \rho\rangle \downarrow z_d)_{d=2,\ldots,k}. \tag{85}$$

By adding one rule to $\lambda_1$, we obtain $\lambda$. $C(\lambda)$ is bounded by

$$C(\lambda_1) + 1 \tag{86}$$

$$\leq C(\kappa_1) + \sum_{q=1}^{l} \mathrm{size}(r_q(\varepsilon, \ldots, \varepsilon)) + \sum_{d=2}^{k} \mathrm{size}(s_d(\varepsilon, \ldots, \varepsilon)) + 1 \tag{87}$$

$$\leq C(\kappa) + l + \sum_{q=1}^{l} \mathrm{size}(r_q(\varepsilon, \ldots, \varepsilon)) + \sum_{d=2}^{m} \mathrm{size}(s_d(\varepsilon, \ldots, \varepsilon)) + 1 \tag{88}$$

$$\leq C(\kappa) + \sum_{d=1}^{m} \mathrm{size}(s_d(\varepsilon, \ldots, \varepsilon)) - 1 \tag{89}$$

The case in which the last rule of $\kappa$ is bit.

$$\frac{\langle r(\vec{x}), \rho'\rangle \downarrow v \quad \langle s(\vec{x}), \rho'\rangle \downarrow w}{\langle \mathrm{bit}(r(\vec{x}), s(\vec{x})), \rho'\rangle \downarrow b} \tag{90}$$

where $b = \mathrm{bit}(v, w)$ if $\mathrm{bit}(v, w) \neq *$ and $b = \rho(\mathrm{bit}(r(\vec{x}), s(\vec{x})))$ if $\mathrm{bit}(v, w) = *$. Let $\kappa_1$ be the computation obtained from $\kappa$ by making $\langle r(\vec{x}), \rho'\rangle \downarrow v$ and $\langle s(\vec{x}), \rho'\rangle \downarrow w$ and removing $\mathrm{bit}(r(\vec{x}), s(\vec{x}))$ from $A$. By induction hypothesis, there are $\lambda_1$, which satisfies the conditions of the claim. We obtain a computation $\delta$ as

$$\frac{\langle r(\vec{u}), \rho\rangle \downarrow v \quad \langle s(\vec{u}), \rho\rangle \downarrow w}{\langle \mathrm{bit}(r(\vec{u}), s(\vec{u})), \rho\rangle \downarrow b'} \tag{91}$$

Because Lemma 8, $\rho(r(\vec{u})) \trianglelefteq v$ and $\rho(s(\vec{u})) \trianglelefteq w$ hold. If $\mathrm{bit}(v, w) \neq *$, then $\rho(\mathrm{bit}(r(\vec{u}), s(\vec{u})) = \mathrm{bit}(v, w) = b$. Otherwise, $b' = \rho(\mathrm{bit}(r(\vec{u}), s(\vec{u}))) = \rho'(\mathrm{bit}(r(\vec{x}), s(\vec{x}))) = b$.

By the construction, $\lambda$ satisfies all conditions in the claim. □

**Lemma 15** ($S^2_2$)**.** *Let*

$$f(\vec{u}(\vec{r})) = t(\vec{r}) \tag{92}$$

*is a substitution instance of the defining axiom $f(u(\vec{y})) = t(\vec{y})$. Let $\rho$ be an maximal environment to a computation $\sigma$ and $\vec{\alpha}$ judgements. If $\sigma$ is a computation of $\langle f(\vec{u}(\vec{r})), \rho\rangle \downarrow v, \vec{\alpha}$, there is a computation $\tau$ of $\langle t(\vec{r}), \rho\rangle \downarrow v, \vec{\alpha}$ such that*

$C(\tau) \leq C(\sigma) + \mathrm{size}(f(\vec{u}(\vec{r})) = t(\vec{r}))$. *Further,* $\tau$ *derives* $\vec{\alpha}$ *from the same assumptions using the same rule to* $\sigma$.

*Proof.* Let $\sigma$ be a computation sequence that derives $\vdash_{|B|} \langle f(\vec{u}(\vec{r})), \rho\rangle \downarrow v, \vec{\alpha}$. The lemma is proven by conducting a case analysis of the inference rule $R$ of $\langle f(\vec{u}(\vec{r})), \rho\rangle \downarrow v$ and the defining axioms of $f$. Because the proof uses Lemmas 13 and 14, the proof requires $\Sigma^b_2$-PIND.

If $R$ is $*$-rule, the proof is evident. If $R$ is not $*$-rule, $R$ is determined by the defining axiom of $f$.

For the case that $f \equiv \varepsilon, 0, 1$, the defining axioms do not exist. Thus, the lemma vacuously holds.

For the case in which $f \equiv \varepsilon^n$, $R$ has the form

$$\frac{(\langle u_i(\vec{r}), \rho^*\rangle \downarrow w_i)_{i=1,\ldots,n}}{\langle \varepsilon^n(u_1(\vec{r}), \ldots, u_n(\vec{r})), \rho\rangle \downarrow \varepsilon.} \tag{93}$$

Then,

$$\frac{}{\langle \varepsilon, \rho\rangle \downarrow \varepsilon.} \tag{94}$$

The case is valid.

For the case in which $f \equiv \mathrm{proj}^n_i$, $R$ has the form

$$\frac{(\langle u_j(\vec{r}), \rho^*\rangle \downarrow w_j)_{j=1,\ldots,n}}{\langle \mathrm{proj}^n_i(u_1(\vec{r}), \ldots, u_n(\vec{r})), \rho\rangle \downarrow w^*_i.} \tag{95}$$

Because $w_i \trianglelefteq w^*_i$, we duplicate $\langle u_i(\vec{r}), \rho^*\rangle \downarrow w_i$ and make it to a conclusion. Then we apply Lemma 2.

For the case in which $f$ is defined by the composition $g(h_1(\overline{x}), \ldots, h_n(\overline{x}))$, the inference $R$ of $\sigma$ that derives $\langle f(u_1(\vec{r}), \ldots, u_n(\vec{r})), \rho\rangle \downarrow v$, has the following form.

(96)
$$\frac{\langle g(\vec{y}), \xi\rangle \downarrow v^+ \quad \langle h_1(\vec{x}), \nu\rangle \downarrow w_1 \quad \cdots \quad \langle h_m(\vec{x}), \nu\rangle \downarrow w_m \quad (\langle u_i(\vec{r}), \rho^*\rangle \downarrow z_i)_{i=1,\ldots,n}}{\langle f(u_1(\vec{r}), \ldots, u_n(\vec{r})), \rho\rangle \downarrow v}$$

Because $\nu(x_i) = z_i, i = 1, \ldots, n$, using Lemma 14 repeatedly,

$$\tau_1 \vdash \langle g(\vec{y}), \xi\rangle \downarrow v^+, \langle h_1(\vec{u}(\vec{r})), \rho\rangle \downarrow w_1, \ldots, \langle h_m(\vec{u}(\vec{r})), \rho\rangle \downarrow w_m, \tag{97}$$

in which

$$C(\tau_1) \leq C(\sigma) + \sum_j^m \mathrm{size}(h_j(\vec{\varepsilon})). \tag{98}$$

Again, using Lemma 14, $\tau \vdash \langle g(\vec{h}(\vec{u}(\vec{r}))), \rho\rangle \downarrow v^+, \vec{\alpha}$, is obtained where

$$C(\tau) \leq C(\tau_1) + \mathrm{size}(g(\vec{\varepsilon})) \tag{99}$$

$$\leq C(\sigma) + \mathrm{size}(g(\vec{\varepsilon})) + \sum_j^m \mathrm{size}(h_j(\vec{\varepsilon})) \tag{100}$$

$$\leq C(\sigma) + \mathrm{size}(g(\vec{h}(\vec{u}(\vec{y})))) + m \tag{101}$$

$$\leq C(\sigma) + \mathrm{size}(f(\vec{u}(\vec{r})) = g(\vec{h}(\vec{u}(\vec{r})))). \tag{102}$$

By Lemma 2, the case is valid.

If $f$ is defined by recursion, the inference that derives $\langle f(u_1(\vec{r}),\ldots,u_n(\vec{r})),\rho\rangle \downarrow v$ has the form.

$$\frac{\langle g_\varepsilon(\vec{x}),\nu\rangle\downarrow v \quad \langle\varepsilon,\rho\rangle\downarrow\varepsilon \quad (\langle u_i(\vec{r}),\rho^*\rangle\downarrow z_i)_{i=2,\ldots,n}}{\langle f(\varepsilon,u_2(\vec{r}),\ldots,u_n(\vec{r})),\rho\rangle\downarrow v} \tag{103}$$

or for each $b=0,1$,

(104)

$$\frac{\langle g_b(y,\vec{x}),\nu[y\mapsto w]\rangle\downarrow v \quad \langle f(x_0,\vec{x}),\nu\rangle\downarrow w \quad \langle bu(\vec{r}),\rho^*\rangle\downarrow bz_0 \quad (\langle u_i(\vec{r}),\rho^*\rangle\downarrow z_i)_{i=2,\ldots,n}}{\langle f(bu(\vec{r}),\vec{u}(\vec{r})),\rho\rangle\downarrow v}$$

where $\nu(x_0)=z_0$ while $\nu(x_k)=z_k, k=1,\ldots,n$. First, consider the case of (103). According to Lemma 14, and Lemma 4, we have $\tau$ that satisfies $\tau\vdash\langle g(u_1(\vec{r}),\ldots,u_n(\vec{r})),\rho\rangle\downarrow v,\vec{\alpha}$ and $C(\tau)\leq C(\sigma)+\mathrm{size}(g(\vec{\varepsilon}))$. Thus, the case is valid. Next, consider the case of (104). According to Lemma 5, $\langle u(\vec{r}),\rho^{**}\rangle\downarrow z_0', z_0'\trianglelefteq z_0$ is contained in $\sigma$. According to Lemma 14, we have $\tau_1$ which has the conclusion $\langle f(u(\vec{r}),\vec{u}(\vec{r})),\rho\rangle\downarrow w$. Because $\tau_1$ contains all the judgements of $\sigma$, $\langle g_b(y,\vec{x}),\nu[y\mapsto w]\rangle\downarrow v$, and $\langle u_i(\vec{r}),\rho^*\rangle\downarrow z_i, i=2,\ldots,n$ appear in $\tau_1$. Using Lemma 14 again, we obtain $\tau\vdash\langle g_b(u(\vec{r}),f(u(\vec{r},\vec{u}(\vec{r})),\vec{u}(\vec{r})),\rho\rangle\downarrow v$.

$$C(\tau)\leq C(\tau_1)+\mathrm{size}(g_b(\vec{\varepsilon})) \tag{105}$$

$$\leq C(\sigma)+\mathrm{size}(f(\vec{\varepsilon}))+\mathrm{size}(g_b(\vec{\varepsilon})) \tag{106}$$

$$\leq C(\sigma)+\mathrm{size}(g_b(u(\vec{y}),f(u(\vec{y}),\vec{u}(\vec{y})),\vec{u}(\vec{y})))+1 \tag{107}$$

The case is valid.

The case in which $f=\mathrm{bit}$ is presented next.

$$\frac{\langle\varepsilon,\rho\rangle\downarrow v_1 \quad \langle t_2,\rho\rangle\downarrow v_2}{\langle\mathrm{bit}(\varepsilon,t_2),\rho\rangle\downarrow b} \tag{108}$$

$v_1$ is either $\varepsilon$ or $*$. If $v_1=\varepsilon$, then $b=\bot$ therefore we prove the conclusion. Otherwise, $b=\rho(\mathrm{bit}(\varepsilon,t_2))$. Therefore, $b=\bot$. Other cases are proved in similar ways.

The case in which $f=p$ is trivial. □

**Lemma 16** ($S^2_2$)**.** *Let*

$$f(\vec{u}(\vec{r}))=t(\vec{r}) \tag{109}$$

*is a substitution instance of the defining axiom $f(\vec{u}(\vec{y}))=t(\vec{y})$ of a numerical function $f$. Let $\rho$ be an environment and $\vec{\alpha}$ judgments. If $\sigma$ is a computation of $\langle t(\vec{r}),\rho\rangle\downarrow v,\vec{\alpha}$, there is a computation $\tau$ of $\langle f(\vec{u}(\vec{r})),\rho\rangle\downarrow v,\vec{\alpha}$ such that $C(\tau)\leq C(\sigma)+\mathrm{size}(f(\vec{u}(\vec{r}))=t(\vec{r}))$. Further, $\tau$ derives $\vec{\alpha}$ from the same assumptions using the same rule to $\sigma$.*

*Proof.* Let $\sigma$ be a computation sequence that derives $\vdash_{|B|}\langle t(\vec{r}),\rho\rangle\downarrow v,\vec{\alpha}$. If $v$ is $*$, the lemma is trivial. Therefore, assume otherwise. The lemma is proven by conducting a case analysis on the defining axiom of $f$. Again, because the lemma uses Lemmas 13 and 14, the proof requires $\Sigma^b_2$-PIND.

In the case in which (109) has the form

$$\varepsilon^n(u_1(\vec{r}),\ldots,u_m(\vec{r}))=\varepsilon, \tag{110}$$

By the computation rule

$$\frac{\langle u_1(\vec{r}),\rho\rangle\downarrow * \quad \ldots \quad \langle u_m(\vec{r}),\rho\rangle\downarrow *}{\langle \varepsilon^n(u_1(\vec{r}),\ldots,u_m(\vec{r})),\rho\rangle\downarrow \varepsilon} \tag{111}$$

$\vdash_{\mathrm{size}(\varepsilon^n(u_1(\vec{y}),\ldots,u_m(\vec{y}))=\varepsilon)} \langle \varepsilon^n(u_1(\vec{r}),\ldots,u_m(\vec{r})),\rho\rangle\downarrow\varepsilon$ holds. By Lemma 5, if $\langle\epsilon,\rho\rangle\downarrow v$ then $v$ is either $*$ or $\varepsilon$. Therefore the case is valid.

In the case for which (109) has the form

$$\mathrm{proj}_i^n(u_1(\vec{r}),\ldots,u_m(\vec{r})) = u_i(\vec{r}), \tag{112}$$

the computation rule

(113)
$$\frac{\langle u_1(\vec{r}),\rho\rangle\downarrow *\cdots\langle u_{i-1}(\vec{r}),\rho\rangle\downarrow * \quad \langle u_i(\vec{r}),\rho\rangle\downarrow v \quad \langle u_{i+1}(\vec{r}),\rho\rangle\downarrow *\cdots\langle u_m,\rho\rangle\downarrow *}{\langle \mathrm{proj}_i^n(u_1(\vec{r}),\ldots,u_m(\vec{r})),\rho\rangle\downarrow v.}$$

applies. By assumption, $\vdash_{|B|}\langle u_i(\vec{r}),\rho\rangle\downarrow v$. Thus, $\vdash_{|B|+m}\langle \mathrm{proj}_i^n(u_1(\vec{r}),\ldots,u_m(\vec{r})),\rho\rangle\downarrow v$. Therefore, the case is valid.

The case for which $f$ is defined by the composition of $g,h_1,\ldots,h_m$ is presented next. Let $\sigma$ be a computation of $\langle g(h_1(\vec{u}(\vec{r})),\ldots,h_m(\vec{u}(\vec{r}))),\rho\rangle\downarrow v$. Then, $\sigma$ has the following form.

$$\frac{\vec{\beta} \quad \langle h_1(\vec{u}(\vec{r})),\rho^*\rangle\downarrow w_1 \quad \ldots \quad \langle h_m(\vec{u}(\vec{r})),\rho^*\rangle\downarrow w_m}{\langle g(h_1(\vec{u}(\vec{r})),\ldots,h_m(\vec{u}(\vec{r}))),\rho\rangle\downarrow v} \tag{114}$$

where $\vec{\beta}$ are simple. Let $v_1 = v(u_1(\vec{r}),\rho,\sigma),\ldots,v_n = v(u_n(\vec{r}),\rho,\sigma)$ and $\nu(x_1) = v_1,\ldots,\nu(x_n) = v_n$. By repeatedly applying Lemma 13, we obtain a computation $\tau_1$ which has the conclusion $\langle g(h_1(\vec{x}),\ldots,h_m(\vec{x})),\nu\rangle\downarrow v$. Because $v$ is not $*$, $\tau_1$ contains the inference

$$\frac{\vec{\gamma} \quad \langle h_1(\vec{x}),\nu^*\rangle\downarrow w_1 \quad \ldots \quad \langle h_m(\vec{x}),\nu^*\rangle\downarrow w_m}{\langle g(h_1(\vec{x}),\ldots,h_m(\vec{x})),\nu\rangle\downarrow v} \tag{115}$$

where $C(\tau_1)\leq C(\sigma)+\sum_{j=1}^m \mathrm{size}((h_j(\vec{\varepsilon}))$. By applying Lemma 13, we obtain $\delta_1$ which contains the judgement $\langle g(y_1,\ldots,y_m),\xi\rangle\downarrow v$ where $\xi(y_1)=w_1^+,\ldots,\xi(y_m)=w_m^+$, and satisfies $C(\delta_1)\leq C(\tau_1)+\mathrm{size}(g(\vec{\varepsilon}))$. We can assume that $\delta_1$ contains $\langle h_1(\vec{x}),\nu^*\rangle\downarrow w_1,\ldots,\langle h_m(\vec{x}),\nu^*\rangle\downarrow w_m$ as conclusions by increasing $C(\delta_1)$ by at most $m$. We can assume that $\delta_1$ contains $\langle u_1(\vec{r}),\rho\rangle\downarrow v_1,\ldots,\langle u_n(\vec{r}),\rho\rangle\downarrow v_n$ by increasing $C(\delta_1)$ at most $n$. Then, we obtain $\delta_1$ which satisfies $C(\delta_1)\leq C(\sigma)+\sum_{j=1}^m \mathrm{size}(h_j(\vec{\varepsilon}))+\mathrm{size}(g(\vec{\varepsilon}))+m+n$. Using these judgements, we can assemble an inference of the judgement $\langle f(x_1,\ldots,x_n),\rho\rangle\downarrow v$.

$$\frac{\langle g(y_1,\ldots,y_m),\xi\rangle\downarrow v \quad (\langle h_j(\vec{x}),\nu\rangle\downarrow w_j)_{j=1,\ldots,m} \quad (\langle u_i(\vec{r}),\rho\rangle\downarrow v_i)_{i=1,\ldots,n}}{\langle f(u_1(\vec{r}),\ldots,u_n(\vec{r})),\rho\rangle\downarrow v} \tag{116}$$

Let $\tau$ be the computation which is created in this way.

$$C(\tau)\leq C(\sigma)+\sum_{j=1}^m \mathrm{size}(h_j(\vec{\varepsilon}))+\mathrm{size}(g(\vec{\varepsilon}))+m+n+1 \tag{117}$$

$$\leq C(\sigma)+\mathrm{size}(g(h_1(\vec{\varepsilon}),\ldots,h_m(\vec{\varepsilon})))+2m+n+1 \tag{118}$$

$$\leq C(\sigma)+\mathrm{size}(f(\vec{u}(\vec{y})=t(\vec{y})) \tag{119}$$

because

$$\text{size}(g(h_1(\vec{\varepsilon}),\ldots,h_m(\vec{\varepsilon}))) \leq \text{size}(t) \tag{120}$$

$$2m \leq \text{size}([\text{Fun}, \text{comp}, g, h_1, \ldots, h_m]) = \text{size}(f) \tag{121}$$

$$n+1 \leq \text{size}((u_1(\vec{y}),\ldots,u_n(\vec{y}))). \tag{122}$$

The case for which $f$ is defined by recursion using $g_\varepsilon, g_0, g_1$ is presented next. For the case of $g_\varepsilon$, the proof is similar to that of the case of the composition. Consider the case in which the defining equation is $f(bu_0(\vec{r}), \vec{u}(\vec{r})) = g_b(u_0(\vec{r}), f(u_0(\vec{r}), \vec{u}(\vec{r})), \vec{u}(\vec{r}))$. Then, there exists a derivation $\sigma$ with the value of $g_b(u_0(\vec{r}), f(u_0(\vec{r}), \vec{u}(\vec{r})), \vec{u}(\vec{r}))$. Let $w_0 = v(u_0(\vec{r}), \rho, \sigma), w_i = v(u_i(\vec{r}), \rho, \sigma), i = 1, \ldots, n$ and $v_0 = v(f(u_0(\vec{r}), \vec{u}(\vec{r})), \rho, \sigma)$. The environment $\xi$ is defined by $\xi(x_0) = w_0, \ldots, \xi(x_n) = w_n$ and $\xi(y) = v_0$. By Lemma 13, a computation $\tau$ with the conclusion $\langle g_b(x_0, y, \vec{x}), \xi\rangle \downarrow v$ is obtained. We can assume that $\tau$ contains $\langle f(u_0(\vec{r}), \vec{u}(\vec{r})), \rho\rangle \downarrow v_0$ as a conclusion by increasing $C(\tau)$ by one. The environment $\nu$ is defined by $\nu(x_0) = w_0, \ldots, \nu(x_n) = w_n$. By Lemma 13, a computation $\mu$ with the conclusion $\langle f(x_0, \vec{x}), \nu\rangle \downarrow v_0$ is obtained. $\mu$ still contains $\langle g_b(x_0, y, \vec{x}), \xi\rangle \downarrow v$. Using these judgements, we can assemble a computation $\delta$ of $\langle f(bu_0(\vec{r}), \vec{u}(\vec{r})), \rho\rangle \downarrow v$

(123)

$$\frac{\langle g_b(x_0, y, \vec{x}), \xi\rangle \downarrow v \quad \dfrac{\langle u_0(\vec{r}), \rho^*\rangle \downarrow w_0}{\langle bu_0(\vec{r}), \rho\rangle \downarrow bw_0} \quad \langle f(x_0, \vec{x}), \nu\rangle \downarrow v_0 \quad (\langle u(\vec{r}), \rho\rangle \downarrow w_i)_{i=1,\ldots,n}}{\langle f(bu_0(\vec{r}), \vec{u}(\vec{r})), \rho\rangle \downarrow v}$$

by adding at most $n+2$ rules to derive assumptions. By summing up,

$$C(\delta) \leq C(\mu) + n + 2 \tag{124}$$

$$\leq C(\tau) + n + 2 + \text{size}(f(\vec{\varepsilon})) + 1 \tag{125}$$

$$\leq C(\sigma) + n + 3 + \text{size}(f(\vec{\varepsilon})) + \text{size}(g_b(\vec{\varepsilon})) \tag{126}$$

$$\leq C(\sigma) + \text{size}(f(bu_0(\vec{y}), \vec{u}(\vec{y})) = g_b(u_0(\vec{y}), f(u_0(\vec{y}), \vec{u}(\vec{y})), \vec{u}(\vec{y}))) \tag{127}$$

because

$$n+3 \leq \text{size}(f(bu_0(\vec{y}), \vec{u}(\vec{y}))) \tag{128}$$

$$\text{size}(f(\vec{\varepsilon})) + \text{size}(g_b(\vec{\varepsilon})) \leq \text{size}(g_b(u_0(\vec{y}), f(u_0(\vec{y}), \vec{u}(\vec{y})), \vec{u}(\vec{y}))). \tag{129}$$

Others cases are trivial. □

**Lemma 17** ($S_2^2$)**.**

$$t(\vec{x}) = u(\vec{x}) \tag{130}$$

*is a valid Boolean axiom $t(\vec{y}) = u(\vec{y})$. Let $\rho$ be an environment and $\vec{\alpha}$ judgments. If $\sigma$ is a computation of $\langle t(\vec{x}), \rho\rangle \downarrow b, \vec{\alpha}$, there is a computation $\tau$ of $\langle u(\vec{x}), \rho\rangle \downarrow b, \vec{\alpha}$ such that $C(\tau) \leq C(\sigma) + \text{size}(t(\vec{y}) = u(\vec{y}))$.*

*Proof.* Argue inside $S_2$. Because $t(\vec{y}) = u(\vec{y})$ can be derived by Boolean axioms, we can construct $\tau$ such that $\text{nodes}(\tau) = \text{nodes}(\sigma)$. □

5.3. **Consistency proof.** This section proves the consistency of $\text{PV}_q^-(D)$ inside $S_2$. To this end, we first prove a kind of soundness $\text{PV}_q^-(D)$ by the notion of computation. We call a sequence $\xi$ of substitutions $[q_1/x_1]\cdots[q_d/x_d], d \leq D$ a *substitution sequence* if $x_1, \ldots, x_d$ are all different. We define $\text{dom}(\xi) = \{x_1, \ldots, x_d\}$.

Let $\xi(t) \equiv t[q_d/x_d]\cdots[q_1/x_1]$ and

$$B(\xi) = \max\{\xi(x_i) \mid i = 1, \ldots, d\}. \tag{131}$$

The concatenation of two substitution sequences are written by :. $\mathrm{Env}(\rho, t, U)$ means that $\rho$ is a maximal environment or $t$ where $U$ is the $U$ in Definition 5.

**Definition 7.** Let $U, B$ be integers such that $U \geq 2B$, $\rho$ a maximal environment of which domain does not contain bound variables, $\vec{\alpha}$ judgments and $\xi$ be a substitution sequence such that $B(\xi) \leq (U-B)^D$ holds. For an equation $t = u$ of $\mathrm{PV}_q^-(D)$, its *bounded satisfaction relation* $\mathrm{Sat}(U, B, \xi, \rho, \vec{\alpha}, t = u)$ holds if: For any computation $\sigma \vdash \langle \xi(t), \rho\rangle \downarrow v, \vec{\alpha}$ such the $C(\sigma) \leq (U-B)^{D+2}$, there is a computation $\tau \vdash \langle \xi(u), \rho\rangle \downarrow v, \vec{\alpha}$ such that $C(\tau) \leq C(\sigma) + B^{D+2}$. The statement which is obtained by flipping $t$ and $u$ is also implied. Note that as long as $D \geq 1$ and $B \geq 1$, $C(\sigma) + B^2 \leq (U-B)^{D+2} + B^{D+2} \leq U^{D+2}$.

**Definition 8.** Let $\mathcal{S} \equiv \mathbf{Q} \implies \phi$. We define $\mathrm{Sat}(U, B, \xi, \rho, \vec{\alpha}, \mathcal{S})$ by recursion. For $\mathcal{S} = \implies \phi$

$$\mathrm{Sat}(U, B, \xi, \rho, \vec{\alpha}, \implies \phi) \iff \mathrm{Sat}(U, B, \xi, \rho, \vec{\alpha}, \phi) \tag{132}$$

$$\begin{aligned}\mathrm{Sat}(U, B, \xi, \rho, \vec{\alpha}, x(z_1, \ldots, z_m), \mathbf{Q} \implies \phi) \iff \\ B \leq \forall B' \leq U/2, \exists q, B([q/x] : \xi) \leq (U-B')^D \text{ s.t. } \mathrm{FV}(q) \subseteq \{z_1, \ldots, z_m\}, \\ \mathrm{Sat}(U, B', [q/x] : \xi, \rho, \vec{\alpha}, \mathbf{Q} \implies \phi)\end{aligned} \tag{133}$$

$$\begin{aligned}\mathrm{Sat}(U, B, \xi, \rho, \vec{\alpha}, \{x\}(z_1, \ldots, z_m), \mathbf{Q} \implies \phi) \iff \\ \forall p, B([q/x] : \xi) \leq (U-B)^D, \text{ s.t. } \mathrm{FV}(q) \subseteq \{z_1, \ldots, z_m\}, \\ \mathrm{Sat}(U, B, [p/x] : \xi, \rho, \vec{\alpha}, \mathbf{Q} \implies \phi)\end{aligned} \tag{134}$$

Therefore, Sat for $\mathrm{PV}_q(D)$ formula is a bounded formula.

**Lemma 18** ($S_2^1$)**.** *If* $B_1 \leq B_2$,

$$\mathrm{Sat}(U, B_1, \xi, \rho, \vec{\alpha}, \mathcal{S}) \implies \mathrm{Sat}(U, B_2, \xi, \rho, \vec{\alpha}, \mathcal{S}) \tag{135}$$

*Proof.* By meta-induction on the depth of $\phi$. □

**Lemma 19** ($S_2^2$)**.** *Let* $p$ *be a term such that* $\xi(p) = p$.

$$\begin{aligned}\big[\forall \rho', \mathrm{Env}(\rho', t(x), U^{D+2}) \wedge \mathrm{Env}(\rho', u(x), U^{D+2})\big] \to \\ \mathrm{Sat}(U, B, \xi, \rho', \vec{\alpha}, t(x) = u(x)) \implies \\ \big[\forall \rho, \mathrm{Env}(\rho', t(p), U^{D+2}) \wedge \mathrm{Env}(\rho', u(p), U^{D+2})\big] \to \\ \mathrm{Sat}(U, B + \mathrm{size}(t(\varepsilon) = u(\varepsilon)), \xi, \rho, \vec{\alpha}, t(p) = u(p))\end{aligned} \tag{136}$$

*Proof.* Let $\sigma$ be a computation of $\langle \xi(t(p)), \rho\rangle \downarrow v$ such that $C(\sigma) \leq (U - B - \mathrm{size}(t(\varepsilon) = u(\varepsilon)))^{D+2}$. By Lemma 13, there are a maximal environment $\rho'$ and a computation $\sigma_1$ of $\langle \xi(t(x)), \rho'\rangle \downarrow v, \langle p, \rho\rangle \downarrow w$ such that $\rho'(x) = w$ and $\rho'(\mathrm{bit}(a(x), b(x))) = \rho(\mathrm{bit}(a(p), b(p)))$. $C(\sigma_1) \leq C(\sigma) + \mathrm{size}(t(\varepsilon))$. Because $\mathrm{Env}(U^{D+2}, \rho')$, we have $\mathrm{Sat}(U, B, \xi, \rho', \vec{\alpha}, t(x) = u(x))$. Therefore, there is a computation $\tau_1$ of $\langle \xi(u(x)), \rho'\rangle \downarrow v, \langle p, \rho\rangle \downarrow w$

such that $C(\tau_1) \leq C(\sigma_1) + B^{D+2}$. Then, by Lemma 14, there is a computation $\tau$ of $\langle \xi(u(p)), \rho\rangle \downarrow v, \langle p, \rho\rangle \downarrow w$ such that

$$C(\tau) \leq C(\sigma) + \mathrm{size}(t(\varepsilon)) + B^{D+2} + \mathrm{size}(u(\varepsilon)) \tag{137}$$

$$\leq C(\sigma) + (B + \mathrm{size}(t(\varepsilon) = u(\varepsilon)))^{D+2}. \tag{138}$$

□

**Proposition 1** ($S_2$)**.** *For any $U, U \geq 2\mathrm{size}(\chi)$, any substitution sequence $\eta \equiv [q_1/x_1]\cdots[q_d/x_d]$ such that $B(\eta) \leq (U - \mathrm{size}(\chi))^d$, any maximal environment $\rho$ and judgements $\vec{\alpha}$ such that $M(\vec{\alpha}) \leq (U - \mathrm{size}(\chi))^{D+2}$, the conclusion $\mathcal{S} = \mathbf{Q} \implies \phi$ of the $\mathrm{PV}_q^-(D)$-proof satisfies $\mathrm{Sat}(U, \mathrm{size}(\chi), \eta, \rho, \vec{\alpha}, \mathcal{S})$ holds.*

*Proof of Proposition 1.* By induction on $\chi$. Induction hypothesis has at most $D+1$-bounded $\exists$, the induction hypothesis can be coded by a $\Pi^b_{D+2}$-formula. We consider the last rule $R$ of $\chi$ one by one. Let $d = \#\{\mathbf{Q}_1\}$ and $d' = \#\{\mathbf{Q}_2\}$. We assume that all environments $\rho$ satisfies $\mathrm{Env}(\rho, t, U^{D+2})$, $t \in \mathrm{dom}(\rho)$.

The case for which $R$ is an axiom:

$$\mathbf{Q} \implies \phi \tag{139}$$

If $\phi$ is a substitution instance of a defining axiom for a numerical function, the proposition is proven using Lemma 15, 16. If $\phi$ is a substitution instance of a valid Boolean equation, The proposition is proved by Lemma 17.

The case for which the rule is a case analysis using $p$, the proof is trivial.

The case for which the axiom is the reflexive axiom is trivial.

The case for which the inference is a symmetric rule is trivial.

The case for which the inference is a transitivity rule is considered next.

$$\frac{\begin{array}{cc}\vdots\ \chi_1 & \vdots\ \chi_2\\ \mathbf{Q}_1 \implies t = u & \mathbf{Q}_2 \implies u = s\end{array}}{\mathbf{Q}_1, \mathbf{Q}_2 \implies t = s}\ u \tag{140}$$

By induction hypothesis on $\chi_1$ and $\chi_2$, we have a substitution sequence $\eta_1$ such that $B(\eta_1) \leq (U - \mathrm{size}(\chi_1))^d$ and for any maximal environments $\rho$, any judgements $\vec{\alpha}$ such that $\mathrm{size}(\vec{\alpha}) \leq (U - \mathrm{size}(\chi_1))^{D+2}$, $\mathrm{Sat}(U, \mathrm{size}(\chi_1), \eta_1, \rho, \vec{\alpha}, t = u)$ holds. Similarly, we have a substitution sequence $\eta_2$ such that $B(\eta_2) \leq (U - \mathrm{size}(\chi_2))^{d'}$ such that for any environments $\rho$, any judgements $\vec{\alpha}$ such that $\mathrm{size}(\vec{\alpha}) \leq (U - \mathrm{size}(\chi_1))^{D+2}$, $\mathrm{Sat}(U, \mathrm{size}(\chi_1), \eta_2, \rho, \vec{\alpha}, u = s)$ holds. Because $\mathbf{Q}_1, \mathbf{Q}_2$ is well-formed, $\eta_1 : \eta_2(y) = \eta_2 : \eta_1(y)$ for any variable $y$. By Lemma 19, we have

1. $\mathrm{Sat}(U, \mathrm{size}(\chi_1) + \mathrm{size}(\eta_2(t)), \eta_1, \rho, \vec{\alpha}, \eta_2(t) = \eta_2(u)))$,
2. $\mathrm{Sat}(U, \mathrm{size}(\chi_2) + \mathrm{size}(\eta_1(u)), \eta_2, \rho, \vec{\alpha}, \eta_1(u) = \eta_1(s)))$.

Let $\sigma$ be a computation of $\langle \eta_1(\eta_2(t)), \rho\rangle \downarrow v$ such that $C(\sigma) \leq (U - \mathrm{size}(\chi))^{D+2}$. By the clause 1. above, we have a computation $\tau$ of $\langle \eta_1(\eta_2(u)), \rho\rangle \downarrow v$ such that $C(\tau) \leq C(\sigma) + (\mathrm{size}(\chi_1) + \mathrm{size}(t))^{D+2}$. Because $\eta_1(\eta_2(u)) = \eta_2(\eta_1(u))$, we have a computation $\kappa$ of $\langle \eta_1(\eta_2(s)), \rho\rangle \downarrow v$ such that

$$C(\kappa) \leq C(\sigma) + (\mathrm{size}(\chi_1) + \mathrm{size}(t = u))^{D+2} + (\mathrm{size}(\chi_2) + \mathrm{size}(u = s))^{D+2} \tag{141}$$

$$\leq C(\sigma) + \mathrm{size}(\chi)^{D+2} \tag{142}$$

Therefore, $\mathrm{Sat}(U, \mathrm{size}(\chi), \eta_1 : \eta_2, \rho, \vec{\alpha}, t = u)$ holds.

The case in which the inference is

$$\frac{\begin{array}{ccc}\vdots\ \chi_1 & & \vdots\ \chi_n\\ \mathbf{Q}_1 \implies u_1 = s_1 & \cdots & \mathbf{Q}_n \implies u_n = s_n\end{array}}{\mathbf{Q}_1, \ldots, \mathbf{Q}_n \implies f(u_1, \ldots, u_n) = f(s_1, \ldots, s_n),}\ f(x_1, \ldots, x_n) \tag{143}$$

is considered. For simplicity, we only consider the case in which $n = 1$. By induction hypothesis, For any substitution sequences $\eta$, $\mathrm{Sat}(U, \mathrm{size}(\chi_1), \eta, \rho, \alpha, u_1 = s_1)$ holds.

Let $\sigma$ be a computation of $\langle f(\eta(u_1)), \rho\rangle \downarrow v$ such that $C(\sigma) \leq (U - \mathrm{size}(\chi))^{D+2}$. By Lemma 13, we have a maximal environment $\rho'$ such that $\rho'(x) = w$ and a computation $\sigma_1$ of $\langle f(x), \rho'\rangle \downarrow v, \langle \eta(u_1), \rho\rangle \downarrow w$ such that $C(\sigma_1) \leq C(\sigma) + \mathrm{size}(f(\varepsilon))$. By induction hypothesis, we have a computation $\langle f(x), \rho'\rangle \downarrow v, \langle \eta(s_1), \rho\rangle \downarrow w$ such that $C(\tau_1) \leq C(\sigma_1) + \mathrm{size}(\chi_1)^{D+2}$. By Lemma 14, we have a computation $\tau$ of $\langle f(\eta(s_1)), \rho\rangle \downarrow v$ such that $C(\tau) \leq C(\tau_1) + \mathrm{size}(f(\varepsilon))$. Therefore,

$$C(\tau) \leq C(\sigma) + \mathrm{size}(f(\varepsilon)) + \mathrm{size}(\chi_1)^{D+2} + \mathrm{size}(f(\varepsilon)) \tag{144}$$

$$\leq C(\sigma) + \mathrm{size}(\chi)^{D+2} \tag{145}$$

Substitution to a free variable:

$$\frac{\vdots\ \chi_1}{\mathbf{Q}, \{x\}(\vec{y}) \implies \phi(x, \vec{x})} \over \mathbf{Q} \implies \phi(t(\vec{y}), \vec{x})} \tag{146}$$

By the assumption, we have $\mathrm{Sat}(U, \mathrm{size}(\chi), \eta, \rho, \mathbf{Q}, \{x\}(\vec{y}) \implies \phi(x, \vec{x}))$. Therefore, any term $t$ such that $\mathrm{size}(t) \leq \mathrm{size}(\chi_1) \leq U - \mathrm{size}(\chi_1)$ and $\mathrm{FV}(t) \subseteq \{\vec{y}\}$, $\mathrm{Sat}(U, \mathrm{size}(\chi), \eta, \rho, \mathbf{Q} \implies \phi(t, \vec{x}))$.

The proof for introduction of a clause in the matrix is considered next.

$$\frac{\vdots\ \chi_1}{\mathbf{Q} \implies \phi(t(\vec{y}), \vec{y})} \over \mathbf{Q}, x(\vec{y}) \implies \phi(x, \vec{y})} \tag{147}$$

Because for $B(\xi) \leq (U - \mathrm{size}(\chi_1))^{\#\mathbf{Q}}$, $B(\xi[t(\vec{y})/x]) \leq (U - \mathrm{size}(\chi_1))\mathrm{size}(\chi_1) \leq (U/2)^{\#\mathbf{Q}+1} \leq (U - B')^D$ for $\mathrm{size}(\chi) \leq B' \leq U/2$, we have $\mathrm{Sat}(U, \mathrm{size}(\chi), \xi[t/x], \rho, \vec{\alpha}, \phi)$.

Removing an unused clause:

$$\frac{\mathbf{Q}_1, x(\vec{y}), \mathbf{Q}_2 \implies \phi}{\mathbf{Q}_1, \mathbf{Q}_2 \implies \phi} \tag{148}$$

Because dependent variables $\mathbf{Q}_2$ and free variables of $\phi$ do not contain $x$, the proof is trivial. The proof for the case for $\{x\}$ is similar.

Exchange rule:

$$\frac{\mathbf{Q}_1, x(\vec{y}), z(x, \vec{w}), \mathbf{Q}_2 \implies \phi}{\mathbf{Q}_1, z(\vec{y}, \vec{w}), x(\vec{y}), \mathbf{Q}_2 \implies \phi} \tag{149}$$

This rule is valid because

$$\eta : [p/x][q/z] : \xi(t) = \eta : [q[p/x]/z][p/x] : \xi(t) \tag{150}$$

and $\mathrm{FV}(q[p/x]) = \mathrm{FV}(q) \cup \mathrm{FV}(p) \backslash \{x\}$.

Universal quantification:

$$\frac{\mathbf{Q} \implies \phi(x)}{\{x\}(\vec{x}), \mathbf{Q} \implies \phi(x)} \tag{151}$$

Let $\phi(x) \equiv t(x) = u(x)$. We will prove that for any term $p, B([p/x] : \xi) \leq (U - \mathrm{size}(\chi))^{d+1}$ such that $\mathrm{FV}(p) \subseteq \{\vec{x}\}$, we have

$$\mathrm{Sat}(U, \mathrm{size}(\chi), [p/x] : \xi, \rho, \vec{\alpha}, t(x) = u(x)). \tag{152}$$

By induction hypothesis,

$$\forall \rho', \mathrm{Env}(U^{D+2}, \rho') \to \mathrm{Sat}(U, B, \xi, \rho', \vec{\alpha}, t(x) = u(x)) \tag{153}$$

holds. Therefore, by Lemma 19, $\mathrm{Sat}(U, \mathrm{size}(\chi_1) + \mathrm{size}(t(\varepsilon) = u(\varepsilon)), \xi, \rho, \vec{\alpha}, t(p) = u(p))$. Therefore, the proposition holds for this case. □

**Theorem 1** ($S_2^{D+2}$)**.** *$S_2$ proves $\mathrm{PV}_q^-(D) \nvdash 0\varepsilon = 1\varepsilon$*

*Proof.* Assume that there is a proof $\pi$ of $0\varepsilon = 1\varepsilon$ in $\mathrm{PV}_q^-(D)$. Let $\sigma$ be a computation of $\langle 0\varepsilon, [\,]\rangle \downarrow 0\varepsilon$. By Proposition 1, there is a computation $\tau$ of $\langle 1\varepsilon, [\,]\rangle \downarrow 0\varepsilon$, which contradicts Lemma 6. □

## 6. First-order PV without induction

The system $\mathrm{PV}_1^-$ is an extension of $\mathrm{PV}^-$ by first-order predicate logic. The language is extended so that it contains propositional connectives $\neg, \wedge, \bot$ and quantifiers. $\phi_1 \vee \phi_2, \phi_1 \to \phi_2$ are abbreviations of $\neg(\neg\phi_1 \wedge \neg\phi_2)$, $\neg(\phi_1 \wedge \neg\phi_2)$ respectively. The inferences of $\mathrm{PV}_1^-$ are formulated by a sequent calculus.

### 6.1. **Axioms.**

$$t = u \implies t = u \tag{154}$$

$$\implies t = t \tag{155}$$

$$t = u \implies u = t \tag{156}$$

$$t = u, u = s \implies t = s \tag{157}$$

$$t_1 = u_1, \ldots, t_m = u_m \implies f(t_1, \ldots, t_m) = f(u_1, \ldots, u_m) \tag{158}$$

$$\implies f(u_1, \ldots, u_m) = t \tag{159}$$

(159) is a substitution instance of the defining axiom for $f$. We only allow the identity law for atomic formulas.

### 6.2. **Structural rules.**

$$\frac{\Gamma \implies \Delta}{\phi, \Gamma \implies \Delta} \qquad \frac{\Gamma \implies \Delta}{\Gamma \implies \Delta, \phi} \tag{160}$$

$$\frac{\phi, \phi, \Gamma \implies \Delta}{\phi, \Gamma \implies \Delta} \qquad \frac{\Gamma \implies \Delta, \phi, \phi}{\Gamma \implies \Delta, \phi} \tag{161}$$

6.3. **Propositional inferences.** We have the following propositional inference rules.

$$\frac{\phi_i, \Gamma \implies \Delta}{\phi_1 \wedge \phi_2, \Gamma \implies \Delta}(i = 1, 2) \qquad \frac{\Gamma \implies \Delta, \phi_1 \quad \Sigma \implies \Pi, \phi_2}{\Gamma, \Sigma \implies \Delta, \Pi, \phi_1 \wedge \phi_2} \tag{162}$$

$$\frac{\Gamma \implies \Delta, \phi}{\neg\phi, \Gamma \implies \Delta} \qquad \frac{\phi, \Gamma \implies \Delta}{\Gamma \implies \Delta, \neg\phi} \tag{163}$$

6.4. **Predicate inferences.**

$$(164)\quad \frac{\phi(t), \Gamma \implies \Delta}{\forall x.\phi(x), \Gamma \implies \Delta} \qquad \frac{\Gamma \implies \Delta, \phi(x)}{\Gamma \implies \Delta, \forall x.\phi(x)} \quad \text{if } x \notin \mathrm{FV}(\Gamma, \Delta)$$

$$(165)\quad \frac{\phi(x), \Gamma \implies \Delta}{\exists x.\phi(x), \Gamma \implies \Delta} \quad \text{if } x \notin \mathrm{FV}(\Gamma, \Delta) \qquad \frac{\Gamma \implies \Delta, \phi(t)}{\Gamma \implies \Delta, \exists x.\phi(x)}$$

6.5. **Cut rule.**

$$(166)\quad \frac{\Gamma \implies \Delta, \phi \quad \phi, \Sigma \implies \Pi}{\Gamma, \Sigma \implies \Delta, \Pi}$$

The system $\mathrm{PV}_1^-(D)$ for an integer $D$ is the subsystem of $\mathrm{PV}_1^-$ in which numbers of quantifiers and equations of all sequents is bounded by $D$.

6.6. **Consistency proof of $\mathbf{PV}_1^-(D)$.** This subsection presents the translation of $\mathrm{PV}_1^-$ to $\mathrm{PV}_q^-$ and prove the consistency of $\mathrm{PV}_1^-$. If the proofs in consideration have only bounded-size sequents, the translation maps these proofs to proofs in which all formula has bounded numbers of quantifiers. The translation is denoted by $[\phi]_q$. Let $\mathbf{Q}$ be a dependency matrix. The *conjugate* $\mathbf{Q}^\perp$ of $\mathbf{Q}$ is the dependency matrix obtained by replacing $x(\vec{x})$ to $\{x\}(\vec{x})$ and $\{x\}(\vec{x})$ to $x(\vec{x})$. For a $\mathrm{PV}_q$ formula $\phi \equiv \mathbf{Q} \implies t = u$, $\phi^\perp \equiv \mathbf{Q}^\perp \implies !t = u$. For $\mathrm{PV}_q$ formulas $\phi_1 \equiv \mathbf{Q}_1 \implies t_1 = \top$ and $\phi_2 \equiv \mathbf{Q}_2 \implies t_2 = \top$, $\phi_1 \wedge \phi_2$ is defined by

$$(167)\quad \phi_1 \wedge \phi_2 \equiv \mathbf{Q}_1, \mathbf{Q}_2 \implies t_1 \& t_2 = \top$$

Using these operators, we define $[\phi]_q$ as

$$(168)\quad [\bot]_1 \equiv \bot = \top$$

$$(169)\quad [t = u]_q \equiv \{i\}(\mathrm{FV}(t), \mathrm{FV}(u)) \implies \mathrm{bit}(t, i) \leftrightarrow \mathrm{bit}(u, i) = \top$$

$$(170)\quad [\neg\phi]_q \equiv [\phi]_q^\perp$$

$$(171)\quad [\phi_1 \wedge \phi_2]_q \equiv [\phi_1]_q \wedge [\phi_2]_q$$

$$(172)\quad [\forall x.\phi]_q \equiv \{x\}(\mathrm{FV}(\phi)), [\phi]_q$$

$$(173)\quad [\exists x.\phi]_q \equiv x(\mathrm{FV}(\phi)), [\phi]_q.$$

If $[\phi_1]_q \equiv \mathbf{Q_1} \implies t_1 = \top$ and $[\phi_2]_q \equiv \mathbf{Q_2} \implies t_2 = \top$, by definition $[\phi_1 \vee \phi_2]_q$ is the formula

$$(174)\quad \mathbf{Q_1}, \mathbf{Q_2} \implies t_1 || t_2 = \top.$$

The righthand side can be $\top$ because $\mathrm{PV}_q$ proves all Boolean laws.

**Lemma 20** ($S_2^1$)**.** *From* $\mathrm{PV}_q^-(D) \vdash [\phi_1 \vee \phi_2]_q$ *we can infer* $\mathrm{PV}_q^-(D) \vdash [\phi_2 \vee \phi_1]_q$ *in* $\mathrm{PV}_q^-(D)$.

*Proof.* Let $[\phi_1]_q \equiv \mathbf{Q_1} \implies t_1 = \top$ and $[\phi_2]_q \equiv \mathbf{Q_2} \implies t_2 = \top$. The lemma holds because of $\mathbf{Q_1}$ and $\mathbf{Q_2}$ commute. □

**Lemma 21** ($S_2^1$)**.** *From* $\mathrm{PV}_q^-(D) \vdash [(\phi_1 \vee \phi_2) \vee \phi_3)]_q$ *we can infer* $\mathrm{PV}_q^-(D) \vdash [(\phi_1 \vee (\phi_2 \vee \phi_3))]_q$ *in* $\mathrm{PV}_q^-(D)$.

*Proof.* Let $[\phi_i] \equiv \mathbf{Q}_i \implies t_i = \top$ for $i = 1, 2, 3$. The lemma holds because $\mathbf{Q}_i, i = 1, 2, 3$ commute. □

Then we use commutativity and associativity of $\vee$ freely.

**Proposition 2** ($S^1_2$)**.** $\mathrm{PV}^-_1(D) \vdash \phi_1, \ldots, \phi_n \implies \psi_1, \ldots, \psi_m$ *implies* $\mathrm{PV}^-_q(D{+}1) \vdash [\neg\phi_1 \vee \cdots \vee \neg\phi_n \vee \psi_1 \vee \cdots \vee \psi_m]_q$.

*Proof.* By induction on $\mathrm{PV}_p$-proofs. When $\Gamma \equiv \phi_1, \ldots, \phi_n$, we write $\phi_1 \vee \cdots \vee \phi_n$ by $\bigvee \Gamma$. $\neg\Gamma$ denotes $\neg\phi_1, \ldots, \neg\phi_n$.

$$t = u \implies t = u \tag{175}$$

Trivial by Boolean reasoning.

For structural rules, we consider the contraction rule.

$$\frac{\Gamma \implies \Delta, \phi, \phi}{\Gamma \implies \Delta, \phi} \tag{176}$$

We need to prove $[\psi \vee \phi]_q$ from $[\psi \vee \phi \vee \phi]_q$. Let

$$[\psi]_q \equiv \mathbf{Q}_1 \implies t = \top \tag{177}$$

$$[\phi]_q \equiv \mathbf{Q}_2 \implies u = \top \tag{178}$$

and $\mathbf{Q}'_2 \implies u' = \top$ be the formula which replace all bound variables to fresh variables to avoid name clash. We have

$$\mathbf{Q}_1, \mathbf{Q}_2, \mathbf{Q}'_2 \implies t||u||u' = \top \tag{179}$$

Let $\mathbf{Q}_2 \equiv \mathbf{Q_3}, y(\vec{x})$ and $q(z) = p(z, y, y')$. By induction on the complexity of $u$, we can prove that

$$\implies u[q(z)/y] = p(z, u, u'). \tag{180}$$

Therefore, by Boolean reasoning,

$$\implies u[q(u)/y] = u||u'. \tag{181}$$

From (179), we have

$$\mathbf{Q}_1, \mathbf{Q}_3, y(\vec{x}), \mathbf{Q}'_3, y'(\vec{x}), z(\vec{x}, y, y') \implies t||u(z) = \top. \tag{182}$$

By exchanging quantifiers, we obtain

$$\mathbf{Q}_1, \mathbf{Q}_3, z(\vec{x}), \mathbf{Q}'_3, y(\vec{x}), , y'(\vec{x}) \implies t||u(z) = \top. \tag{183}$$

By removing unused variables, we obtain

$$\mathbf{Q}_1, \mathbf{Q}_3, z(\vec{x}) \implies t||u(z) = \top. \tag{184}$$

For the case $\mathbf{Q_2} \equiv \mathbf{Q_3}, \{z\}(\vec{x})$, the proof is easy.

The $L\wedge$, $R\neg$, $L\neg$ and quantification rules are trivial from the definition of the translation of sequents into $\mathrm{PV}_q$.

For the Cut rule.

$$\frac{\Gamma_1 \implies \Delta_1, \phi \quad \phi, \Gamma_2 \implies \Delta_2}{\Gamma_1, \Gamma_2 \implies \Delta_1, \Delta_2} \tag{185}$$

By induction hypothesis and renaming bound variables,

$$\mathbf{Q_1}, \mathbf{Q_2} \implies t_1||u = \top \tag{186}$$

$$\mathbf{Q_3}, {\mathbf{Q'_2}}^{\perp} \implies t_2||!u' = \top \tag{187}$$

Using equality $\top = \top||\top$, we have

$$\mathbf{Q_1}, \mathbf{Q_3}, \mathbf{Q_2}, {\mathbf{Q'_2}}^{\perp} \implies u||!u'||t_1||t_2 = \top \tag{188}$$

By a similar trick for contraction, we have

$$\mathbf{Q_1}, \mathbf{Q_3}, \mathbf{Q_2}, {\mathbf{Q'_2}}^{\perp}, z(\vec{x}.\vec{y}, \vec{y'}, y, y') \implies u(z)||!u(z)||t_1||t_2 = \top \tag{189}$$

By Boolean reasoning,

$$\mathbf{Q_1}, \mathbf{Q_3}, \mathbf{Q_2}, {\mathbf{Q'_2}}^{\perp}, z(\vec{x}, \vec{y}, \vec{y'}, y, y') \implies t_1||t_2 = \top \tag{190}$$

Removing unused variables,

$$\mathbf{Q_1}, \mathbf{Q_3} \implies t_1||t_2 = \top \tag{191}$$

which is what we need to prove the lemma. □

**Theorem 2** ($S_2^{D+2}$)**.** *$S_2$ proves consistency of* $\mathrm{PV}_1^-(D)$

*Proof.* By Proposition 2, the sequents which have $D$ quantifiers and equations are translated to $\mathrm{PV}_q^-(D+1)$-formulas. Therefore, if $\mathrm{PV}_1^-(D)$ is inconsistent, there is a proof of $\perp = \top$ in $\mathrm{PV}_q^-(D)$, which is impossible. □

## 7. $\mathrm{PV}_1^-(D)$ with finitely many axioms

In this section, we prove that $S_2$ also proves the consistency of $\mathrm{PV}_1^-(D)$ with $\mathrm{BASIC}^e$ and P = NP if P = NP is correct. More generally, let $\mathbf{Ax}$ be a finite set of propositions which are defined in Section 2. Therefore, we prove that $S_2$ proves the consistency of $\mathrm{PV}_1^-(D) + \mathbf{Ax}$. Let $\mathbf{A}_1$ be the universal closure of the formula obtained by concatenating equations in $\mathbf{Ax}$ by $\wedge$.

**Theorem 3.** *Let $\mathbf{A}_1$ be the formula above. If $\mathbf{A}_1$ can be written in the language of $\mathrm{PV}_1^-(D)$, $S_2^{D+2}$ proves the consistency of $\mathrm{PV}_1^-(D) + \mathbf{A}_1$.*

To prove the theorem, we use the following lemma.

**Lemma 22** ($S_2^1$)**.** *Let $t(x_1, \ldots, x_n)$ be a term of type and $\rho$ be an exact maximal environment. Assume that $t(\rho(x_1), \ldots, \rho(x_n)) = d$ in the standard interpretation. Then, there is a computation $\sigma$ such that $\sigma \vdash \langle t, \rho\rangle \downarrow d$ and $\mathrm{nodes}(\sigma) \leq P_t(\mathrm{nodes}(\rho))$ for a polynomial $P_t$.*

*Proof.* By meta-induction on the definition on $t$ and induction on $\mathrm{nodes}(\rho)$. □

Let $P$ be a polynomial such that $P(|n_1| + \cdots + |n_m|) \leq |t(n_1, \ldots, n_m)|$ for any term $t$ which appears in $\mathbf{A}_1$.

*Proof of Theorem 3.* Argue inside $S_2^{D+2}$. $\mathbf{A}$ is embedded into $\mathrm{PV}_q^-(D)$ as $t_{\mathbf{A}_1} = \top$. If $\mathrm{PV}_1^-(D) + \mathbf{Ax}$ is inconsistent, we have

$$\mathrm{PV}_q^-(D) \vdash i_1, \ldots, i_m \implies t_{\mathbf{A}_1}(i_1, \ldots, i_m) = \perp \tag{192}$$

Let $U$ as $U^D \geq P(\mathrm{size}(\pi)^{D+2})$. Soundness of $\mathrm{PV}_q^-(D)$ and by definition of Sat,

$$\mathrm{Sat}(U, \mathrm{size}(\pi), \rho, t_{\mathbf{A}_1}(u_1, \ldots, u_m) = \perp) \tag{193}$$

for any maximal environment $\rho$. Because $\perp$ has the obvious computation, there is a computation $\sigma$ of $\langle t_{\mathbf{A}_1}(u_1, \ldots, u_m), \rho\rangle \downarrow \perp$.

**Claim 3.** *Assume that $u_1, u_2, \ldots, u_m$ are ordered by increasing order of their sizes. Then, there are maximal environments $\rho$ and $\rho_e$ such that*

- *$\rho_e$ is an exact maximal environment for $x_1, \ldots, x_m$.*
- *$\rho_e(t(x_1, \ldots, x_m)) \trianglelefteq \rho(t(u_1, \ldots, u_m))$ for any subterm $t$ of $t_{\mathbf{A}_1}$*

We construct $\rho$ and $\rho_e$ by induction on the size of $t$. Because $\mathrm{Env}(\rho, t, U)$ is $\Sigma^b_1$-formula, the claim can be proven in $S^1_2$. Two cases, $t \equiv x_k$ and $t \equiv \mathrm{bit}(a(x_1, \ldots, x_k), b(x_1, \ldots, x_k))$ are considered.

The case in which $t \equiv x_k$. Because $u_k$ has type $\mathbf{W}$, $w_i = \rho(u_k)$ can be computed. Let $\rho_e(x_i) = w_i^+$ where $w_i^+$ is a value obtained by replacing $*$ to $\varepsilon$. Then conditions above hold.

The case in which $t \equiv \mathrm{bit}(a(x_1, \ldots, x_k), b(x_1, \ldots, x_k))$. $v_1 = \rho(a(x_1, \ldots, x_k))$ and $w_1 = \rho(a(x_1, \ldots, x_k))$ are exact by Lemma 22. By induction hypothesis, $v_2 = \rho(a(u_1, \ldots, u_k))$ and $w_2 = \rho(a(u_1, \ldots, u_k))$ satisfy $v_1 \trianglelefteq v_2$ and $w_1 \trianglelefteq w_2$. Therefore, $\mathrm{bit}(v_1, w_1) \trianglelefteq \mathrm{bit}(v_2, w_2)$. Because $t$ does not contain $x_{k+1}, \ldots, x_m$, either $\rho_e(t) \trianglelefteq \rho(t)$ or we can let $\rho(t) := \rho_e(t)$. □

By Claim 3, there is a pair of $\rho_e$ and $\rho$ which satisfy the conditions of Claim 3. There is a computation $\sigma$ of $\langle t_{\mathbf{A}_1}(u_1, \ldots, u_m), \rho\rangle \downarrow \bot$. Therefore, by Lemma 8, $\rho(t_{\mathbf{A}_1}(u_1, \ldots, u_m)) = \bot$. By Claim 3, $\rho_e(t_{\mathbf{A}_1}(x_1, \ldots, x_m)) = \bot$. However, by Lemma 22, there is a computation $\tau$ of $\langle t_{\mathbf{A}_1}(x_1, \ldots, x_m), \rho_e\rangle \downarrow \top$. This is against Lemma 8. □

## 8. Unprovability of consistency of bounded propositional PV without induction by $S^1_2$

In this section, we prove $S^1_2$ cannot prove the consistency of $\mathrm{PV}^-_1(D)$ for a sufficiently large $D$. Because $S^1_2$ is a conservative extension of PV on $\Pi^0_1$-sentences, we show that PV cannot prove the consistency of $\mathrm{PV}^-_1(D)$ for a sufficiently large $D$. The proof is adapted from Buss and Ignjatović [2], Section 4. The only novelty is that we check the number of quantifiers and equations in the $\mathrm{PV}^-_1$-sequents needed to perform the proof. As shown in [4], any propositional formula $A$ can be coded by a equation $t_A$ and

$$\mathrm{PV}_1 \vdash A \iff \mathrm{PV} \vdash t_A = 0 \tag{194}$$

Let $\mathbf{Ax}_e \equiv t_{\mathbf{Ax}} = 0$ be the equation which codes $\bigwedge \mathbf{Ax}$.

**Lemma 23.** *$S^1_2$ is a conservative extension of* $\mathrm{PV} + \mathbf{Ax}_e$.

*Proof.* Let $S^1_2$ be Buss' original $S^1_2$. $S^1_2$ is a conservative extension of PV. Therefore, if $S^1_2 \vdash \bigwedge \mathbf{Ax} \to r = s$, $\mathrm{PV} \vdash (1 - t_{\mathbf{Ax}}) \cdot t_{r=s} = 0$ holds. Therefore, $\mathrm{PV} + \mathbf{Ax}_e \vdash r = s$. □

From here, we reason inside $S^1_2$. Let $\mathrm{sq}(x)$ be the unary function defined by $\mathrm{sq}(x) = x^2$. Define the meta-notation $\mathrm{sq}^k$ as $\mathrm{sq}^0(x) = x$ and $\mathrm{sq}^{k+1}(x) = \mathrm{sq}(\mathrm{sq}^k(x))$. Then, $\mathrm{PV}^-_1$ proves

$$\forall x, y, z.(x \le \mathrm{sq}^m(y) \wedge y \le \mathrm{sq}^k(z)) \to x \le \mathrm{sq}^{m+k}(z) \tag{195}$$

using only sequents of which number of quantifiers and equations are bounded by a constant $D$ which does not depend on $m$ or $k$. Further, the proof can have a tree form, and its number of nodes is quadratic in $m + k$.

Note that the proof of (195) in [2] has a sequential, not tree form. Because we want to obtain the unprovability result for tree forms, we use quantifiers in (195).

Define

$$\bigwedge_{i \le k} (a_i \le b_i) \iff \sum_{i \le k} (a_i \dot{-} b_i) = 0. \tag{196}$$

From here, $\vec{a}, \vec{b}$ be integers which codes the sequence $a_1, \ldots, a_k$ and $b_1, \ldots, b_k$ respectively. Then, $\mathrm{PV}_1^-(D)$ proves

$$\bigwedge_{i \le k} (a_i \le b_i) \to a_j \le b_j \tag{197}$$

$$a_{k+1} \le b_{k+1} \wedge \bigwedge_{i \le k} (a_i \le b_i) \to \bigwedge_{i \le k+1} (a_i \le b_i) \tag{198}$$

by proofs of linear size respect to $k$ in which only seqeuents with a constant number of equations appear. Using $\bigwedge_{i \le k}(a_i \le b_i)$ allows us write multiple formulas $a_i \le b_i, i = 1, \ldots, k$ by a single formula. $\bigwedge_{i \le k}(a_i < b_i)$ is defined in the similar way.

**Lemma 24.** *Let $t(x_1, \ldots, x_k)$ be a term of* PV*. Then,* $\mathrm{PV}_1^-$ *proves*

$$\forall \vec{x}, y. \bigwedge_{i \le k} (|x_i| \le y) \wedge (1 < y) \to |t(\vec{x})| \le \mathrm{sq}^{\mathrm{size}(t)}(y) \tag{199}$$

*with a proof in a tree form whose number of nodes is quadratic in* $\mathrm{size}(t)$*, which only contains sequents with a constant number $D$ of quantifiers and equations.*

*Proof.* The same as in [2] holds. The proof only uses sequents with a constant number $D$ of quantifiers and equations. Further, we note that (199) has quantifiers, which allows a tree form proof instead of a sequential proof. □

**Lemma 25.** *For all natural number $n$, there is a* $\mathrm{PV}_1^-$ *proof of*

$$||\,\mathrm{sq}^n(x)|| \le n + ||x|| \tag{200}$$

*whose number of nodes is quadratic in $n$, and only contains sequents with a constant number $D$ of quantifiers and equations.*

**Corollary 2.** *Let $t(x_1, \ldots, x_k)$ be a term of* PV*. Then,* $\mathrm{PV}_1^-$ *proves*

$$\forall \vec{x}, y. \bigwedge_{i \le k} (||x_i|| \le y) \wedge (1 < y) \to |||t(\vec{x})||| \le \mathrm{size}(t) + ||y|| \tag{201}$$

*with a proof in a tree form whose number of nodes is quadratic in* $\mathrm{size}(t)$*, which only contains sequents with a constant number $D$ of quantifiers and equations.*

Next, we apply Solovay's cut shortening technique to PV and $\mathrm{PV}_1^-$. Let $2^x_{|y|} = 2^{\min\{x, |y|\}}$. Let $A(x)$ is an equation of PV with a free variable $x$ and $t$ be a term which does not have $y$ and $z$ as free variables. We define

$$A_0(y) = \forall x. |x| \le y \to A(x) \tag{202}$$

$$A_1^t(z) = \forall y, y' \le t. (y' \le y \wedge y \le y' + 2^z_{|t|} \wedge A_0(y')) \to A_0(y) \tag{203}$$

$$A_2^t(w) = \forall z, z' \le |t|. (z' \le z \wedge z \le z' + 2^w_{||t||} \wedge A_1^t(z')) \to A_1^t(z) \tag{204}$$

$$A_3^t(v) = \forall w, w' \le |||t|||. (w' \le w \wedge w \le w' + 2^v_{|||t|||} \wedge A_2^t(w')) \to A_2^t(w). \tag{205}$$

**Lemma 26.** *The following formulas are provable in* $\mathrm{PV}_1^-$*.*

$$A_0(y) \wedge y' \le y \to A_0(y') \tag{206}$$

$$A(\varepsilon) \to A_0(0) \tag{207}$$

$$\forall x. (|x| \le t \wedge A(x) \to A(0x) \wedge A(1x)) \to \forall y \le t. (A_0(y) \to A_0(y+1)) \tag{208}$$

*Proof.* Immediate. □

**Lemma 27.** *The following formulas are provable in* $\mathrm{PV}_1^-$.

$$A_1^t(z) \wedge z' \leq z \to A_1^t(z') \tag{209}$$

$$\forall y \leq t.(A_0(y) \to A_0(y+1)) \to A_1^t(0) \tag{210}$$

$$\forall z \leq |t|.A_1^t(z) \to A_1^t(z+1) \tag{211}$$

$$A_1^t(|t|) \to A_0(0) \to A_0(t) \tag{212}$$

*Proof.* (209) and (210) are trivial.

To prove (211), assume $A_1^t(z)$. Let $y, y'$ be integers such that $y' \leq y$ and assume that $y \leq y' + 2_{|t|}^{z+1}$. Assume $A_0(y')$. If $y \leq y' + 2_{|t|}^z$, then by $A_1^t(z)$ we have $A_0(y)$. In particular, $A_0(y' + 2_{|t|}^z)$. If $y' + 2_{|t|}^z < y \leq y' + 2_{|t|}^{z+1}$, by letting $y' := 2_{|t|}^z$ and $y := y$ in $A_1^t(z)$, we have

$$(y' + 2_{|t|}^z \leq y \wedge y \leq y' + 2_{|t|}^z + 2_{|t|}^z \wedge A_0(y' + 2_{|t|}^z)) \to A_0(y) \tag{213}$$

Therefore, we have $A_0(y)$.

To prove (212), let $y' = 0$ and $y = t$ in $A_1^t(|t|)$. Then we obtain $A_0(t)$. □

**Lemma 28.** *The following formulas are provable in* $\mathrm{PV}_1^-$.

$$A_2^t(w) \wedge w' \leq w \to A_2^t(w') \tag{214}$$

$$A_2^t(0) \tag{215}$$

$$\forall w \leq ||t||.A_2^t(w) \to A_2^t(w+1) \tag{216}$$

$$A_2^t(||t||) \to A_1^t(0) \to A_1^t(|t|) \tag{217}$$

*Proof.* The same as the proof of Lemma 27 . □

**Lemma 29.** *The following formulas are provable in* $\mathrm{PV}_1^-$.

$$A_3^t(v) \wedge v' \leq v \to A_2^t(v') \tag{218}$$

$$A_3^t(0) \tag{219}$$

$$\forall v \leq |||t|||.A_3^v(w) \to A_3^t(v+1) \tag{220}$$

$$A_3^t(|||t|||) \to A_2^t(0) \to A_2^t(||t||) \tag{221}$$

*Proof.* The same as the proof of Lemma 27 . □

Note that Lemmas the proofs of 27, 28 and 29 has the same structure regardless to $A(x)$. Therefore, the number of quantifiers and equations in the proofs of these lemmas is bounded by a constant $D$.

**Proposition 3** ($S_2^1$)**.** *Let $D$ be the fixed integer which we take it large enough. Assume that $t(x_1, \ldots, x_n) = u(x_1, \ldots, x_n)$ is provable in* $\mathrm{PV} + \mathbf{Ax}_e$. *For any integers $m_1, \ldots, m_n$, we have*

$$\mathrm{PV}_1^-(D) + \mathbf{Ax} \vdash \bigwedge_{i \leq n} (|x_i| < \mathrm{sq}^{m_i}(2)) \implies t(\vec{x}) = u(\vec{x}) \tag{222}$$

*Proof.* Let $A(\vec{x})$ be $t(x_1, \ldots, x_n) = u(x_1, \ldots, x_n)$. By induction on the proof of $A(\vec{x})$ in PV. We only consider the substitution rule and the induction rule.

$$\frac{t(\vec{x}, y) = u(\vec{x}, y)}{t(\vec{x}, r(\vec{x})) = u(\vec{x}, r(\vec{x}))} \tag{223}$$

Let $m = \max\{m_1, \ldots, m_n\}$. By Lemma 24, $\mathrm{PV}_1^-$ can prove

$$\bigwedge_{i\leq n}(|x_i| < \mathrm{sq}^{m_i}(2)) \implies |r(\vec{x})| \leq \mathrm{sq}^{m+\mathrm{size}(r)}(2). \tag{224}$$

By induction hypothesis, we have

$$\bigwedge_{i\leq n}(|x_i| < \mathrm{sq}^{m_i}(2)) \implies t(\vec{x}, r(\vec{x})) = u(\vec{x}, r(\vec{x})). \tag{225}$$

Therefore, the proposition is proved.

(226)
$$\frac{t_1(\varepsilon,\vec{y}) = t_2(\varepsilon,\vec{y}) \quad t_1(bz,\vec{y}) = u_b(t_1(z),\vec{y}) \quad t_2(bz,\vec{y}) = u_b(t_2(z),\vec{y}) \quad (b=0,1)}{t_1(x,\vec{y}) = t_2(x,\vec{y})}$$

Let $|x| < \mathrm{sq}^{m_0}(2), |y_1| < \mathrm{sq}^{m_1}(2), \ldots, |y_n| < \mathrm{sq}^{m_n}(2)$ be the bound for variables $x, \vec{y}$. Let $A(x,\vec{y})$ be $t_1(x,\vec{y}) = t_2(x,\vec{y})$. Assume that we bound $|x|$ by $\mathrm{sq}^s(2)$. By induction hypothesis, $\mathrm{PV}_1^-$ proves

$$\bigwedge_{i\leq n}(|y_i| < \mathrm{sq}^{m_i}(2)) \implies t_1(\varepsilon,\vec{y}) = t_2(\varepsilon,\vec{y}) \tag{227}$$

(228)
$$\bigwedge_{i\leq n}(|y_i| < \mathrm{sq}^{m_i}(2)), |z| \leq \mathrm{sq}^s(2), t_1(z,\vec{y}) = t_2(z,\vec{y}) \implies t_1(bz,\vec{y}) = t_2(bz,\vec{y}) \quad (b=0,1)$$

By letting $A(z) := t_1(z,\vec{y}) = t_2(z,\vec{y})$ we have $A_0(0)$. Further, by letting $t := \mathrm{sq}^s(2)$ in (208), we have

$$\bigwedge_{i\leq n}(|y_i| < \mathrm{sq}^{m_i}(2)) \implies \forall y \leq \mathrm{sq}^s(2).(A_0(y) \to A_0(y+1)) \tag{229}$$

Therefore, by (210), we have $A_1^t(0)$. By (219) and (220), togather with Lemma 25, we have the $\mathrm{PV}_1^-$-proof of $A_3^t(s+2)$ with the size quadratic to $s$. Therefore, by (217) and (215), we have $A_2(|t|)$. Further, by (217) and $A_1^t(0)$, we have $A_1(|t|)$. Finally, by (212) and $A_0(0)$, $A_0(t)$ is obtained. Therefore, $\mathrm{PV}_1^-$ proves

$$\bigwedge_{i\leq n}(|y_i| < \mathrm{sq}^{m_i}(2)), |x| \leq \mathrm{sq}^s(2) \implies A(x) \tag{230}$$

The proposition is proved.

It is easy to see that all sequencts which appear in the constructed $\mathrm{PV}_1^-$ proofs have quantifiers and equations less than a fixed constant $D$. □

**Corollary 3.** *$S_2^1$ does not prove the consistency of $\mathrm{PV}_1^-(D) + \mathbf{Ax}$, if $D$ is large enough.*

*Proof.* Let $\mathbf{Ax}_e$ be equations which represent $\mathbf{Ax}$. By Proposition 3, $S_2^1$ proves

$$\mathrm{Prf}_{\mathrm{PV}+\mathbf{Ax}_e}(p, \lceil 0=1\rceil) \to \mathrm{Prf}_{\mathrm{PV}_1^-(D)+\mathbf{Ax}}(f(p), \lceil 0=1\rceil) \tag{231}$$

for a polynomial-time function $f$. Therefore, $S_2^1$ proves

$$\mathrm{Con}(\mathrm{PV}_1^-(D) + \mathbf{Ax}) \to \mathrm{Con}(\mathrm{PV} + \mathbf{Ax}_e). \tag{232}$$

Because we can prove $\mathrm{PV} + \mathbf{Ax}_e \nvdash \mathrm{Con}(\mathrm{PV} + \mathbf{Ax}_e)$ in the same way as [5], we have $S_2^1 \nvdash \mathrm{Con}(\mathrm{PV}_1^-(D) + \mathbf{Ax})$ using Lemma 23. □

## 9. Separation of bounded arithmetics and a proof of $P \neq NP$

Let $D$ be a large integer. By Corollary 3, $S^1_2 \nvdash \mathrm{Con}(\mathrm{PV}^-_1(D)+\mathbf{Ax})$. By Theorem 3, $S^{D+2}_2 \vdash \mathrm{Con}(\mathrm{PV}^-_1(D)+\mathbf{Ax})$. Therefore, $S^1_2 \subsetneq S^{D+2}_2$. Further, if P = NP, we obtain a contradiction as in [7]. Therefore, P $\neq$ NP.

## References


[1] A. Beckmann. Proving consistency of equational theories in bounded arithmetic. *Journal of Symbolic Logic*, 67(1):279–296, mar 2002.

[2] S. R. Buss and A. Ignjatović. Unprovability of consistency statements in fragments of bounded arithmetic. *Annals of pure and applied Logic*, 74:221–244, 1995.

[3] Samuel R. Buss. *Bounded arithmetic*. Bibliopolis, 1986.

[4] S. Cook and A. Urquhart. Functional interpretations of feasibly constructive arithmetic. *Annals of Pure and Applied Logic*, 63(2):103–200, sep 1993.

[5] Stephen A. Cook. Feasibly constructive proofs and the propositional calculus (preliminary version). In *Proceedings of seventh annual ACM symposium on Theory of computing*, pages 83–97. ACM, 1975.

[6] Michał Krynicki and Marcin Mostowski. Henkin quantifiers. In *Quantifiers: logics, models and computation*, pages 193–262. Springer, 1995.

[7] Gaisi Takeuti. Incompleteness theorems and $S^i_2$ versus $S^{i+1}_2$. In *Logic Colloquium '96: Proceedings of the Colloquium held in San Sebastián, Spain, July 9-15, 1996*, volume 12 of *Lecture notes in logic*, pages 247–261, 1996.

[8] Yoriyuki Yamagata. Consistency proof of a fragment of pv with substitution in bounded arithmetic. *The Journal of Symbolic Logic*, 83(3):1063–1090, 2018.



National Institute of Advanced Science and Technology (AIST), 1-8-31 Midorigaoka, Ikeda, Osaka 563-8577 Japan

*URL*: `https://staff.aist.go.jp/yoriyuki.yamagata/en/`

*Email address*: `yoriyuki.yamagata@aist.go.jp`

「マス・フォア・インダストリ研究」シリーズ刊行にあたり

本シリーズは，平成 23 年 4 月に設立された九州大学マス・フォア・インダストリ研究所(IMI)が，平成 25 年 4 月に共同利用・共同研究拠点「産業数学の先進的・基礎的共同研究拠点」として，文部科学大臣より認定を受けたことにともない刊行するものである．本シリーズでは，主として，マス・フォア・インダストリに関する研究集会の会議録，共同研究の成果報告等を出版する．各巻はマス・フォア・インダストリの最新の研究成果に加え，その新たな視点からのサーベイ及びレビューなども収録し，マス・フォア・インダストリの展開に資するものとする．

平成 30 年 10 月
マス・フォア・インダストリ研究所
所長 佐伯　修

**代数・論理・幾何と情報科学――理論から実世界への展開**

マス・フォア・インダストリ研究 No.17, IMI, 九州大学
ISSN 2188-286X
発行日　2020 年 2 月 10 日
編　集　河村　彰星，津曲　紀宏，西澤　弘毅，溝口　佳寛
発　行　九州大学マス・フォア・インダストリ研究所
〒819-0395 福岡市西区元岡 744
九州大学数理・IMI 事務室
TEL 092-802-4402　FAX 092-802-4405
URL http://www.imi.kyushu-u.ac.jp/
印　刷　城島印刷株式会社
〒810-0012 福岡市中央区白金 2 丁目 9 番 6 号
TEL 092-531-7102　FAX 092-524-4411

# シリーズ既刊

| Issue | Author ／ Editor | Title | Published |
|---|---|---|---|
| マス・フォア・インダストリ研究　No.1 | 穴田　啓晃<br>安田　貴徳<br>Xavier Dahan<br>櫻井　幸一 | Functional Encryption as a Social Infrastructure and Its Realization by Elliptic Curves and Lattices | 26 February 2015 |
| マス・フォア・インダストリ研究　No.2 | 滝口　孝志<br>藤原　宏志 | Collaboration Between Theory and Practice in Inverse Problems | 12 March 2015 |
| マス・フォア・インダストリ研究　No.3 | 筧　三郎 | 非線形数理モデルの諸相：連続，離散，超離散，その先<br>(Various aspects of nonlinear mathematical models : continuous, discrete, ultra-discrete, and beyond) | 24 March 2015 |
| マス・フォア・インダストリ研究 No.4 | 穴田　啓晃<br>安田　貴徳<br>櫻井　幸一<br>寺西　勇 | Next-generation Cryptography for Privacy Protection and Decentralized Control and Mathematical Structures to Support Techniques | 29 January 2016 |
| マス・フォア・インダストリ研究 No.5 | 藤原　宏志<br>滝口　孝志 | Mathematical Backgrounds and Future Progress of Practical Inverse Problems | 1 March 2016 |
| マス・フォア・インダストリ研究 No.6 | 松谷　茂樹<br>佐伯　修<br>中川　淳一<br>上坂　正晃<br>濵田　裕康 | 結晶のらせん転位の数理 | 10 January 2017 |
| マス・フォア・インダストリ研究 No.7 | 滝口　孝志<br>藤原　宏志 | Collaboration among mathematics, engineering and industry on various problems in infrastructure and environment | 1 March 2017 |
| マス・フォア・インダストリ研究 No.8 | 藤原　宏志<br>滝口　孝志 | Practical inverse problems based on interdisciplinary and industry-academia collaboration | 20 February 2018 |
| マス・フォア・インダストリ研究 No.9 | 阿部　拓郎<br>高島　克幸<br>縫田　光司<br>安田　雅哉 | 代数的手法による数理暗号解析<br>Workshop on analysis of mathematical cryptography via algebraic methods | 1 March 2018 |
| マス・フォア・インダストリ研究 No.10 | 阿部　拓郎<br>落合　啓之<br>高島　克幸<br>縫田　光司<br>安田　雅哉 | 量子情報社会に向けた数理的アプローチ<br>Mathematical approach for quantum information society | 26 December 2018 |

| Issue | Author ／ Editor | Title | Published |
|---|---|---|---|
| マス・フォア・インダストリ<br>研究 No.11 | 松谷　茂樹<br>佐伯　修<br>中川　淳一<br>濵田　裕康<br>上坂　正晃 | 結晶転位の先進数理解析<br>Advanced Mathematical Investigation for Dislocations | 7 January 2019 |
| マス・フォア・インダストリ<br>研究 No.12 | 滝口　孝志 | Non-destructive inspection for concrete structures and related topics | 13 February 2019 |
| マス・フォア・インダストリ<br>研究 No.13 | 宇波　耕一<br>長野　智絵<br>吉岡　秀和<br>田上　大助<br>白井　朋之 | 数理農学における時系列データのモデル化と解析<br>Modeling and Analysis of Time Series Data in Math-Agro Sciences | 28 February 2019 |
| マス・フォア・インダストリ<br>研究 No.14 | 佐久間　弘文<br>大津　元一<br>小嶋　泉<br>福本　康秀<br>山本　昌宏<br>納谷　昌之 | ドレスト光子に関する基礎的数理研究 | 18 March 2019 |
| マス・フォア・インダストリ<br>研究 No.15 | 松谷　茂樹<br>佐伯　修<br>中川　淳一<br>濵田　裕康<br>富安　亮子 | 結晶の界面，転位，構造の先進数理解析 | 2 December 2019 |
| マス・フォア・インダストリ<br>研究 No.16 | Takuro Abe<br>Yasuhiko Ikematsu<br>Koji Nuida<br>Yutaka Shikano<br>Katsuyuki Takashima<br>Masaya Yasuda | Quantum computation, post-quantum cryptography and quantum codes | 17 January 2020 |

Institute of Mathematics for Industry
Kyushu University


九州大学マス・フォア・インダストリ研究所

〒819-0395 福岡市西区元岡744

http://www.imi.kyushu-u.ac.jp