SEAMAIL

Newsletter from Software Engineers Association

Vol. 15, Number 9 November, 2007

目　次

ソフトウェア技術者協会

Software Engineers Asociation

ソフトウェア技術者協会 (SEA) は，ソフトウェアハウス，コンピュータメーカ，計算センタ，エンドユーザ，大学，研究所など，それぞれ異なった環境に置かれているソフトウェア技術者または研究者が，そうした社会組織の壁を越えて，各自の経験や技術を自由に交流しあうための「場」として，1985年 12月に設立されました．

その主な活動は，機関誌 SEAMAIL の発行，支部および研究分科会の運営，セミナー／ワークショップ／シンポジウムなどのイベントの開催，および内外の関係諸団体との交流です．発足当初約 200人にすぎなかった会員数もその後増加し，現在，北は北海道から南は沖縄まで，300 余名を越えるメンバーを擁するにいたりました．法人賛助会員も 15社を数えます．支部は，東京以外に，関西，横浜，名古屋，九州，広島，東北の各地区で設立されており，その他の地域でも設立準備をしています．分科会は，東京，関西，名古屋で，それぞれいくつかが活動しており，その他の支部でも，月例会やフォーラムが定期的に開催されています．

「現在のソフトウェア界における最大の課題は，技術移転の促進である」といわれています．これまでわが国には，そのための適切な社会的メカニズムが欠けていたように思われます．SEA は，そうした欠落を補うべく，これからますます活発な活動を展開して行きたいと考えています．いままで日本にはなかったこの新しいプロフェッショナル・ソサイエティの発展のために，ぜひとも，あなたのお力を貸してください．

代表幹事： 田中一夫

常任幹事： 荒木啓二郎 熊谷章 高橋光裕 中野秀男

幹事： 石川雅彦 落水浩一郎 窪田芳夫 蔵川圭 小林修 小林允 近藤康ニ
桜井麻里 酒匂寛 塩谷和範 篠崎直二郎 新谷勝利 新森昭宏 杉田義明
鈴木裕信 玉井哲雄 中来田秀樹 奈良隆正 野中哲 野村行憲 野呂昌満
端山毅 平尾一浩 藤野誠治 松原友夫 渡邊雄一

事務局長： 岸田孝一

会計監事： 吉村成弘 橋本勝

分科会世話人 環境分科会(SIGENV)：塩谷和範 田中慎一郎 渡邊雄一
教育分科会(SIGEDU)：君島浩 篠崎直二郎 杉田義明 米島博司 森泉清
ネットワーク分科会(SIGNET)：人見庸 松本理恵
プロセス分科会 (SEA-SPIN))：伊藤昌夫 塩谷和範 新谷勝利 高橋光裕 田中一夫 端山毅 藤野誠治
フォーマルメソッド分科会(SIGFM)：荒木啓二郎 伊藤昌夫 熊谷章 佐原伸 張漢明 山崎利治
オープンソース分科会(SIGOSS)：石川雅彦 岸田孝一 杉田義明 鈴木裕信 中野秀男

支部世話人 関西支部：小林修 中野秀男 横山博司
横浜支部：野中哲 藤野晃延 北條正顕
名古屋支部：石川 雅彦 角谷裕司 野呂昌満
九州支部：荒木啓二郎 武田淳男 平尾一浩
広島支部：佐藤康臣 谷純一郎
東北支部：布川博士 野村行憲

賛助会員会社：ＳＲＡ ＰＦＵ オムロンソフトウェア キヤノン 新日鉄ソリューションズ ダイキン工業
オムロン 富士電機リテイルシステムズ NTTデータ ヤマハ オープンテクノロジーズ ＳＲＡ西日本
ＳＲＡ東北 エフビクス 電盛社
（以上15社）

ＳＥＡＭＡＩＬ Ｖｏｌ．１５， Ｎｏ．９ ２００７年１１月２０日発行 編集人 岸田 孝一
発行人 ソフトウェア技術者協会（ＳＥＡ）
〒１６０-０００４ 東京都新宿区四谷３－１２ 丸正ビル５Ｆ
T: 03-3356-1077 F: 03-3356-1072 E-mail: sea@sea.or.jp URL: http://www.sea.jp/
印刷所 市田印刷株式会社 〒１１４-００１４ 東京都北区田端２－３－２５
定価 ５００円 （禁無断転載）

SEA Forum September 2007

プログラミングにとって数学的思考は役に立つのか？

- ソフトウエアと数学のあいだ -

社会を支えるインフラストラクチャとしてのソフトウエアの責任は日々増大し，プログラム上のほんの軽微な欠陥も，時としてわれわれの生活に大きな影響を与えるようになってきました．したがって，ソフトウエアの品質を確保することは，きわめて重要な課題になりつつあります，今回のフォーラムでは，もう一歩原点に立ち帰って，プログラムの仕様化・設計およびコーディングにおける「思考」をテーマとして取り上げます．

「できる」プログラマが書いたプログラムはどうして「よい」プログラムになるのでしょうか？ 「よい」プログラムを書くうえで，数学的思考はどのように寄与するのでしょうか？ プログラミングに対する数学的アプローチ思考の意味するところについては，長年さまざまな角度から議論されてきましたが，しかし，抽象的・形式的・数学的思考の効用は，一部のアカデミズムの世界を除いてはあまり重要視されず，特に産業界においては，プログラミングのための数学的思考やプログラマの数学的素養といった問題は，ほとんど無視されてきたように思われます．

今月のフォーラムでは，プログラミングと数学の関係に焦点をあてて，ソフトウエア開発における数学の位置づけについて、真剣に議論してみたいと考えています．晩夏のひととき，この本質的な問題をめぐっての議論を楽しみませんか？

多くの方々の参加をお待ちしています。

開催要領

日時:2007 年 9 月 14 日(金)13:00-17:00

場所:新宿歴史博物館 2階ホール

東京都新宿区三栄町22(地下鉄丸の内線「四谷3丁目」駅から徒歩8分)

地図は:

http://www.regasu-shinjuku.or.jp/shinjuku-rekihaku/public_html/access.html

プログラム

12:30-13:00 受付

13:00-14:00 講演 「数学・数学的」とは　山崎利治(フリー)

数値解析・ORなどの応用事例ではなく，プログラミングやソフトウエア工学,そのものの中に数学がどうあらわれたか？ あるいは，プログラミングなどに出現する対象がいかにして数学の対象になったか？

14:15-17:00 パネル討論

テーマ: ソフトウエア工学・プログラミングにおいて，私が数学的思考を可とする，あるいは無意味とする理由は？

司会: 野中哲(トゥルーロジック)

パネラー: 荒木啓二郎(九州大学)

権藤克彦(東京工業大学)

玉井哲雄(東京大学)

山崎利治(フリー)

SEA フォーラムを終えて

野中 哲

（トゥルーロジック）

このフォーラムのきっかけとなったのは、日頃お世話になっている元日本ユニシスの山崎さんとの、新橋にある酒亭での雑談でした。以前から「圏論について勉強しなさい」というご指導を受けていたので、その席では圏論の勉強会を開催し講師をお願いしたいというお願いをしました。山崎さんにはご快諾頂き、その後の SEA の幹事会で、この勉強会の構想を紹介したところ「せっかくだから、仲間を募る意味からも、もう少し間口を広げて多くの人向けに数学とプログラミングの関係を論ずるフォーラムを開催してはどうか？」という意見が出て、今回のフォーラムの開催となりました。

企画段階では、このタイトルでいったい何人の人が参加してくれるだろうかという心配もしましたが、蓋をあけてみると三十余名という、思いのほか多くの人が参加されたので、少々驚きました。当日、私はパネルディスカッションの司会をさせていただきましたが、フォーラムの感想を一言でいえば「楽しかった！」の一言に尽きます。発表者の方々からいろいろ興味深いお話を聞くことができたうえに、これから勉強しなければいけない課題もたくさん見つけることができたので、個人的には大変収穫の多いフォーラムでした。

私個人の数学体験といえば、学生時代はあまり数学に強い興味があったとはいえません。大学の教養科目での線形代数や微積分といった授業も面白いと思ったことはほとんどありませんでした。もちろん成績も単位取得ぎりぎり。後年、数学を面白いと思うようになったのは、おそらく、かの有名な「ゲーデル・エッシャー・バッハ」を読んでからだと思いいます。この本で集合論の危機、ラッセルのパラドックといったものを知り、そこから、抽象代数、位相といった分野にも興味が広がっていきました。

その過程で感じたのは、20 世紀数学の辿った抽象化の道は「よい」プログラムを書く上で参考となる部分が多いのではないだろうか？ということでした。抽象位相や圏といった概念は、数学的事象の抽象化を極限まで推し進めて非常に単純化されています。その一方、その定義の単純さからは信じられないような驚くほど豊かな議論が展開されます。この一連のプロセスは、共通した部分をなるべく再利用可能な手続きに切り取り出して、プログラムを短くしようという「よい」プログラミング態度との共通点が多いように感じられます。

私は、本格的に数学の勉強をしたわけではありませんが、今後とも「よい」プログラムを書くために、少しずつでも数学の勉強を続けていきたいと考えています。また、今回のフォーラムがあまりにも楽しかったので、毎年恒例のイベントに仕立て上げてしまおうか、などという野望なども芽生えてきました。もしもご賛同いただける方がおられましたら、ぜひ声をかけて下さい。

数学的思考とプログラミング

山崎利治

数学的思考はプログラミングに有効か？

プログラミングを分野に依存しない側面，つまり，算法を無視して考えたい．すなわち，ソフトウエア工学が対象とする分野に議論を限りたい．数値解析，オペレーションズ・リサーチ，先物取引，暗号系，制御理論を使う組込系など，課題対象が数学的な成果を要求するようなものは今日は対象外としたい．

今日の話題

- 「数学」と「数学的」とプログラミング
- 記憶を辿って (数学的思考が役立った例)
- 数学的を良しとすれば 数学の道具が使える　道具箱としての圏
- 情報科学：ソフトウエア工学＝物理学：電気工学

プログラムとは

プログラムとは

- 計算機を動かす処方箋である
- 数学的な定理である

という見方がある.
後のそれではプログラムが仕様の模型であることを厳密に証明しなければならない．それは数学的な動といえる.

A.P.Ershov

AFIPS 主催の SJCC における昼食会での講演で Ershov はプログラマに必要な能力についてつぎをあげている：

- 第一級の数学者の論理性
- エヂソンのような技術者の才能
- 銀行員の精確さ
- 推理作家の発想力
- ビジネスマンの勤勉さ
- 同僚との協調性

第一項について，私は，むしろ，「数学者の模型・理論構成能力」としたい.」

* Aesthetics and the human factor in programming.CACM,1972.

数学

数学自体は議論しない．数学はプログラミングとは無関係のようである.

「数学は人間精神の名誉のためだけに存在する

> **Fourierは数学の主目的は自然現象の解明と公共の効用のためといったが，究極の目的は人間精神の名誉のためだけに存在し，その限りにおいて数の問題が自然についての問題とおなじ価値をもつのである.」(Carl Jacobi).**

ただ，他分野のひとが勝手に実用に供するわけである.

数学的思考・数学活動とは

私は数学者ではないので直接に答えられない．つぎのようなことではないかと想像する．

- 関心の対象とそれが置かれた環境とを観察し
- それを抽象・一般化・公理化し
- その模型ないし理論を形成する
- それを厳密な言語・推論によって行う

そうなら数学活動はプログラミングに役立つ!

数学活動

数学活動 (S.Mac Lane,Mathematics:Form and Function.)

Activity	Idea	Formulation
Collecting	Collection	Set(of elements)
Counting	Next	Successor;order
		Ordinal number
Comparing	Enumeration	Bijection
		Cardinal number
Computing	Combination(of nos)	Rules for addition
		Rules for multiplication
		Abelian group

数学活動

Activity	Idea	Formulation
Rearranging	Permutation	Bijection
		Permutation group
Timing	Before and after	Linear order
Observing	Symmetry	Transformation group
Building,shaping	Figure;symmetry	Coollection of points
Measuring	Distance;extent	Metric space
Moving	Change	Rigid motion
		Transformation group
		Rate of change

数学活動

Activity	Idea	Formulation
Estimating	Approximation	Continuity
		Limit
	Nearby	Topological space
Selecting	Part	Subset
		Boolean algebra
Arguing	Proof	Logical connectives
Choosing	Chance	Probability(favorable/total)
Successive action	Followed by	Composition
		Transformation group

数学的思考，つまり，理論や模型の形成が必要であった事例を想起する．個人の体験以外にもそのように感じたものも記す．

Whirlwind のための J.H.Laning - W.Zierler 系 (算術式解釈系)　(Sarvomechanisms Laboratory MIT, 1953.)

「演算子文法はまだなく，翻訳プログラムは難しく，正しく解析できない正しい算術式がないかと心配した」(Laning)

Warnier 法と ISP

３本の順編成ファイルの併合プログラムは初心プログラマにとってやさしくない．どう考えどうプログラムするかの指針が必要であった．
Warnier 法や Jackson Structured Programming はそのようなプログラムの作成方法である．

コンサルタントの常で，Warnier や Jackson は手の内を明かさない．

昔，事務計算は順編成ファイル (磁気テープ) を利用しての一括処理プログラムが中心であった．そのころ CODASYL が Information Algebra* を発表した．そのような仕様からコボル・コード生成をねらった処理系を作成したことがある．彼らの云う bundle をファイバー積，glump をファイバー和と整理して，仕様記述，生成系ともに簡素化することができた．

bundle は「突合せ」を，glump は「集計」をそれぞれ定義するためのファイル (area) 構成子である．

* R.Bosak,et al., An information algebra: phase I report. CACM,(5,4),1962 pp190-204.

事務システムの開発で概念データ設計と称して ER 模型が使われてきた．
オブジェクト指向・UML の流行からクラス図を使わなければならないという脅迫観念が取沙汰されている．
この両者は同一視できる．たとえば

ER 図＋ SQL ＝クラス図＋ OCL

静構造に専念する ER 図に対してクラス図はクラス間相互作用の動構造を並行して考慮するだけである．

ところで，同一視できるとはどのようにいえば説得できるか？

系全体であれプログラムの断片であれ，その仕様記述は開発の基礎として最重要である．仕様記述についてプログラム言語の仕様書から多くを学んだ．

- Algol 60(17 頁), Algol 68(236 頁), Algol N(＜ 70 頁)
- The Definition of Standard ML(R.Milner et al.)(114 頁)

これらの仕様書は数学的思考の産物だと思う．それだけに読解には努力が必要になる．
「涙なしの **Algol 68**」という本があった．
cf. Java や UML の仕様書　これらの浩瀚さと上の簡潔さを対比せよ

数学の道具

数学的思考を良しとすれば，数学で常用の道具が利用できる．

それらにつぎがある：

- 集合と写像の概念
- 圏・関手・関射の概念

前者は Z や VDM などで馴染みである．後者は abstract nonsense と云われながらも，前者に代って論議領域の記述・理論の形成に数学以外でも実に有用であった．後者について一言したい．

圏

圏とは極めて簡単な数学上の概念で，単位半群の抽象化，あるいは，多重有向グラフにいくつかの制約を加味したものである．

圏 $\mathbb{C}$ とは有向多重グラフ $(s,t : A \to O)$ のことである．ただし，

- 集合 A, O をそれぞれ射，対象という．
- 写像 s,t は射の元と先を示す．射 f が $s(f) = A, t(f) = B$ のとき $A \overset{f}{\to} B$ などと書く．
- $i : O \to A$ があって，各対象 A に対して $i(A) \overset{\Delta}{=} 1_A : A \to A$ を恒等射という．
- $A \overset{f}{\to} B \overset{g}{\to} C$ のとき，射の合成という射 $g \circ f$ が存在する．
- 合成 $\circ$ は結合的であり，合成に対して恒等射は左右単位要素になる．

コボル・コード生成系再訪

事務計算は入力ファイル群 F_i から一つの出力ファイル G を構成することとしてよい．その仕様を $S \triangleq (F = \phi(F_i), G, f : F \to G)$ とする．ここで F は F_i 上で構成関数 ϕ によって一つのファイルに纏めたものである．f は F 上で定義した関数である．ϕ は **bundle** と **glump** を構成する関数である．

ファイルやいくつかの定めたデータ領域を対象，関数を射として圏が構成できる．これは集合と関数の圏 Set と同等になる．Set は双完備である．そこでファイル構成演算 ϕ は空ファイルや単要素 (レコード) のファイルの構成・存在は自明であり，ファイバー積 (引戻) とファイバー和 (押出) を基本演算として用意すれば，(Information Algebra 的な) 事務計算を定義する仕様は上の枠組みで十分である．

NB (引戻，押出，始対象，終対象があれば有限双完備)

ER図＋SQL＝クラス図＋OCL

はなにをいっているのか？

本当は両者を統一して議論できる基盤を用意する必要がある．つまり,「データ・アーキテクト」の云う両「データ・モデル」の構成法が「同じ」にみえる理論を提示しなければならない．圏の用語を利用したいくつかの試論がみられる．

圏の利用

情報科学やソフトウエア工学での圏概念の活用は枚挙に暇がない．

- プログラムの数学的作成
- 抽象データ型の始代数意味論
- プロセスの余代数終意味論
- 型理論
- 方式記述言語構成

レビュー

仲間内のコードの精読は大切である．書かれたコードの良否を指摘するには局所，大域ともに，その理由を具体的・客観的に明確に述べなければならない．

- 解りにくい
- 効率的でない
- 変更しにくい・再利用しにくそう
- 頑強ではない

などの指摘のためには「理論」が必要である．たとえば，

仕様記述論，設計論，解析論，検査・検証論 など

無論，その現場から新しい理論ができることも望ましい．構造的プログラミングがソフトウエア工学の進展の基になったように．

仕様記述論

仕様に限らず設計その他なんであれ「記述」は大切である.
記述に関する超論がありうる. ソフトウエア開発関係でつぎのような項目が議論が可能であろう.

- 素材　(ZFC, トポス)
- 言語　(順序組言語, 列言語, 図式言語)
- 構造化機構　((余) 代数, (余) 単子 (co)monad)

素材は論議領域の存在・関係の記述方法を定めるものである. 集合と写像によるか, 対象と射によるかなどである. VDM のアイルランド派は圏論でいうトポスを基に考えている.
順序組言語は抽象データ型などを書いた代数的言語のことであり, 列言語は一般の論理的言語のことであり, 図式言語は UML や, 圏論でいう素描 (sketch) を基にした言語である.

[illegible]

構造化は仕様化・設計・コード化・検査・検証などすべてに必要である. つまり, 理論・模型作成が難しい複雑で巨大な系に対しては抽象・分割・合成が不可避であり, そのための機構を用意しなければならない.

- 抽象データ型 (代数) とプロセス (余代数)
- (余) 単子の利用 (monadic denotational semantics)
- 方式記述と allegories の利用

設計論

これは大域プログラミングの方法の議論である．分割と構造化，および，分割部分間の協調が主題である．

- 分割部分としての抽象データ型とプロセス
- 協調手段としてのプロセスの並行性
- 全体構造としての方式 (分割部分とその間の相互作用) とその記述
- 構造化機構としての (余) 単子

分析

プログラム分析方法は言語処理系作成技術によって発展した．それらはコード批評．改良を支援する．つぎは Nielson et al. の教科書* からの引用である．

- 流れ解析
- 制約中心解析
- 抽象解釈
- 型・効果系

* F. Nielson et al., Principles of Program Analysis. Springer, 1999.

検査・検証論

正しさの検査・検証作業は実に数学的な作業である.

- 古典的検証論の構造化
- 無限入力列に対応するオートマトンが時間論理式を満たすか？
- ブール式が真になる変数への割当があるか？

などが実行しやすいプログラムを書いているか？

情報科学：ソフトウエア工学 ＝ 物理学：電気工学
情報科学：ソフトウエア工学 ＝ 数学：情報科学

上の対比は成立しそうだから議論はもともと不要だった？

プログラミングにとって 数学的思考は役に立つのか

2007年9月14日

荒木 啓二郎
九州大学大学院システム情報科学研究院

遍歴

- 工学部 電気系(電気、電子、情報) 入学
  - 電子工学科志望:物性
- 工学部 情報工学科 配属
  - 「波動情報工学」講座
- 卒論:電磁波の数値計算 (Fortran + SSL)
  - 端点を有する誘電体による電磁波の散乱問題
  - Hankel 関数による近似式
    - 完備な関数空間、一様収束アルゴリズム
- 修士課程:プログラムの検証
  - Floyd-Hoare Logic
  - V. Pratt

役に立った(と思う)本

- 培風館 新数学シリーズ
  - 吉田夏彦:論理学, 培風館, 1958
  - 赤摂也:集合論入門, 培風館, 1959
    - 小型で薄いのが良かった
    - formalismの基本を学んだ
- ヒルベルト、アッケルマン(伊藤潔訳):記号論理学の基礎, 大阪教育図書, 1954
  (石本、竹尾訳:改訂最新版, 1974)
- 高木貞治:数の概念, 改版, 岩波書店, 1970

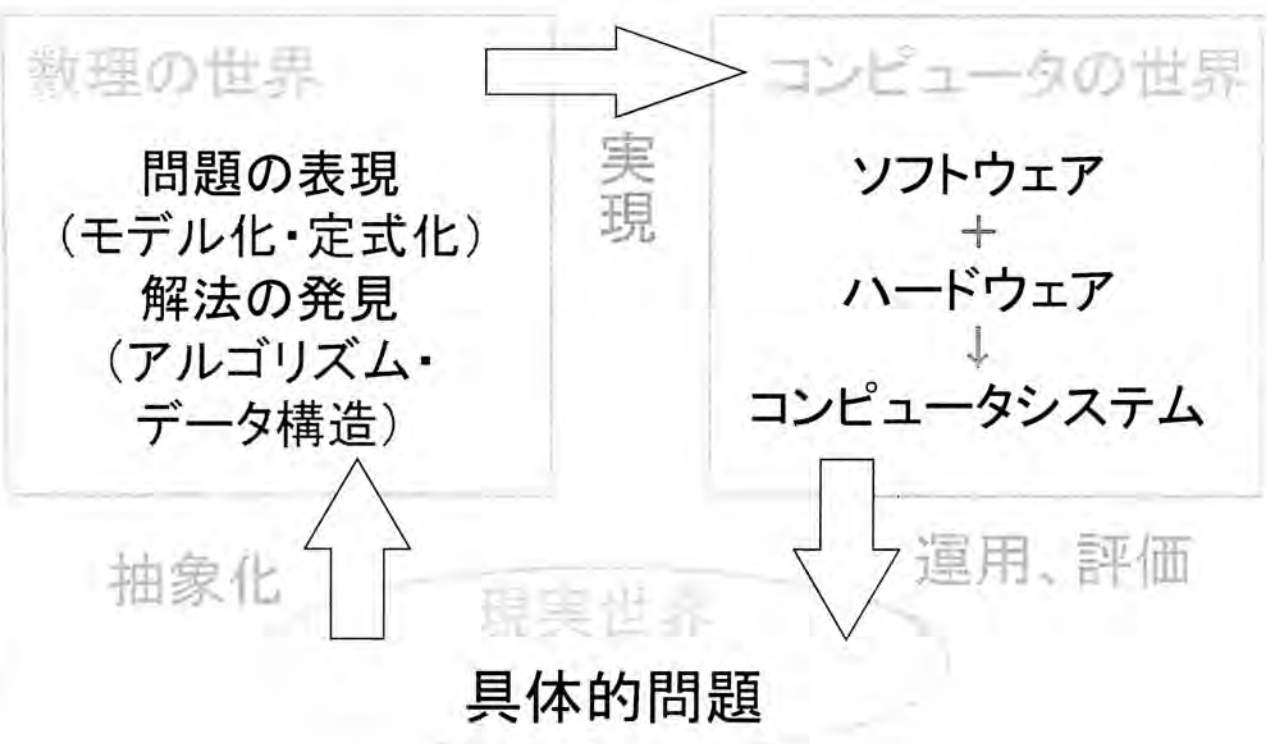
役に立った(と思う)本(続き)

- 島内剛一:数学の基礎, 日本評論社, 1971
- 広瀬健:数学的帰納法, 教育出版, 1979
- 小野寛晰:関係の代数, 教育出版, 1974
- Z. Manna (五十嵐滋訳):プログラムの理論, 日本コンピュータ協会, 1974
- 随筆、教養書
  - 岩波新書、岩波文庫、共立全書、学生社、etc.
    - 教科書を読んでも解らない
    - 教えてくれる人もいない
- *bit*

役に立った数学概念・理論

- 無限と連続
  - 裏表
- 帰納法
  - 帰納的定義
  - 帰納法による証明
- 型理論
  - 型検査
  - 多相型
  - データ抽象化

具象と抽象の間の往き来



形式手法 (formal methods)

- [A. Hall: Seven Myths of Formal Methods, IEEE Software, Vol.7, No.5, pp.11-19, 1990] より
  - ソフトウェアが完全であることを保証するわけではない
  - プログラムの証明だけではない
  - 高度な数学の知識を必要とするわけではない
  - 顧客の理解を助ける
- [J.P. Bowen and M.G. Hinchey: Seven More Myths of Formal Methods, IEEE Software, Vol.12, No.4, pp.34-41, 1995] より
  - 形式手法の人達は常に形式手法を用いるわけではない

形式手法の効用

- 直接
  - 生産物
    - プログラム、各種文書
  - 分析、証明
- 間接
  - 問題の理解
  - 概念の明確化
  - 認識、経験の共有
  - コミュニケーション

システムの数理モデル化

- 参照モデル
  - 「実際には、そんなに単純・綺麗ではない」という批判
  - モデルは現実とは違う
    - 「フォーマルモデル歌舞伎論」
- コミュニケーションのための道具
- 記述のための道具建て
  - e.g., Mathematical Toolkit in Z
- 分析・検証のための基礎

実は、一番役に立った本:-)

- プラトン(久保勉訳):ソクラテスの弁明・クリトン, 岩波文庫, 1964
- モデル化における「対話法」
  - ドメイン知識:しつこく訊く
  - 「汝自身を知れ」:専門家の意識覚醒

某大学における情報専門教育

- 「先導的ITスペシャリスト育成推進プログラム」
  - 実践重視
    - PBL (Project-Based Learning)
    - プロジェクト管理
    - インターンシップ
  - 基礎あっての実践教育
- 改組の計画
  - 基礎と実践・応用
  - 科学的思考力と工学的センス

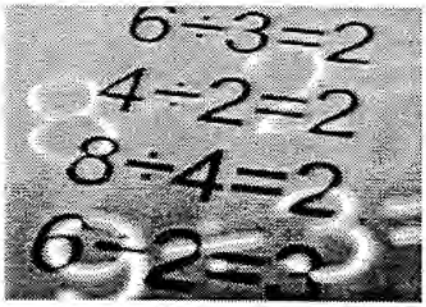
結論

- プログラミングにとって数学的思考は役に立つのか
  - 勿論！
- どの程度？
- いつ？

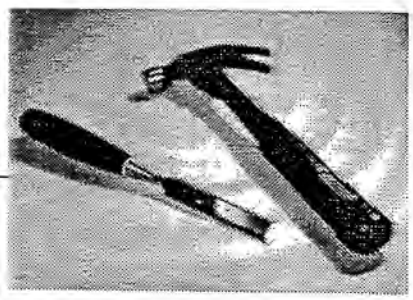
必須の基本的素養

「プログラミングにとって数学的思考は役に立つのか？」 - ソフトウエアと数学のあいだ -

権藤克彦
東京工業大学 計算工学専攻



自己紹介



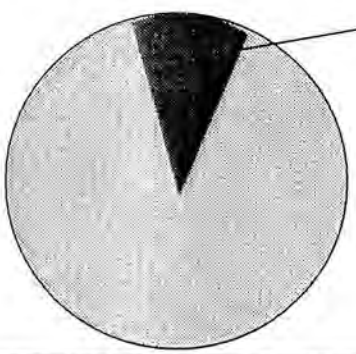
- 名前:権藤克彦
- 仕事:大学教員(准教授)
- 専門:ソフトウェア開発環境, ツール

- 大学に居ながらプログラマ魂を持つ.
  - 教育用OS udos, ANSI Cインタプリタ, 開発ツール (DWARF2-XML, TBCppA)などを開発.
- 昔は理論的な研究(属性文法)をしていた.

権藤のポジション

□ プログラミングに**数学的思考はあまり寄与しない**.

- 数学的思考が役立つ場面はある.
- 数学的思考以外に大切なことの方が多い.
- 大学は「数学的思考は役立つ」と言い過ぎ.



二極論はやめましょうね

たとえ話

□ 野球選手に「足の速さ」は必要か？

□ 足が速ければ，そりゃ有利でしょ.

□ でも，足さえ速ければいい，ってもんじゃない.

なぜ，そう思うか?

- データ構造やアルゴリズムは既存のものを使うことが多い.
- 他にも大切な能力は多くある.
  - 技術的要素：UNIXシグナルの正確なセマンティクス.
  - 社会学的要素：
    - そもそも厳しすぎる納期.
    - リスク管理，レビュー，コミュニケーション，プロジェクト管理.
  - 知的好奇心，芸術性，作文技術，推理力，弁護士の能力....

弁護士の能力

- プログラマは仕様書や規格と戦います.
  - 例：ISO/IEC 9899:1990 - C言語の規格
- 仕様書や規格はとても理解しづらい.
  - 法律にそっくり. →弁護士の能力が必要.
- このCコードの実行結果は?
  - 答え：未定義動作（違法なプログラム）

```
int x=0;
x = x++;
```

6.3 式：.... 直前の副作用完了点から次の副作用完了点までの間に，式の評価によってオブジェクトに格納された値を変更する回数は，高々1回でなければならない.

でも...(1)

- そこそこ数学が出来るので，無意識に「数学的思考」を使っているかも．
- ソフトウェア工学が進化すれば，将来は「数学的思考」がもっと重要になるかも．
- 「数学的思考」は良いプログラマを選別するメトリクス（のひとつ）として使えるかも．
- 「数学的思考」が重要な応用ドメインもある．

でも...(2)

- 最低限の数学的知識は欲しい．
  - 例：ド・モルガンの法則

```
if (!(A || B))   ⇔   if (!A && !B)
```

等価

- 線形代数の固有値・固有ベクトルの知識は必要？
  - 院試の頻出問題．
  - でも私はプログラムで使ったことがない．

数学的思考って何?

- 論理的思考?
  - プログラマは必ずしも論理的ではない.
- 抽象化を用いる手法?
  - 抽象化は漏れるので万能ではない.
- 形式的手法?
  - まだ現場では一般的には使われてない.
  - スケールする?コスト高?適用範囲?

たとえ話

- 棋士は自分の一手の発見法を説明できない.
  - NP完全問題を解いている訳ではない.
- 数学を勉強して身につくか?
  - たぶんNO.
- この数学的センス?
  は必要かも.

プログラマはあまり論理的ではない.

- 見積もり, リスク計算は直感や経験にも頼る.
- デバッグはアルゴリズミックに発見できない.
  - 演繹, 帰納ではなく, 仮説的推論(abduction)
  - 仮説的推論には豊富な経験・知識が必要.
- 形式的手法の導入に抵抗感を感じる(たぶん).
  - プログラマは論理的な思考を部分的に使う.
  - でも, 論理的な思考を全面的に強制されるのは嫌.

現場に必要なのは?

- あまり必要ないもの:数学的思考.
- 必要なもの:
  - 突貫工事ではなく, 品質を作りこむ時間・予算・人.
  - 新しい技術を学ぶ時間.
  - 休暇!
- 大学では社会人修士・博士を歓迎します.
  - 東京工業大学では先導的ITスペシャリスト人材育成推進プログラムを推進中.

なぜ大学は「数学的思考」を重要視?

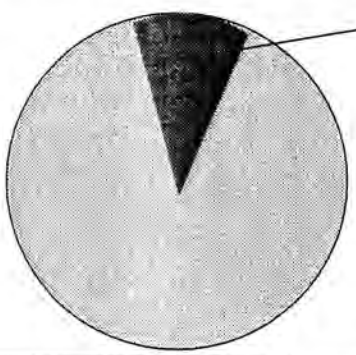
- プログラムを書けないから.
  - ≒大学の先生は真のソフトウェア工学を知らない.
  - クイックソートが書ければC言語を分かったつもり.
- 銀の弾丸を追いかけてるから.
  - 素朴に科学の勝利を信じてる. アホだと思う.
  - 抽象化が漏れることを知らない.
- ソフトウェア工学の最先端の研究がすぐに現場で使えると誤解しているから.
  - 導入コスト, 導入リスクなどを考えない.

「漏れのある抽象化の法則」

- 「自明でない抽象化はすべて, 程度の差こそあれ, 漏れがある. 」
  - Joel on Software, Joel Spolsky, ISBN 1590593898, 2004. http://www.joelonsoftware.com/Archive.html
  - うまく抽象化・モジュール化できないものはたくさんある.
  - cf. 数学は抽象化の塊.
- やさしい例:スタック
  - 仕様が小さい. 外部仕様と内部仕様がきれいに分離.
- 難しい例:分散ファイルシステム
  - キャッシュがUNIXセマンティクスを破壊.

まとめ

- プログラミングに
  数学的思考はあまり寄与しない.
- 数学的センス≠数学的思考



プログラマに
必要な資質

せいぜいこのぐらい

二極論はやめましょうね

関連しそうな記事

- Computational Thinking by Jeannette M. Wing, CACM, 49[3], 2006.
- Is Abstraction the Key to Computing? by Jeff Kramer, CACM, 50[4], 2007.

SEA Forum

プログラミングにとって数学的思考は役に立つのか？

2007年9月14日

玉井 哲雄
（東京大学大学院総合文化研究科）

数学的思考が役立つのは自明？

- 昔，Ershovが言った(CACM, July 1972)
  プログラマに必要な資質は
  - 第一級の数学者の論理性
  - エジソンのような工学的才能
  - 銀行員の正確さ
  - 探偵小説家の発想力
  - ビジネスマンの勤勉さ
  - 共同作業ができるような協調性

D. Knuthの場合

- ***The Art of Computer Programming*** の半分以上は数学の話
- TeXは数学の著述をきれいに印刷するために創造
- ***Mathematical Writing*** という本も書いている

他の工学でこの質問は意味があるか？

- 機械を作るのに数学的思考は役に立つか？
- 建物を建てるのに数学的思考は役に立つか？
- 「数学」は役に立つだろうが，数学的思考は？

質問を変えると

- プログラミングに論理(学)的思考は役に立つか？
- プログラミングに数理的思考は役に立つか？
- 数学が得意な人はプログラミングも得意か(あるいはその逆)？

数学に対する感情

- 数学への憧れ
- 数学コンプレックス
- 数学アレルギー
- 形式手法が極端な好きと嫌いの感情を呼び起こす原因か

より具体的に

■ Floydのアルゴリズム

全頂点対最短路アルゴリズム

```
for (m=0;m<n;m++)
  for (i=0;i<n;i++)
    for (j=0;j<n;j++)
      for (k=0;k<n;k++)
        x(i,j)=min(x(i,k)+x(k,j),x(i,j));
```

```
for (k=0;k<n;k++)
  for (i=0;i<n;i++)
    for (j=0;j<n;j++)
      x(i,j)=min(x(i,k)+x(k,j),x(i,j));
```

Floydのアルゴリズム(2)

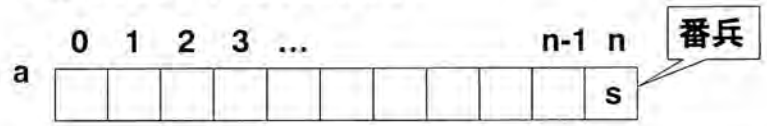
■ n個のものからランダムに重複なくm個選ぶ

```
S = Φ; //S を空集合に初期化
for (j=n-m+1; j<=n; j++) {
  t = randInt(1,j);
  if (! S.in(t)) S.insert(t)
  else S.insert(j);}
```

天才的，数学的

もうちょっとプログラム的な例

- 番兵: ループの終了条件とデータが尽きたという判定を1つにまとめる
  - 配列aの中のsを探す




```
i=0
while(a[i]!=s && i<n) i++
```

⇒

```
i=0
while(a[i]!=s) i++
```

巧み，プログラミング的

一方，数学の中にも計算的思考がある

- 多項式の簡約
  - 等式で書いているが，向きのある書換え
- 大学1年生がつまずく $\varepsilon - \delta$ 論法
  - 任意の ε に対して δ がとれるという手続き
- 逆にプログラミングで最初につまずくのが

  x = x+1

もっと日常的な例

- みずほ証券の誤発注問題
  - 東証のシステムに注文取消しができないという欠陥 → 裁判で係争中
- 数学的思考があれば防げたか？
  - 場合分けを論理的に尽くしていれば
  - しかし根本原因は拙劣なデータベース設計と開発体制

まとめ

- 数学的思考は役に立つ
- 数学的思考に似ているが性質の違うプログラミング的思考がある

編集後記

☆

当初の計画では，この号には昨年暮れに行われた Forum の報告を載せ，以降今年の１月以降の Forum Report を順次掲載して行こうというのが編集部の考えだったのですが，度々の催促にもかかわらず，なかなか原稿が集まりません．

☆☆

そこで，この夏以前の Forum は思い切ってあきらめ，９月以降の Forum の報告をまとめて行くことにしました．

☆☆☆

その第１弾として，９月 Forum 「プログラミングにとって数学的思考は役に立つのか？」の基調講演およびパネル討論の発表スライド１式をまとめました．

☆☆☆☆☆

コーディネータの野中さんが書いておられるように，当初は参加者がどれだけ集まるのかが不安だったのですが，ふたを開けてみれば予想を春かに越える数の方々に参加していただきました．今後もこのような SEA ならではの企画を試みて行きたいと考えています．

☆☆☆☆☆☆

なお，この Forum をきっかけとして，数学(特に「圏論」)についての勉強会を始めようという企画が進行中のようです．いずれは，そのグループを中心に数学分科会 (SIG-Mathematics) が立ち上がるかも知れません．

☆☆☆☆☆☆☆

新しい分科会といえば，もう１つ，今年の SS in 新潟 で「品詞保証」の討論グループに集まった方々が，SS 終了後も ML での議論を続けてきて，１１月にはオフラインのワークショップを開催し，近く分科会を正式に発足させようと考えておられるようです．

☆☆☆☆☆☆☆☆

こうした動きに興味を持たれた方は sea@sea.or.jp まで e-mail で御連絡ください．担当の幹事会メンバーに御紹介します．

☆☆☆☆☆☆☆☆☆

ソフトウェア技術者協会
〒160-0004 東京都新宿区四谷3-12 丸正ビル5F
Tel:03-3356-1077　Fax:03-3356-1072
E-mail:sea@sea.or.jp
URL:http://www.sea.jp/