

コンピュータ概論 A/B

-- VPS & Mathematica Programming (1) --

数学科 栗野 俊一 (TA: 浜津 翔 [院生 1 年])

2013/10/22 コンピュータ概

伝言

私語は慎むように !!

- 席は自由です : できるだけ前に詰めよう
- すぐやること
 - Note-PC の電源ケーブルを継ぎ、電源を入れ、今週の資料を読む
- 色々なお知らせについて
 - 栗野の Web Page に注意する事
<http://edu-gw2.math.cst.nihon-u.ac.jp/~kurino>
- 【注意】
 - winscp のインストールが未だの人は直ぐする(今日使う)
 - 講議の速度が早過ぎたら
 - ▷ 「遅くするように / もう一度説明するように」と申し出る
 - 今聞いた内容を、友達に聞きたければ
 - ▷ 「友達に確認したいので、時間が欲しい」と申し出る(勝手に話をしない)
 - 来週(10/28)は補習はありません(水野先生の補講があるため)

IP Address : 10.9.209.107

前回(2013/10/15)の復習

□ 講義

○ World Wide Web (WWW) の仕組みと HTML

- ▶ WWW は、HTML ファイルの集まり (HTML は Web Page を記述する言語)
- ▶ Web Page が相互に Link しあって Web (網) を作っている

○ Web Service の利用

- ▶ URL (Domain Name + Path Name) によって Web Page を指定できる
- ▶ Domain Name は DNS によって IP Address に変換
- ▶ IP Address で指定された Server から Internet 経由で Web Page を DL

○ HTML (超簡易) 入門

- ▶ 普通のテキストでもよい / タグで、文字列を囲むと、「属性」がつく
- ▶ XHTML5 形式の HTML ファイル (厳密な規則があるので、それを守って作る)

□ 実習

○ HTML ファイルの作成

本日(2013/10/22)の予定

□ 講義

- 「メタシステム」とは
- Mathematica によるプログラミング基礎

□ 実習

- [演習 1] 仮想マシンサーバの公開
- [演習 2] 友人の仮想マシンサーバの利用
- [演習 3] Mathematica の変数の利用法
- [演習 4] Mathematica の関数の作成方法
- [演習 5] 課題の作成

本日の課題 (2013/10/22)

□ 前回 (2013/10/15) の課題

- ▶ 表題 : 自分で作成した Web Page
- ▶ ファイル名 : 20131015-QQQQ.html (QQQQ は学生番号)
- ▶ 詳しくは、配布した sample-20131015.html の内容を参照

□ 今回 (2013/10/22) の課題

○ 次のファイルを提出しなさい

- ▶ 表題 : 1 から n までの 3 乗和を計算する関数 `cubeSum[n]` の作成
- ▶ ファイル名 : 20131022-QQQQ.nb (QQQQ は学生番号)
- ▶ 詳しくは、配布した sample-20131022.nb の内容を参照

実習 1: VM を外からアクセスできるようにする

□[実習 1-1] ネットワーク設定の変更

○ ネットワークの設定の変更 (NAT → ブリッジ)

▶ [player(P)]→[管理(M)]→[仮想マシン設定(S)]

▶ [ハードウェア]→[ネットワークアダプタ]

▶ [NAT ホストの IP アドレスを共有] を [ブリッジ:物理ネットワークに直接接続] に変更

○ VM を再起動し IP Address を確認

▶ VM 上で「ifconfig」を実行する (自分の VM の IP Address を記録)

□[実習 1-2] winscp によるファイルの転送

○ winscp を利用して、ファイルを自分の VM に転送する

▶ winscp を起動 : [新規] IP Address(自分 VM) / id:admin / pw:admin

▶ 先週の課題ファイルを VM の /www/web に転送

▶ URL [http://\[VM IP Address\]/20131015-QQQQ.html](http://[VM IP Address]/20131015-QQQQ.html) で参照

実習 2: 友人の VM の利用

□[実習 2-1] 友人の VM の利用

- skype で自分の VM の IP Address を交換

 - ▶ skype から友人の IP Address を調べ、参照してみる

- 友人の VM に自分のファイルを転送 (id:amdin / id:admin)

 - ▶ skype で URL を公開して、友人に参照してもらう(自分も友人の物を確認する)

- [注意] このままだと「危険」である事に注意

□[実習 2-2] root/admin のパスワードの変更

- 自分の VM で、「rw」としてパスワード変更を可能にする

- 「passwd admin」として、新しいパスワードを二度入れる

 - ▶ 画面上には入力したパスワードが表示されないので注意(弱いパスワードもだめ)

 - ▶ パスワードの変更後、新しいパスワードで入れるようになった事をチェック

 - ▶ これで、勝手に友人から変更される事がなくなった

- root のパスワードも変更 (「passwd root」と) する事

 - ▶ amdin, root のパスワードを忘れないように携帯電話に記録しておく

システム

□「システム」とは

○「対象」と「操作」の対 (この二つは完全に区別される)

▷「対象(Object)」: 興味の対象となる集合の要素(広い意味での「数」/「何」)

▷「操作(Operation)」: 「対象」の要素を「扱う」もの(広い意味での「関数」/「どうする」)

表 1: システムの例

システム	対象	操作	事例
計算 (電卓) 代幾 微積 数学 (入門)	数 ベクトル 関数 数学の対象 (集合)	演算 (四則/関数) 行列 (線形変換) 極限・微分・積分操作 論理 (集合演算)	$1 + 1 = 2$, $\sin\left(\frac{\pi}{3}\right) = \frac{\sqrt{3}}{2}$ $A\vec{v} = \vec{u}$ $\int f(x)dx = F(x) + C$ 「仮定」から「結論」
TeX Excel の計算 Mathematica OS アプリケーション 計算機	文章/文字列 セルの値 数式 (数を含む) ファイル データファイル (テキスト) データ	マクロ 式 Mathematica の関数 ファイル操作 プログラム (エディタ) プログラム	$\frac{1}{2}$ 他のセルの値から値を計算 $D[\text{Sin}[x], x] = \text{Cos}[x]$ ファイルのコピー/削除 ファイルの内容の変更 入力データから出力データ
料理 現実	食材 (素材の味) 世界	調理 (調味量) 人間 (愛)	食物 (味) から 食物 (味) 愛は世界を救う (アニメか!!)

□「学問」とは

○「対象」を処理するために「操作」の性質を学ぶ事

システム図

○ 個々の役割

- ▶ 興味の対象は Object
- ▶ 対象を操作するために Operation(方法)を利用
- ▶ Operation の仕組みを知る事が学問する事

○ 学習

- ▶ 数を操作するために計算方法を学ぶ
- ▶ 食事を作るためには料理法を学ぶ
- ▶ 情報を得るためには情報検索方法を学ぶ

○ 応用

- ▶ 学んだ内容結果を実地に適用する
- ▶ 学んだ操作方法を利用して、対象を操作する

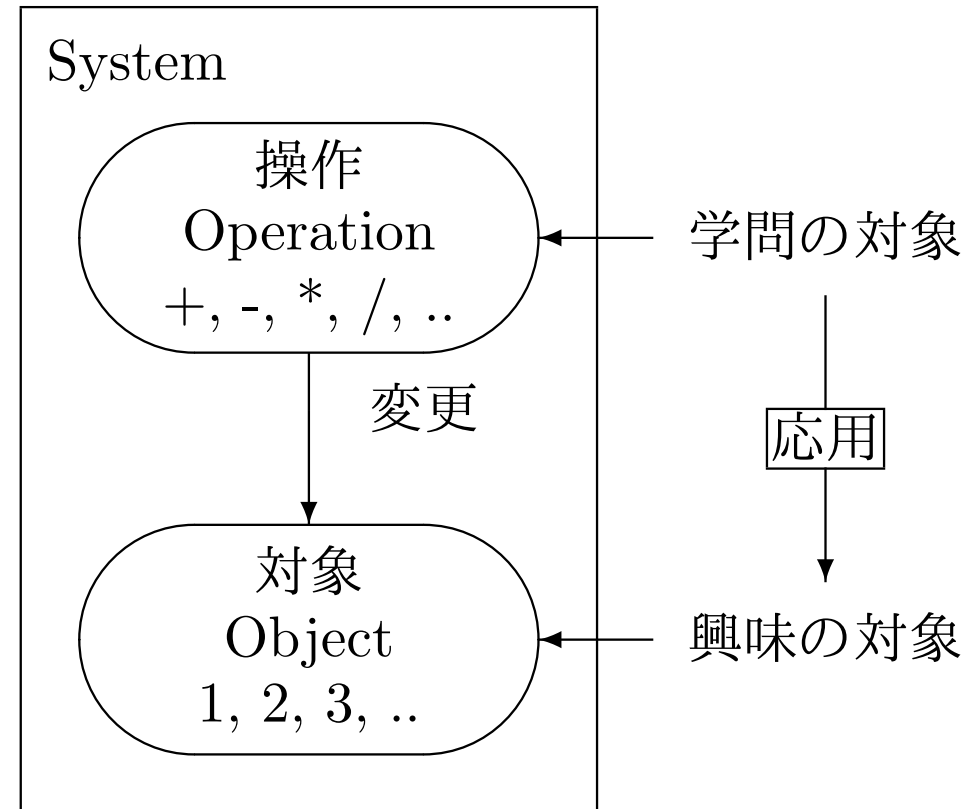


図 1: システム図

メタ概念

□メタ～(超～)

○「『システムの操作』を操作する」仕組み

- ▶「メタ○○」は、「○○の○○」と置き換える事ができる(cf. メタ情報)

表 2: メタシステムの例

システム	対象	操作	メタ論
計算 代幾 数学	数 ベクトル 数学の対象	演算 (四則/関数) 行列 (線形変換) 論理	汎関数/関数操作 (微積/関数解析) 行列の計算 (線形空間) 帰納法, 背理法 (証明論/ヒルベルトのプログラム)
TeX Mathematica 計算機	文章/文字列 数式 データ	マクロ Mathematica の関数 プログラム	マクロ定義 関数定義 (今日の課題) プログラミング
現実 (料理)	世界 (食物)	人物 (コック)	人物 (コック) の評判

○「対象を操作する操作方法」そのものを操作する

- ▶「対象の操作」を「操作方法の操作」で「間接的に操作」する
- ▶「間接」なので、「強力」だが「理解/応用するのは難しい」

○メタ論の例

- ▶あの議論(操作)の内容(対象)は解らないが、矛盾した議論(メタ)なので主張は正くない(背理法)
- ▶あの料理人(操作)の料理(対象)は食べた事はないが、三つ星のシェフ(メタ)なので、安心(評判)
- ▶あの政治家(操作)の政策(対象)は知らないが、顔が嫌い(メタ)だから票は入れない(感情)

▶平面幾何学の定理は、点と線を入れ替えても成立する(双対原理)

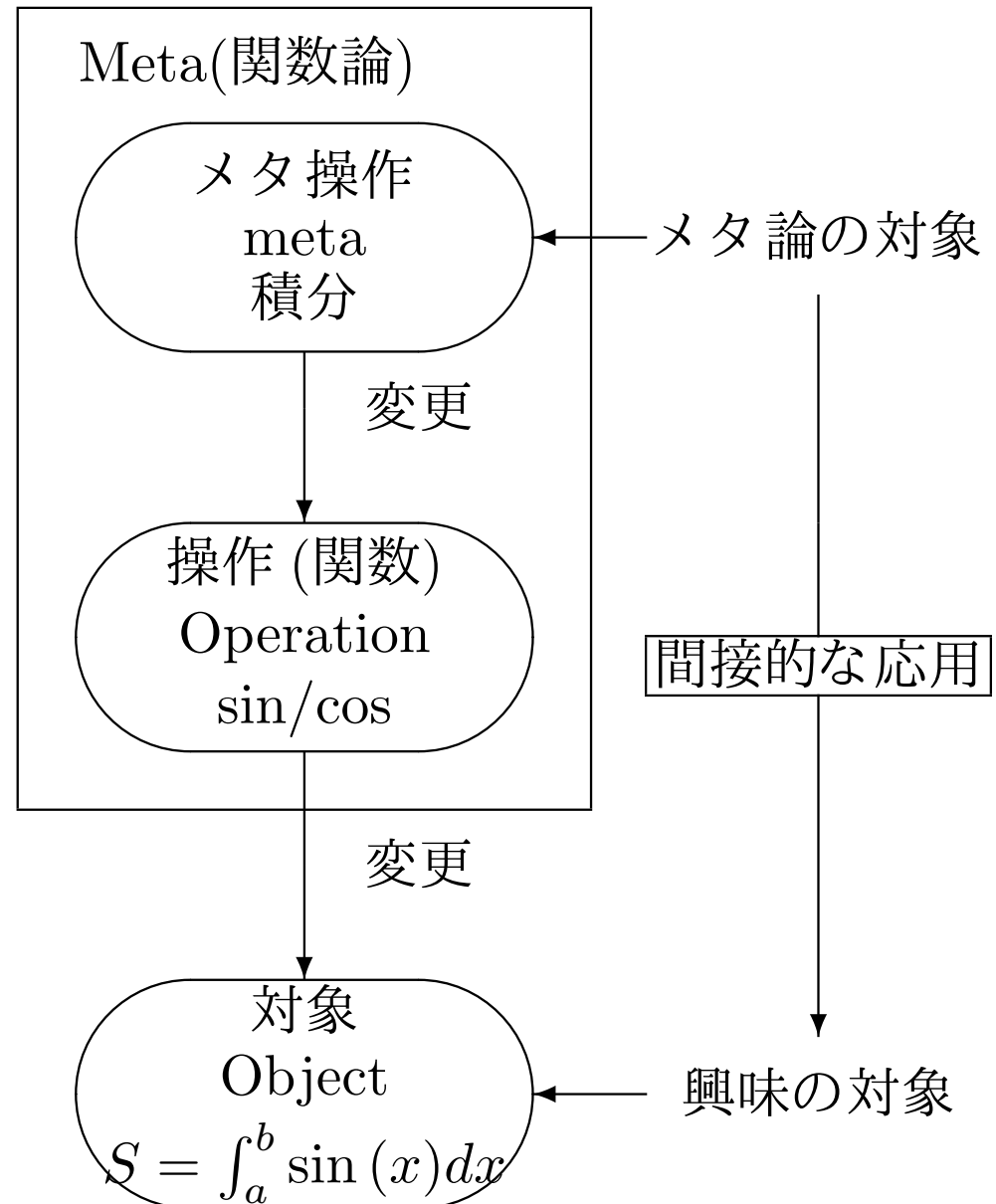
メタシステム図

○メタシステム

- ▶ 特定なシステム上に構成される
- ▶ システムの操作がメタシステムの対象
- ▶ 操作の操作により間接的に対象を操作

○関数論の例

- ▶ 対象:数 / 操作:関数 / メタ:微分積分
 - ◇ 関数 f と x 軸、直線 $x=a$, $x=b$ で囲まれた面積
 - ◇ 細分化して総和を計算してもよいが..
 - ◇ 定積分で計算可能
 - ◇ 原始関数を求める必要がある(メタ論の利用)



メタ関数と Mathematica プログラミング

□ メタ関数とは

- 関数を引数、あるいは結果とする関数 (汎函数とも言う)

 - ▶ 普通の関数 (数を引数にして結果を数とする) の例 : $\text{Sin}[x]$

 - ▶ メタ関数の例 : $D[\text{Sin}[x]] \rightarrow \text{Cos}[x]$ (引数も結果も関数)

□ Mathematica プログラミング

- メタ関数を定義する事

 - ▶ メタ関数には広義の意味で普通の関数を含めてしまう

□ Mathematica は数式「処理言語」システム

- 新しい関数が定義できる

 - ▶ 「定義のための表現形式」が含まれている (言語システム)

- 「言語」システムの定義には、「構文規則」と「意味規則」が必要

 - ▶ 構文規則 : どのような「(表現)形:文字列」が適切か

 - ▶ 意味規則 : その「(表現)形」はどんな「意味」に対応するか

□ Mathematica の関数定義とは

- 関数を表すシンボルに、その関数の意味を表す式を「割り当てる」事

 - ▶ 「 $\text{add}[x_,y_] := x+y$ 」とすれば、「 $\text{add}[2,3]$ が5になる」ように定義した事になる

構文(表現)と意味

□ 構文規則

- 「文字の並び」を「何の表現か?」と「判断するための規則」
 - ▶ 「 $123 + a$ 」は、「123」、「a」、「+」に分解される
 - ▶ 「123」は、「1」、「2」、「3」という「数字列」なので「数値」
 - ▶ 「a」は、「a」という「英文字から始まる文字列」なので「変数」
 - ▶ 「+」は、「足し算」を表す「演算子」
 - ▶ 「 $123 + a$ 」全体は、「式」になる
- 「どんな文字から構成されるか」という基準で区別される
 - ▶ 「何か」を表す、「表現方法」の規則
 - ▶ 規則に従って、「文字列」をみると、幾つかの「構文要素」に分解される

□ 意味規則

- 構文規則に対応した「意味」を決める規則
 - ▶ 「123」という「数字列」は、123 という「整数値」
 - ▶ 「a」という「シンボル」は、a という「変数」
 - ▶ 「 $123 + a$ 」は、「二つの数を加える」という「文字式」

□ 言語

- 構文規則 (文法)と、意味規則(「辞書」ならびに、「意味」) から成る
 - ▶ 「 $1 + 2$ 」は「構文的」には、「1」と「2」の「足し算」を表す「式」となる
 - ▶ 「 $1 + 2$ 」は「意味的」には、「3」(1, 2 は数値を表し、+ は足し算を表すから)

Mathematica(復習)

□ Mathematica とは

○ 数式処理言語システム

▶ 「数式」を計算したり、数式の計算を行うプログラムが作れる

○ 数式電卓

▶ (文字を含む)数式の計算を行う

▶ cf. 電卓: 「数」の計算が出来る(数の式を入れると数の計算を行う)

▶ [スタート]→[すべてのプログラム]→[アクセサリ]→[電卓]

□ Mathematica の使い方

○ ノートブックを開く

○ 式を入力して [Shift]+[Enter] (以下 [SE] と表現)で評価開始

▶ 計算に時間がかかりそうなら.. [Alt]+[,] で中断できる

○ この講義では「Mathematica の導入」のみを扱う

▶ 自分で色々調べて、試してみる (Help/チュートリアル)

Mathematica Notebook

□ Mathematica Notebook (*.nb)

- Mathematica の計算結果を保存する形式
- 結果の保存方法
 - ▶ [ファイル]→[別名で保存] : 好きな名前で保存できる
 - ▶ [ファイル]→[保存] (Ctrl-S) : 今の名前で内容を更新する(古い内容は失われる)
- 結果の利用方法
 - ▶ [ファイル]→[開く] (Ctrl-O) : 以前に保存した内容を読み込む

□ 電卓としての利用

- 「式」を入れて [Shift]+[Enter] (以下、[SE]) すると計算
 - ▶ 入力した式は In[番号] の形で表示される
 - ▶ 計算した結果は Out[番号] の形で表示される
 - ▶ 番号は、[SE] の順番につけられる
- [SE] の効果
 - ▶ 「式」の評価が行われる
 - ▶ 「番号」が増える
 - ▶ In[番号]/Out[番号]が、それぞれ「定義」される

変数とその値

□ シンボル (構文的な要素)

○ シンボルとは

- ▶ 「英字」から始まり「英数字」からなる文字列の事 (a , abc , $a1$, $a12z$)
- ▶ (注意) 「英字」には、「ギリシャ文字」も含まれる (π , α)
- ▶ cf. 「数字」から始まる文字列はシンボルではない ($2a$ は $2 \times a$ となる)

○ Mathematica は、大文字で始まるシンボルの幾つかを利用

- ▶ (注意) 自分が利用する場合は、小文字で始まるシンボルを利用する事

□ 変数 (意味的な要素)

○ 変数とは

- ▶ 名前として「シンボル」をもち、「値を持つ事ができるもの」
- ▶ 「変数名」は、「変数」の名前で、それは「シンボル」である

○ 変数の(即)値とは

- ▶ (現時点で)変数に対応している「式(の評価結果)」の事
- ▶ 「環境」は、「変数名」と「変数の値」の対応表をもっている

○ 変数の値の参照

- ▶ 「シンボル」の形を示す物を入れると、「その値」が評価され、その結果が表示される
- ▶ 「値」が設定されていない場合は、その「変数自身」が「変数の値」となる
- ▶ ?変数名で、「変数の値」である「式」を見る事ができる

変数への代入と定義

□ 代入と定義

- 共に変数に値を「割り当て」る
- 代入：「変数 = 式」
 - ▶ 「式」は「即時評価」される。その評価結果が「変数の値」となる
- 定義：「変数 := 式」
 - ▶ 「式」は「遅延評価」される。その「式」そのものが「変数の値」となる
- 変数の値は何度でも変更できる

□ 環境

- 変数名とその値の対応表をもっている
 - ▶ 代入も定義も、その対応表を書換えている
 - ▶ 書き換える値が違うだけ

□ 「式」の評価(変数の場合)

- 式の中に変数名が含まれていた場合に、それを変数の値に書き換える

関数の利用

□ Mathematica の関数

○ Mathematica の関数とは

▶ 「シンボル[引数列]」の形で表現され、値を持つ事ができる「もの」

▶ cf. `Sin[Pi] / f[3] / g[x,y^4]`

○ 関数の評価

▶ 「シンボル[引数列]」の形を、その値に置き換える

□ 引数のパターンマッチ

○ 「`_`」(アンダースコア)を利用して「式」の「抽象パターン」が表現できる

○ 例1

▶ `next[0] := 1`

▶ `next[1] := 2`

▶ `next[_] := 0`

○ 例2

▶ `fib[1]=1`

▶ `fib[2]=1`

▶ `fib[x_]:=fib[x-1]+fib[x-2]`

自然数

□ ペアノの公理

○ 自然数を定義する公理

- ▷ 0 は自然数である
- ▷ n が自然数なら $n+1$ も自然数である
- ▷ 上記の二つ以外に自然数はない

□ Mathematica でペアノの公理の形の「自然数」を考える

○ 自然数 (これを「ペアノ形式の表現」と呼ぶ事にする)

- ▷ 0 は 0 で表現
- ▷ $n + 1$ は $s[n]$ で表現

○ 自然数の和

- ▷ $\text{padd}[0, x_]$:= x
- ▷ $\text{padd}[s[x_], y_]$:= $s[\text{padd}[x, y]]$

○ これを繰り返すと、「分数(有理数)」まで表現できる

- ▷ 「実数」を表現するには、「収束概念」が必要になる

自然数による数の表現 (nat.txt)

□ 自然数

○ 0 と +1 (サクセッサ) のみで作る

▷ $3 = 0+1+1+1 = s[s[s[0]]]$

□ 整数

○ 自然数の二つ組 (m,n) で整数 z を表現する

▷ $z = m - n \Rightarrow pp[m,n]$

▷ 同じ整数に対する異なる (m,n) 組があるので、同値類(\sim)を作る

▷ 例 $pp[4,2] \sim pp[3,1] \sim pp[2,0]$

□ 有理数

○ 自然数と整数の組 (z,n) で整数 q を表現する

▷ $q = z/n \Rightarrow qq[z,n]$

▷ 同じ有理数に対する異なる (z,n) 組があるので、同値類(\sim)を作る

▷ 例 $qq[18,12] \sim pp[9,4] \sim pp[6,2]$

「ペアノ形式」と普通の表現の関係

表 3: 「S 表現」と普通の表現の関係

概念	普通の表現 (例)	ペアノ形式 (例)	コメント
0	0	0	0 は自然数
自然数 (0 以外)	1,2,3,..	s[0], s[s[0]], s[s[s[0]]], ..	x が自然数なら s[x] も自然数
n の次	n + 1	s[n]	s はサクセッサ (後継) 関数
和	1 + 2	padd[s[0],s[s[0]]]	
差	1 - 2	psub[s[0],s[s[0]]]	引く数の方が大きい場合は 0 にな
積	1 * 2	pmul[s[0],s[s[0]]]	
商	1 / 2	pdiv[s[0],s[s[0]]]	整数割り算 (小数点以下は切り捨て
最大公約数	gcd(1,2)	pgcd[s[0],s[s[0]]]	
整数	1, -1	pp[s[0],0], s[0,s[0]]	自然数の対を同値類で割った物が
整数の正規化		zbar	ペアのどちらか一方を 0 にする
整数の四則	+, -, *, /	zadd,zsub,zmul,zdiv	
有理数	1/2	qq[pp[s[0],0],s[s[0]]]	分子は整数で、分母が自然数の対
有理数の正規化		qbar	約分する
有理数の四則	+, -, *, /	qadd,qsub,qmul,qdiv	

□ 可換性(well defined)

- 通常 of 自然数とペアノ形式 of 自然数は ntop, pton で同型になっている
- 個々のペアノ形式 of 関数は ntop, pton に関して可換になっている

▷ pton[padd[ntop[1],ntop[2]]]=3=1+2