ソフトウェア基礎論配布資料 (2)
算術式の言語

五十嵐 淳
京都大学 大学院情報学研究科知能情報学専攻
e-mail: igarashi@kuis.kyoto-u.ac.jp

平成 16 年 10 月 12 日

1 算術式の文法定義

自然数上の加算・乗算式の集合を算術式の集合 **Aexp** を定義する．構成要素は最小の自然数 `0`,「次の数」を示す `S`, 加算・乗算の `+`, `*` である．算術式は無限に存在するので，集合を厳密に定義するためには，要素を列挙するわけにはいかない．以下ではそのような集合を定義する方法をいくつか見る．

1.1 BNF 記法による定義

最初は **BNF 記法**(*Backus-Naur form*) による定義である．

1.1.1 Definition: 算術式 (メタ変数 a) の集合 **Aexp** は以下の文法で定義する．

$$a \in \mathbf{Aexp} ::= \mathtt{0} \mid \mathtt{S(}a\mathtt{)} \mid a\ \mathtt{+}\ a \mid a\ \mathtt{*}\ a$$

□

演算子の優先順位と抽象構文木　上の文法から，例えば `S(0)` や `S(0) + 0` は **Aexp** の要素である．これらを組み合わせて，`S(0) + 0 * S(0)` も算術式である．しかし，`S(0)` と `0 * S(0)` を `+` で組み合わせても，(文字列として) 見た目が同じ算術式が得られる．別の言い方をすると，上の文法は，文字列からそれに対応する算術式が一意に得られないので曖昧である．より厳密には，上の文法は算術式を木として定義していると考えるのが正しい．このような木構造は言語処理系で**抽象構文木**(*abstract syntax tree*) と呼ばれるものである．また，このような (文字列の解析のためではない) 文法を**抽象構文**(*abstract syntax*) と呼ぶ．

もちろん，上の例のような「生成の仕方」の違いを区別する必要はでてくるので適宜 (メタな記号である) “(” “)” を使用し，`(S(0) + 0) * S(0)` などと記述する．以下では，木構造として「等しい」ことを示すために $\equiv$ という記号を用いる．また，記号の優先度を指定して括弧を省略していく．ここでは，メタ言語の常識に合わせて `*` は `+` よりも強く結合し，`+`, `*` ともに左結合するとする．よって，

$$\mathtt{S(0) + 0 * S(0)} \equiv \mathtt{S(0) + (0 * S(0))} \not\equiv \mathtt{(S(0) + 0) * S(0)}$$
$$\mathtt{0 + 0 + 0} \equiv \mathtt{(0 + 0) + 0}$$
$$\mathtt{0 * 0 * 0} \equiv \mathtt{(0 * 0) * 0}$$

である．

1.2 帰納的な定義

上の BNF 記法は，より厳密には以下のような**帰納的定義**(*inductive definition*) の簡潔な記法と考えられる．

1.2.1 定義: 算術式の集合 **Aexp** は，以下の条件を満たす集合 A のうち**最小**のものである．

1. $\{\texttt{0}\} \subseteq A$
2. もし $a \in A$ ならば $\{\texttt{S(}a\texttt{)}\} \subseteq A$
3. もし $a_1 \in A$ かつ $a_2 \in A$ ならば $\{a_1 \texttt{ + } a_2, a_1 \texttt{ * } a_2\} \subseteq A$

□

「最小の」というのは，上の条件を満たす集合はいくらでも考えられるので，ゴミが入っていないことを保証するための条件である．例えば，上の条件を満たす集合があったときに，$\texttt{1}, \texttt{S(1)}, \texttt{1+1}, \ldots$ などを加えたような集合も，同じ条件を満たすので，最小性を言うことは非常に大事である．BNF 記法での定義ではわざわざ言わないことが多い．

また，上の「定義」は **Aexp** の満たすべき条件のみが記述されており，その存在は自明ではない[1]が，このような集合が存在することも，証明することができる．

1.3 規則による定義

帰納的な定義は，以下のように**規則**(*rule*) を使って記述されることもある．

1.3.1 定義: 算術式の集合 **Aexp** は，以下の規則で定義される．

$$\frac{}{\texttt{0} \in \textbf{Aexp}} \quad \text{(A-Zero)} \qquad \frac{a_1 \in \textbf{Aexp} \qquad a_2 \in \textbf{Aexp}}{a_1 \texttt{ + } a_2 \in \textbf{Aexp}} \quad \text{(A-Plus)}$$

$$\frac{a \in \textbf{Aexp}}{\texttt{S(}a\texttt{)} \in \textbf{Aexp}} \quad \text{(A-Succ)} \qquad \frac{a_1 \in \textbf{Aexp} \qquad a_2 \in \textbf{Aexp}}{a_1 \texttt{ * } a_2 \in \textbf{Aexp}} \quad \text{(A-Mult)}$$

□

規則による定義でも，BNF 記法による定義と同様，最小性は暗黙の仮定とされることが多い．また，規則を用いて何かが **Aexp** の要素であることを，**導出木**(*derivation tree*) というものを使って記述することがある．以下は $\texttt{0 + S(0)} \in \textbf{Aexp}$ の導出木である．

$$\dfrac{\dfrac{}{\texttt{0} \in \textbf{Aexp}}\ \text{A-Zero} \quad \dfrac{\dfrac{}{\texttt{0} \in \textbf{Aexp}}\ \text{A-Zero}}{\texttt{S(0)} \in \textbf{Aexp}}\ \text{A-Succ}}{\texttt{0 + S(0)} \in \textbf{Aexp}}\ \text{A-Plus}$$

[1] このような定義を超越的な定義と呼ぶ

1.4 帰納法による証明・帰納法による関数定義

自然数 (を表現する算術式) の帰納的な定義

$$n \in \mathbf{Nv} ::= \mathtt{0} \mid \mathtt{S}(n)$$

からは以下の数学的帰納法

$$(P(0)\ \&\ \forall n \in \mathbf{Nv}.(P(n) \Rightarrow P(n+1))) \Rightarrow \forall n \in \mathbf{Nv}.P(x)$$

が導かれる．これと同様に帰納的に集合を定義すると，それに対応した**帰納法の証明原理**(*induction principle*) が導かれる．

算術式の構造帰納法　a が算術式ということは，その構成の仕方より以下のどれかが成立する．

1. $a \equiv \mathtt{0}$.
2. $a \equiv \mathtt{S}(a_0)$ であり，a_0 は a より「小さい」算術式.
3. $a \equiv a_1\ \mathtt{+}\ a_2$ であり，a_1, a_2 は a より「小さい」算術式.
4. $a \equiv a_1\ \mathtt{*}\ a_2$ であり，a_1, a_2 は a より「小さい」算術式.

これより算術式の**構造に関する帰納法**(*structural induction*) の原理は以下のようになる．

1.4.1 定理 [算術式の構造に関する帰納法]: $\forall a \in \mathbf{Aexp}.P(a)$ を示すには,

1. $P(\mathtt{0})$
2. $\forall a \in \mathbf{Aexp}.P(a) \Rightarrow P(\mathtt{S}(a))$
3. $\forall a_1, a_2 \in \mathbf{Aexp}.P(a_1)\ \&\ P(a_2) \Rightarrow P(a_1\ \mathtt{+}\ a_2)$
4. $\forall a_1, a_2 \in \mathbf{Aexp}.P(a_1)\ \&\ P(a_2) \Rightarrow P(a_1\ \mathtt{*}\ a_2)$

を示せばよい. □

また，帰納法の原理から,「帰納法による関数定義」が可能になる．例えば，算術式の大きさを示す関数 $size \in \mathbf{Aexp} \to \mathbf{Nat}$ は以下の 4 行だけで定義したことになる (以下の 4 式を満たすような関係はただひとつしかないということが帰納法の原理から証明できる).

$$\begin{aligned}
size(\mathtt{0}) &= 1 \\
size(\mathtt{S}(a_0)) &= size(a_0) + 1 \\
size(a_1\ \mathtt{+}\ a_2) &= size(a_1) + size(a_2) + 1 \\
size(a_1\ \mathtt{*}\ a_2) &= size(a_1) + size(a_2) + 1
\end{aligned}$$

1.4.2 例: $depth \in \mathbf{Aexp} \to \mathbf{Nat}$ を以下のように定義する.

$$\begin{aligned} depth(\mathtt{0}) &= 1 \\ depth(\mathtt{S(}a_0\mathtt{)}) &= depth(a_0) + 1 \\ depth(a_1 \mathtt{\ +\ } a_2) &= \max(depth(a_1), depth(a_2)) + 1 \\ depth(a_1 \mathtt{\ *\ } a_2) &= \max(depth(a_1), depth(a_2)) + 1 \end{aligned}$$

このとき, $\forall a \in \mathbf{Aexp}.size(a) \leq 2^{depth(a)} - 1$ である.

Proof: 算術式の構造に関する帰納法で証明する.

1. $size(\mathtt{0}) \leq 2^{depth(\mathtt{0})} - 1$ は各関数の定義より明らか.
2. $size(a) \leq 2^{depth(a)} - 1$ を仮定して, $size(\mathtt{S(}a\mathtt{)}) \leq 2^{depth(\mathtt{S(}a\mathtt{)})} - 1$ を示す.

$$\begin{aligned} size\mathtt{S(}a\mathtt{)} &= size(a) + 1 \\ &\leq 2^{depth(a)} \leq 2^{depth(a)+1} - 1 \\ &\leq 2^{depth(\mathtt{S(}a\mathtt{)})} - 1 \end{aligned}$$

3. $size(a_1) \leq 2^{depth(a_1)}-1$ と $size(a_2) \leq 2^{depth(a_2)}-1$ を仮定して, $size(a_1 \mathtt{\ +\ } a_2) \leq 2^{depth(a_1 \mathtt{\ +\ } a_2)}-1$ を示す.

$$\begin{aligned} size(a_1 \mathtt{\ +\ } a_2) &= size(a_1) + size(a_2) + 1 \\ &\leq 2^{depth(a_1)} - 1 + 2^{depth(a_2)} - 1 + 1 \\ &\leq 2^{\max(depth(a_1),depth(a_2))} + 2^{\max(depth(a_1),depth(a_2))} - 1 \\ &\leq 2^{depth(a_1 \mathtt{\ +\ } a_2)} - 1 \end{aligned}$$

4. $size(a_1) \leq 2^{depth(a_1)}-1$ と $size(a_2) \leq 2^{depth(a_2)}-1$ を仮定して, $size(a_1 \mathtt{\ *\ } a_2) \leq 2^{depth(a_1 \mathtt{\ *\ } a_2)}-1$ を示す. (省略)

□

2 算術式の評価と性質

2.1 評価関係の定義

評価関係 $\longrightarrow$ ($\subseteq \mathbf{Aexp} \times \mathbf{Aexp}$) も集合であるので, 算術式と同様に (規則を使って) 帰納的に定義する. 以降, $(a_1, a_2) \in \longrightarrow$ を $a_1 \longrightarrow a_2$ と書く.

2.1.1 定義: 関係 $\longrightarrow$ は, 図 1 の規則で定義される.

また, 別の評価関係 $\longrightarrow_v$ も別の規則で定義する.

2.1.2 定義: 関係 $\longrightarrow_v$ は, 図 2 の規則で定義される.

$$\frac{}{a_0 \texttt{ + } \texttt{0} \longrightarrow a_0} \quad \text{(E-PlusZero)}$$

$$\frac{}{a_1 \texttt{ + } \texttt{S(}a_2\texttt{)} \longrightarrow \texttt{S(}a_1 \texttt{ + } a_2\texttt{)}} \quad \text{(E-PlusSucc)}$$

$$\frac{}{a_0 \texttt{ * } \texttt{0} \longrightarrow \texttt{0}} \quad \text{(E-MultZero)}$$

$$\frac{}{a_1 \texttt{ * } \texttt{S(}a_2\texttt{)} \longrightarrow a_1 \texttt{ * } a_2 \texttt{ + } a_1} \quad \text{(E-MultSucc)}$$

$$\frac{a \longrightarrow a'}{\texttt{S(}a\texttt{)} \longrightarrow \texttt{S(}a'\texttt{)}} \quad \text{(E-Succ)}$$

$$\frac{a_1 \longrightarrow a_1'}{a_1 \texttt{ + } a_2 \longrightarrow a_1' \texttt{ + } a_2} \quad \text{(E-PlusL)}$$

$$\frac{a_2 \longrightarrow a_2'}{a_1 \texttt{ + } a_2 \longrightarrow a_1 \texttt{ + } a_2'} \quad \text{(E-PlusR)}$$

$$\frac{a_1 \longrightarrow a_1'}{a_1 \texttt{ * } a_2 \longrightarrow a_1' \texttt{ * } a_2} \quad \text{(E-MultL)}$$

$$\frac{a_2 \longrightarrow a_2'}{a_1 \texttt{ * } a_2 \longrightarrow a_1 \texttt{ * } a_2'} \quad \text{(E-MultR)}$$

図 1: 評価規則 (1)

$$\frac{}{n_0 \texttt{ + } \texttt{0} \longrightarrow_v n_0} \quad \text{(Ev-PlusZero)}$$

$$\frac{}{n_1 \texttt{ + } \texttt{S(}n_2\texttt{)} \longrightarrow_v \texttt{S(}n_1 \texttt{ + } n_2\texttt{)}} \quad \text{(Ev-PlusSucc)}$$

$$\frac{}{n_0 \texttt{ * } \texttt{0} \longrightarrow_v \texttt{0}} \quad \text{(Ev-MultZero)}$$

$$\frac{}{n_1 \texttt{ * } \texttt{S(}n_2\texttt{)} \longrightarrow_v n_1 \texttt{ * } n_2 \texttt{ + } n_1} \quad \text{(Ev-MultSucc)}$$

$$\frac{a \longrightarrow_v a'}{\texttt{S(}a\texttt{)} \longrightarrow_v \texttt{S(}a'\texttt{)}} \quad \text{(Ev-Succ)}$$

$$\frac{a_1 \longrightarrow_v a_1'}{a_1 \texttt{ + } a_2 \longrightarrow_v a_1' \texttt{ + } a_2} \quad \text{(Ev-PlusL)}$$

$$\frac{a_2 \longrightarrow_v a_2'}{n_1 \texttt{ + } a_2 \longrightarrow_v n_1 \texttt{ + } a_2'} \quad \text{(Ev-PlusR)}$$

$$\frac{a_1 \longrightarrow_v a_1'}{a_1 \texttt{ * } a_2 \longrightarrow_v a_1' \texttt{ * } a_2} \quad \text{(Ev-MultL)}$$

$$\frac{a_2 \longrightarrow_v a_2'}{n_1 \texttt{ * } a_2 \longrightarrow_v n_1 \texttt{ * } a_2'} \quad \text{(Ev-MultR)}$$

図 2: 評価規則 (2)

ここで，規則のメタ変数を置き換える時には，メタ変数の名前に応じて具体化を行なうことが暗黙のうちに仮定されている．例えば，E-PLUSRV を使用するときには，n_1 は上で定義した「自然数」(S(S(...(0)...))) で具体化しなければならない．

2.1.3 例:

$$\texttt{(S(S(0)) + 0) * S(0+S(0))} \longrightarrow \texttt{S(S(0)) * S(0+S(0))}$$
$$\texttt{(S(S(0)) + 0) * S(0+S(0))} \longrightarrow \texttt{(S(S(0)) + 0) * S(S(0+0))}$$
$$\texttt{(S(S(0)) + 0) * S(0+S(0))} \longrightarrow \texttt{(S(S(0)) + 0) * (0+S(0)) + S(0+S(0))}$$
$$\texttt{(S(S(0)) + 0) * S(0+S(0))} \longrightarrow_v \texttt{S(S(0)) * S(0+S(0))}$$
$$\texttt{(S(S(0)) + 0) * S(0+S(0))} \not\longrightarrow_v \texttt{(S(S(0)) + 0) * S(S(0+0))}$$
$$\texttt{(S(S(0)) + 0) * S(0+S(0))} \not\longrightarrow_v \texttt{(S(S(0)) + 0) * (0+S(0)) + S(0+S(0))}$$

上の関係は帰納的に定義されたので，やはり，それに対応する帰納法の証明原理が導かれる．

2.1.4 定理 [評価関係の導出に関する帰納法]: $\forall a_1, a_2 \in \mathbf{Aexp}.a_1 \longrightarrow a_2 \Rightarrow P(a_1, a_2)$ を示すには，以下を示せばよい．

1. $\forall a \in \mathbf{Aexp}.P(a_0\ \texttt{+}\ \texttt{0}, a_0)$
2. $\forall a_1, a_2 \in \mathbf{Aexp}.P(a_1\ \texttt{+}\ \texttt{S(}a_2\texttt{)}, \texttt{S(}a_1\ \texttt{+}\ a_2\texttt{)})$
3. $\forall a \in \mathbf{Aexp}.P(a_0\ \texttt{*}\ \texttt{0}, \texttt{0})$
4. $\forall a_1, a_2 \in \mathbf{Aexp}.P(a_1\ \texttt{*}\ \texttt{S(}a_2\texttt{)}, a_1\ \texttt{*}\ a_2\ \texttt{+}\ a_1)$
5. $\forall a, a' \in \mathbf{Aexp}.P(a, a') \Rightarrow P(\texttt{S(}a\texttt{)}, \texttt{S(}a'\texttt{)})$
6. $\forall a_1, a_1', a_2 \in \mathbf{Aexp}.P(a_1, a_1') \Rightarrow P(a_1\ \texttt{+}\ a_2, a_1'\ \texttt{+}\ a_2)$
7. $\forall a_1, a_2, a_2' \in \mathbf{Aexp}.P(a_2, a_2') \Rightarrow P(a_1\ \texttt{+}\ a_2, a_1\ \texttt{+}\ a_2')$
8. $\forall a_1, a_1', a_2 \in \mathbf{Aexp}.P(a_1, a_1') \Rightarrow P(a_1\ \texttt{*}\ a_2, a_1'\ \texttt{*}\ a_2)$
9. $\forall a_1, a_2, a_2' \in \mathbf{Aexp}.P(a_2, a_2') \Rightarrow P(a_1\ \texttt{*}\ a_2, a_1\ \texttt{*}\ a_2')$

□

2.2 評価に関する性質

上の二つの関係については，以下のような性質が成立する．

2.2.1 定理 [評価の決定性]: $\longrightarrow_v \in \mathbf{Aexp} \rightharpoonup \mathbf{Aexp}$ である．すなわち，$\forall a, a', a'' \in \mathbf{Aexp}.a \longrightarrow_v a' \ \&\ a \longrightarrow_v a'' \Rightarrow a' \equiv a''$.

Proof: $a \longrightarrow_v a'$ の導出に関する帰納法で証明する．$P(a, a')$ は $\forall a'' \in \mathbf{Aexp}.a \longrightarrow_v a'' \Rightarrow a' \equiv a''$ である． □

2.2.2 定義: $a \in \mathbf{Aexp}$ が，$\neg\exists a' \in \mathbf{Aexp}.a \longrightarrow a'$ の時，a を**正規形**(*normal form*) である，という．

2.2.3 定義: $a \longrightarrow^* a'$ を $\longrightarrow$ の反射的推移的閉包とする．つまり，$a \longrightarrow^* a' \iff a \underbrace{\longrightarrow \cdots \longrightarrow}_{n} a'$ (ただし $n \geq 0$) である．$\longrightarrow_v^*$ も同様に 定義する．

2.2.4 定理 [合流性, confluence]: $a_1 \longrightarrow^* a_2 \,\&\, a_1 \longrightarrow^* a_3 \Rightarrow \exists a_4.a_2 \longrightarrow^* a_4 \,\&\, a_3 \longrightarrow^* a_4$.

Proof: a_1 の構造に関する帰納法で証明する． □

2.2.5 系 [正規形の唯一性, uniqueness of normal forms]: $a \longrightarrow^* a'$ かつ $a \longrightarrow^* a''$ かつ a', a'' が正規形ならば，$a' \equiv a''$ である．

2.2.6 定理 [評価の停止性, termination of evaluation]: 任意の算術式 a に対し，$a \longrightarrow^* a'$ なる正規形の a' が存在する．

Proof: $a \longrightarrow \cdots \longrightarrow a' \longrightarrow \cdots$ なる無限列がないことを示す．まず，$w(a) \in \mathbf{Aexp} \to \mathbf{Nat}$ を以下のように定義する．

1. $w(\mathtt{0}) = 1$
2. $w(\mathtt{S(}a\mathtt{)}) = w(a) + 1$
3. $w(a_1 \ \mathtt{+}\ a_2) = w(a_1) + w(a_2) + 1$
4. $w(a_1 \ \mathtt{*}\ a_2) = w(a_1) \cdots w(a_2) + 1$

このとき，$\forall a, a' \in \mathbf{Aexp}.a \longrightarrow a' \Rightarrow w(a) > w(a')$ である．また $\forall a \in \mathbf{Aexp}.w(a) > 0$ なので，$a \longrightarrow \cdots \longrightarrow a' \longrightarrow \cdots$ が成立したとすると，$w(a) > \cdots > w(a') > \cdots$ なる無限列が存在することになり矛盾． □