知能ソフトウェア特論
Intelligent Software

項書換え系（３）
合流性

Term Rewriting Systems(3)
Confluence

1．抽象書換え系 と合流性(1/3)

(Abstract reduction systems and confluence)

■定義

抽象書換え系 (A,R)

集合 A と二項関係 $R \subseteq A \times A$ の対.
今後は R を $\rightarrow$ と書く.
$(a,b) \in \rightarrow$ のとき，$a \rightarrow b$ と書く.

a は b に簡約可能

■Definition

Abstract reduction system (A,R)
consists of a set A and a binary relation $R \subseteq A \times A$.
We will write $\rightarrow$ for R.
When $(a,b) \in \rightarrow$, we write $a \rightarrow b$ and say that a is reducible to b.

【Example】

$$A = \{a, b, c, d, e\}$$

$$R = \rightarrow = \{(a,b),(b,a),(a,c),(b,d),(e,d)\}$$

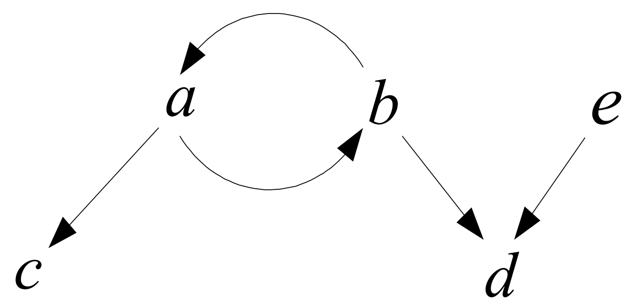


図1

1．抽象書換え系 と合流性(2/3)

■定義

反射推移閉包　$a \to^* b$

a から b へ 0 ステップ以上の
$\to$ で書換え可能

推移閉包　$a \to^+ b$

a から b へ 1 ステップ以上の
$\to$ で書換え可能

会同性　$a \downarrow b$

$a \to^* c$ かつ $b \to^* c$ を満たす
$c \in A$ が存在する

■Definition

Reflexive transitive closure $\to^*$:
We write $a \to^* b$, if b can be obtained from a by zero or more steps of reduction by $\to$.

Transitive closure $\to^+$:
We write $a \to^+ b$, if b can be obtained from a by one or more steps of reduction by $\to$.

Joinability $\downarrow$:
a and b are joinable ($a \downarrow b$), if there exists c in A such that $a \to^* c$ and $b \to^* c$.

【Example】

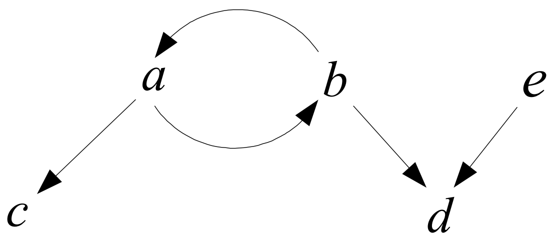


図1

$a \to^* d$,　$a \to^* a$

$a \to^+ d$

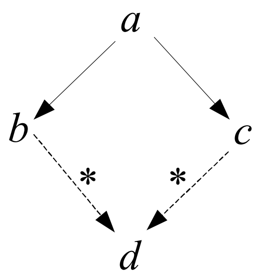
$a \downarrow e$,　$a \downarrow b$

1．抽象書換え系 と合流性(3/3)

■定義

弱合流性

任意の $a,b,c \in A$ に対して，
$a \to b$ かつ $a \to c$ ならば $b \downarrow c$



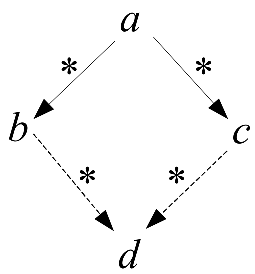
弱合流性

■Definition

Weak confluence:
$(A, \to)$ is weakly confluent, if
for all a, b and c in A,
$a \to b$ and $a \to c$ implies $b \downarrow c$.

合流性

任意の $a,b,c \in A$ に対して，
$a \to^* b$ かつ $a \to^* c$ ならば $b \downarrow c$



合流性

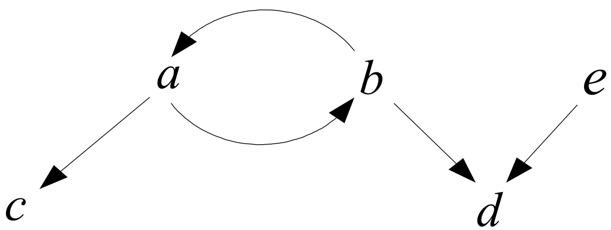
Confluence:
$(A, \to)$ is confluent,
if for all a, b and c in A,
$a \to^* b$ and $a \to^* c$ implies $b \downarrow c$.

合流性をもつならば，弱合流性をもつ．
その逆は成り立たない．

Confluence implies weak confluence, but its converse does not hold as the example in this figure shows.

【Example】

弱合流性をもつが
合流性をもたない例
(cとdがjoinしない)



2．合流性の基本性質 (1/3)

(Basic properties of confluence)

■定義

$a \in A$ は正規形である：

$a \to b$ なる $b \in A$ が存在しない

$a \in A$ は正規形をもつ：

$a \to^* b$ なる正規形 $b \in A$ が存在する

$(A, \to)$ は一意の正規形をもつ：

任意の $a \in A$ に対し，
a が高々１つの正規形をもつ

■Definition

$a \in A$ is a normal form,
if there exists no $b \in A$ such that $a \to b$.

$a \in A$ has a normal form,
if there exists a normal form $b \in A$ such that $a \to^* b$.

$(A, \to)$ has a unique normal form,
if every $a \in A$ has at most one normal form.

2．合流性の基本性質 (2/3)

■定理(合流性⇒正規形が一意)

(*A*,→)は，合流性をもつならば，
一意の正規形をもつ

（証明）
a の２つの正規形を b,c とすると，
$a \to^* b$，$a \to^* c$ および合流性より $b \downarrow c$.
b,c は正規形なので，$b=c$.

■Theorem (Confluence ⇒ Unique normal form)

Every confluent system has a unique normal form.

(Proof)
If a has two normal forms b and c, then from $a \to^* b$ and $a \to^* c$, we have $b \downarrow c$ from the confluence. Since b and c are normal forms, we have $b=c$.

合流性は関数型プログラムに望まれる性質
（関数の返す値は非決定的な並列計算をしても一意）

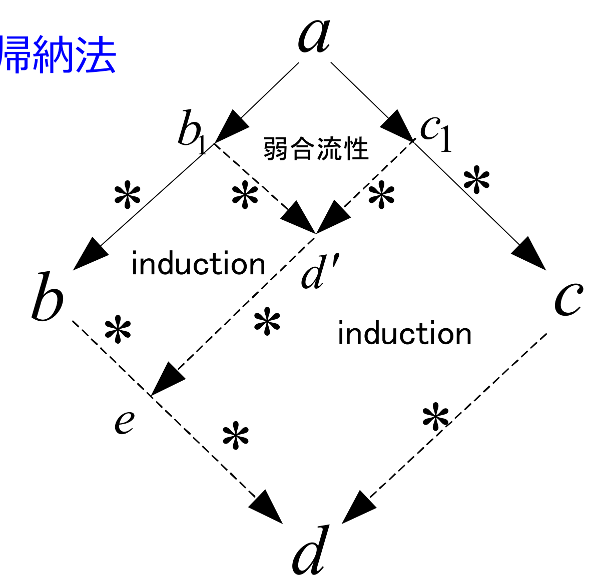
Confluence is a desirable property for functional programs, because the values returned by functions should be unique even under nondeterministic, parallel computation.

2．合流性の基本性質 (3/3)

■定理［ニューマンの補題］
(停止性＋弱合流性⇒合流性)

停止性と弱合流性をもつ $(A,\to)$ は合流性をもつ

(証明) 整礎帰納法



停止性の検証方法は前回学んだので，合流性を検証するにはあと弱合流性の検証法を学べばよい

■Theorem [Newmann's lemma]
(Termination + Weak confluence ⇒ Confluence)

Every terminating, weakly confluent system is confluent.

Proof by *well-founded induction*:

Termination of → allows us to use *well-founded induction*, where $a \to^+ b$ is a strict partial order that can be interpreted as "b is smaller than a", and then we can use the induction hypotheses $P(b)$ to establish the induction step $P(a)$.

Having already studied verification of termination, we will study verification of weak confluence to verify confluence.

【準備】代入(1/4)

(Preliminaries: Substitution)

■定義

代入 $\sigma = \{x_1/s_1, x_2/s_2, \cdots, x_n/s_n\}$

変数 x_i (i=1,2,⋯,n) を項 s_i に
同時に置き換える操作を表す関数

$$\sigma(x_i) = s_i$$

■Definition

A substitution $\sigma=\{x_1/s_1,\cdots, x_n/s_n\}$ represents the operation (function) that replaces all the occurrences of variables x_i (i=1,2,⋯,n) by the terms s_i. We will write $\sigma(x_i)=s_i$.

【Example】 $\sigma = \{x/a, y/g(z)\}$

x	y
a	$g(z)$

$$\sigma(x) = a$$

$$\sigma(y) = g(z)$$

As this example shows, a substitution can be well represented as a table that maps each variable to a term.

【準備】代入(2/4)

■定義

代入の（項 t への）適用 $\sigma(t)$

項 t 内のすべての変数に対して，σ で指定された置き換えを同時に一回行った結果を表す

■Definition

Application of substitution σ to term t is denoted by $\sigma(t)$.
This represents the term obtained from t by replacing all the variables x_i (i=1,2,$\cdots$,n) by s_i.

【Example】 $\sigma = \{x/a, y/g(z)\}$

$$t = g(x, f(c, x, y), z)$$

⇩

$$\sigma(t) = g(a, f(c, a, g(z)), z)$$

x	y
a	$g(z)$

代入σは，項の集合T からTへの関数
σ: $T \rightarrow T$
とみなすことができる

A substitution σ can be seen as a function from the set of terms T to T.

【準備】代入(3/4)

■定義　代入の合成 $\sigma_1 \circ \sigma_2$

$$(\sigma_1 \circ \sigma_2)(t) = \sigma_1(\sigma_2(t))$$

■Definition

The composition of two substitutions σ_1 and σ_2 is the substitution $\sigma_1 \circ \sigma_2$ defined by

$$(\sigma_1 \circ \sigma_2)(t) = \sigma_1(\sigma_2(t))$$

【Example】

$$\begin{tikzcd}
 & g(x,y) \arrow[dl, "\sigma = \{x/y\}"'] \arrow[dd, "\sigma' = \tau \circ \sigma = \{x/b, y/b\}"] \\
g(y,y) \arrow[dr, "\tau = \{y/b\}"'] & \\
 & g(b,b)
\end{tikzcd}$$

【準備】代入(4/4)

■定義　代入の一般性

代入 σ は代入 σ' より一般的である

$\Updownarrow$

ある代入 τ に対し，　$\sigma' = \tau \circ \sigma$

■Definition

A substitution σ is more general than a subsitution σ', if there exists a substitution τ such that $\sigma' = \tau \circ \sigma$.

【Example】

$$
\begin{tikzcd}
 & g(x,y) \arrow[dl, "\sigma = \{x/y\}"'] \arrow[d, "\sigma' = \tau \circ \sigma = \{x/b, y/b\}"] \\
g(y,y) \arrow[r, "\tau = \{y/b\}"'] & g(b,b)
\end{tikzcd}
$$

In this example, σ is more general than σ′. Note that, intuitively, the result *g*(*y*,*y*) of σ is more general than the result *g*(*b*,*b*) of σ′, because the universally quantified variable *y* is more general than the specific constant *b*.

3．単一化 (1/6) (Unification)

■定義 ■Definition

単一化

s, t に対し，ある代入 σ が存在して

$$\sigma(s) = \sigma(t)$$

とできるとき,
σ を s, t の単一化代入といい,
s, t は単一化可能であるという

Two terms s and t are unifiable, if there exists a substitution σ, called a unifier of s and t, such that

$$\sigma(s)= \sigma(t).$$

最汎単一化代入(*mgu*) σ

単一化代入のうち，最も一般的なもの.
（σ よりも一般的な単一化代入がない）

The most general unifier (*mgu*) σ of two terms s and t is a unifier σ such that there exists no unifier more general than σ.

3．単一化 (2/6)

【Example】 $f(x, g(x))$ $f(h(y), z)$

UNIFY

This is the *mgu*.

$$\sigma = \{x/h(y),\ z/g(h(y))\}$$

This is not the *mgu*.

$$\sigma' = \{x/h(a),\ y/a,\ z/g(h(a))\}$$

$$f(h(y), g(h(y)))$$

$$\tau = \{y/a\}$$

$$f(h(a), g(h(a)))$$

3．単一化 (3/6)

単一化アルゴリズム

【入力】項 s,t

【出力】項 s,t が単一化可能ならば *mgu*，
単一化可能でなければ「失敗」を出力

【手順】関数記号を解釈しない方程式 $s = t$ を変形し，$x_i = u_i$ の形の複数の方程式に変換して，代入(*mgu*)

$$\sigma=\{ x_i / u_i \mid 1 \leqq i \leqq n\}$$

を構成する．

A unification algorithm accepts two terms s and t as input, and outputs their *mgu* if they are unifiable; or the *failure* otherwise. Its procedure is basically the transformation of the equation $s=t$ (in which the function symbols are not interpreted) into a set of equations of the form $x_i=u_i$, from which the resultant substitution (*mgu*) $\sigma=\{x_i / u_i \mid 1 \leqq i \leqq n\}$ will be constructed.

【Example】

Input: $\underbrace{f(x,g(x))}_{s} = \underbrace{f(h(y),z)}_{t}$

Output: $\sigma = \begin{cases} x = h(y) \\ z = g(h(y)) \end{cases}$

3．単一化 (4/6)

単一化アルゴリズム

Step1. 連立方程式 $\{s = t\}$ に対し，等式に対するつぎの５つの変換操作を任意に繰り返し適用する（定数は引数0個の関数記号とみなす）

Unification algorithm

Step1. Apply repeatedly the following five transformation operations to the initial set of equations $\{s=t\}$. (Constants are regarded as function symbols with no arguments.)

(1) $f(s_1,\cdots,s_n) = f(t_1,\cdots,t_n)$ ⇒ transform to $s_1=t_1,\cdots,s_n=t_n$

(2) $f(s_1,\cdots,s_n) = g(t_1,\cdots,t_n)$ $(f \neq g)$ ⇒ return *failure*

(3) $t = x$ ⇒ transform to $x=t$

(*x is a variable and t is a non-variable term.*)

(4) $x = x$ ⇒ remove this equation

(5) $x = t$ $(t \neq x)$ ⇒ if x occurs in t,
then return *failure*
else if x occurs in other equations,
then apply the substitution $\{x/t\}$ to them

Occurrence check:
$x = f(x)$

3．単一化 (5/6)

Step2.　Step1のどの操作も適用できなくなったとき，連立方程式は

$$\{x_1=u_1,\cdots,x_n=u_n\}$$

の形になっており，左辺の各変数はどの右辺の項の中にも出現していない.

　このとき s, t は単一化可能で，

$$\sigma = \{x_1/u_1,\cdots,x_n/u_n\}$$

が*mgu*である

Step2. When no operations of Step 1 are applicable any more, the set of equations should be in the form

$$\{ x_i = u_i \mid 1 \leqq i \leqq n\}$$

and no variables in the left-hand sides occur in the right-hand sides.

In this case, s and t are unifiable and their *mgu* is

$$\sigma=\{ x_i / u_i \mid 1 \leqq i \leqq n\}$$

3．単一化 (6/6)

【Example】 Unify $f(x, g(x))$ and $f(h(y), z)$.

$$\{ f(x, g(x)) = f(h(y), z) \}$$

$$\Rightarrow \begin{cases} x = h(y) \\ g(x) = z \end{cases}$$

$$\Rightarrow \begin{cases} x = h(y) \\ z = g(x) \end{cases}$$

$$\Rightarrow \begin{cases} x = h(y) \\ z = g(h(y)) \end{cases}$$

$$\sigma = \{x/h(y), z/g(h(y))\}$$

4．危険対による合流性の判定 (1/5)

(Decision on confluence by critical pairs)

【動機】 2つのルール

$$f(f(x,y),z) \to f(x,f(y,z))$$
$$f(i(w),w) \to e$$

のどちらでも書換え可能な一般性のある重なりを求めるため，1つめのルールの左辺の部分項と2つめの左辺の全体の単一化を試す

$f(x,y)$ and $f(i(w),w)$ are unifiable with *mgu* $\sigma = \{x/i(w), y/w\}$.

【Motivation】 Consider two rules

f(*f*(*x*, *y*), *z*) → *f*(*x*, *f*(*y*, *z*))

f(*i*(*w*), *w*) → e.

To find general overlaps which can be reduced by any of them, try to unify a subterm of the left-hand side (LHS) of the first rule and the whole LHS of the second rule.

弱合流性を乱す可能性のパターン

σ(1つめのルールの左辺) ＝ 重なり

$$f(\underline{f(i(w),w)},z)$$

$$f(e,z) \quad \text{危険対} \quad f(i(w),f(w,z))$$

The instance (by σ) of the left-hand side of the first rule is called an overlap, representing a pattern that might violate weak confluence.

Rewriting the overlap in two ways, we have a critical pair.

4．危険対による合流性の判定 (2/5)

$l_1 \to r_1,\ l_2 \to r_2$

互いに共通の変数を持たないように適切に変数名を付け替えてある２つの書換え規則

■定義　危険対（形式的な定義：詳しい解説は省略）

$l_1[s] \to r_1$　：s は l_1 の部分項で非変数

$l_2 \to r_2$

$\sigma(s) = \sigma(l_2)$　：s と l_2 は σ により単一化可能

危険対 $\langle \sigma(l_1)[\sigma(r_2)], \sigma(r_1) \rangle$

$$\sigma(l_1[s]) = \sigma(l_1)[\sigma(s)]$$

$\sigma(l_1)[\sigma(r_2)]$　　　$\sigma(r_1)$

Let $l_1 \to r_1$ and $l_2 \to r_2$ be two rules in which variables are renamed so that they share no variables in common.

■Definition

A critical pair is a pair of terms $\sigma(l_1)[\sigma(r_2)]$ and $\sigma(r_1)$, where σ is the *mgu* of l_2 and a non-variable subterm s of l_1 (i.e., $\sigma(s) = \sigma(l_2)$). The notation $l_1[s]$ emphasizes that l_1 contains s as its subterm. $\sigma(l_1)[\sigma(r_2)]$ represents the term obtained from the overlap $\sigma(l_1)[\sigma(s)]$ by replacing the subterm $\sigma(s)$ $(=\sigma(l_2))$ by $\sigma(r_2)$.

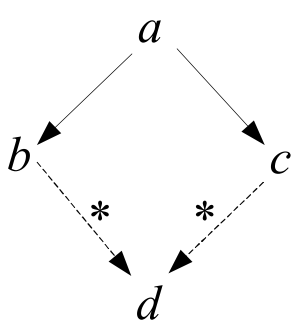
4．危険対による合流性の判定 (3/5)

■定理（危険対定理）

項書換え系 R が弱合流性をもつための必要十分条件は，R のすべての危険対 $<p,q>$ が会同すること $(p\downarrow q)$ である

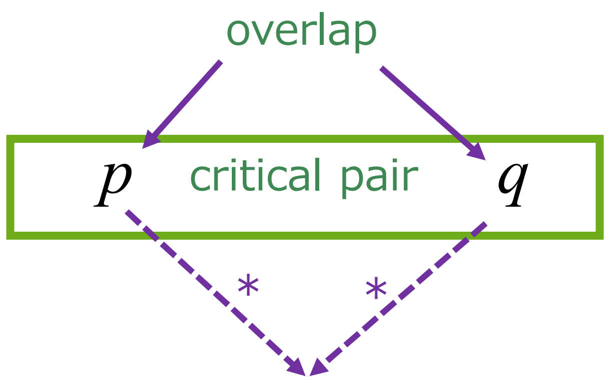
■Theorem (Critical pair theorem)

A term rewriting system R is weakly confluent if, and only if, every critical pair $<p,q>$ is joinable, i.e. $p\downarrow q$.



弱合流性

(weak confluence)



有限個の危険対だけ考えればよい

We need to think about only a *finite* number of critical pairs to verify the weak confluence.

4．危険対による合流性の判定 (4/5)

系（停止性 ⇒ 危険対による合流性判定）

停止性をもつ項書換え系 R が合流性をもつための必要十分条件は，R のすべての危険対 <p,q> について p と q の正規形が一致することである

（証明）
ニューマンの補題（停止性∧弱合流性⇒合流性）と危険対定理を組み合わせる

■Corollary (Confluence check by critical pairs for terminating TRS)

A terminating, term rewriting system R is confluent if and only if for all critical pairs <p,q>, normal forms of p and q are identical.

(Proof) Combine the Newmann's lemma with the Critical pair theorem.

Algorithm
Step 1. Let S = the finite set of all critical pairs of R.
Step 2. For each critical pair <p,q> in S
Let p^* = a normal form of p;
q^* = a normal form of q.
If $p^* \neq q^*$, then return *false* (R is *not* confluent).
Step 3. Return *true* (R is confluent).

4．危険対による合流性の判定 (5/5)

【Example】

$$R = \begin{cases} f(g(x)) \to h(x,x) \\ g(e) \to e \\ f(e) \to h(e,e) \end{cases}$$

停止性：あり　(証明は省略)

It is not hard to show that R is terminating.

$f(e) \leftarrow \underline{f(\underline{g(e)})} \to h(e,e)$ から

唯一の危険対 $\langle f(e), h(e,e) \rangle$ が得られる

R has only one critical pair <$f(e)$,$h(e,e)$>.

２つの項の正規形はともに $h(e,e)$

なので，R は合流性を満たす

Since the normal forms of both $f(e)$ and $h(e,e)$ are the same term $h(e,e)$, R is confluent.

演習問題７

(1) 停止性を満たすつぎの項書換え系 *R* が
合流性をもつことを示せ.
（危険対は２つある）

$$R = \begin{cases} plus(x,0) & \to & x \\ plus(0,y_1) & \to & y_1 \\ plus(s(x_2),y_2) & \to & s(plus(x_2,y_2)) \end{cases}$$

(2) つぎの項書換え系 *S* は合流性をもたない
ことを示せ.
（１番目と２番目の書換え規則から得られる２つの
危険対のうちの一方で，正規形が一致しない）

$$S = \begin{cases} plus(x,plus(y,z)) & \to & plus(plus(x,y),z) \\ plus(0,y_1) & \to & y_1 \\ plus(s(x_2),y_2) & \to & s(plus(x_2,y_2)) \end{cases}$$

EXERCISE 7

Show that the following terminating TRS *R* is confluent.
(Hint: *R* has two critical pairs.)

Verify that the following TRS *S* is *not* confluent.
(Hint: The first two rules have two critical pairs, one of which consists of two terms that have normal forms different from each other.)