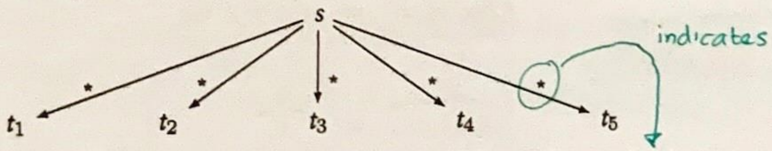
10

REWRITING pt. 2

Canonical Normal Form

For some rewrite rule sets, order of application might affect result.

→ We might have:



where all of t_1, t_2, t_3, t_4, t_5 are in normal form after multiple (zero or more) rewrite rule applications.

→ If all the normal forms are identical we can say we have a **canonical** normal form for s.

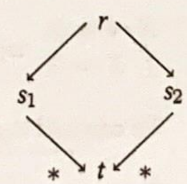
→ This is a very nice property! — You don't have to worry about search/backtracking.

- Means that order of rewrite rule application doesn't matter
- In general, means our rewrites are simplifying the expression in a canonical (safe) way.

Local Confluence

The properties of Church-Rosser and confluence can be difficult to prove. A weaker definition is useful:

A set of rewrite rules is **locally confluent** if for all terms r, s_1, s_2 such that $r \longrightarrow s_1$ and $r \longrightarrow s_2$ there exists a term t such that $s_1 \longrightarrow^* t$ and $s_2 \longrightarrow^* t$.



Theorem (Newman's Lemma)
local confluence + termination = confluence

Also: local confluence is decidable (due to Knuth and Bendix)

Both theorem and the decision procedure use idea of **critical pairs**

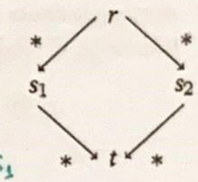
Confluence and Church-Rosser

QN: How do we know when a set of rules yields canonical normal forms?

A set of rewrite rules is **confluent** if for all terms r, s_1, s_2 such that $r \longrightarrow^* s_1$ and $r \longrightarrow^* s_2$ there exists a term t such that $s_1 \longrightarrow^* t$ and $s_2 \longrightarrow^* t$. — rewrites in zero/more steps

equivalent

A set of rewrite rules is **Church-Rosser** if for all terms s_1 and s_2 such that $s_1 \leftrightarrow^* s_2$, there exists a term t such that $s_1 \longrightarrow^* t$ and $s_2 \longrightarrow^* t$. — $s_1 \to^* s_2$ OR $s_2 \to^* s_1$



Theorem
Church-Rosser is equivalent to confluence.

Theorem
*For **terminating** rewrite sets, these properties mean that any expression will rewrite to a canonical normal form.*

termination + confluence = canonical normal form

Choices in Rewriting

How can choices arise in rewriting?

- Multiple rules apply to a single redex: order might matter
- Rules apply to multiple redexes:
  - if they are separate: order does not matter — i.e. $s + t$
  - if one contains the other: order might matter — i.e. $\underbrace{succ(\overbrace{a+0}^{①})}_{②}$ — ② Potentially there's a way of reducing ① or ②

We are interested in cases where the order matters:

Rules	Rewrites	Concrete Critical Pair
$X^0 \Rightarrow 1$ $0^Y \Rightarrow 0$	0^0 rewrites to 0 and to 1	$\langle 0, 1\rangle$
$X \cdot e \Rightarrow X$ $(X \cdot Y) \cdot Z \Rightarrow X \cdot (Y \cdot Z)$	$(x \cdot e) \cdot z$ rewrites to $x \cdot z$ and $x \cdot (e \cdot z)$	$\langle x \cdot z, x \cdot (e \cdot z)\rangle$

$\underbrace{(\overbrace{x \cdot e}^{①}) \cdot z}_{②}$

Critical Pairs

- Given two rules $L_1 \Rightarrow R_1$ and $L_2 \Rightarrow R_2$, we are concerned with the case when there exists a *non-variable* sub-term s of L_1 such that $s[\theta] = L_2[\theta]$, with most general unifier θ.

Applying these rules in different orders gives rise to a **critical pair**, where $L_1[\theta]\{R_2[\theta]/s[\theta]\}$ denotes replacing $s[\theta]$ by $R_2[\theta]$ in $L_1[\theta]$.

$$L_1[\theta]$$

$$R_1[\theta] \qquad L_1[\theta]\{R_2[\theta]/s[\theta]\}$$

$\langle R_1[\theta], L_1[\theta]/\{R_2[\theta]/s[\theta]\}\rangle$ is the critical pair.

- Note: the variables in the two rules should be *renamed* so they do **not** share any variable names.
- Note: A rewrite rule may have critical pairs with itself e.g. consider the rule $f(f(x)) \Rightarrow g(x)$.

With $W \cdot e \Rightarrow W$ and $(X \cdot Y) \cdot Z \Rightarrow X \cdot (Y \cdot Z)$, where X, Y and Z are variables, we can have $\theta = [W/X, e/Y]$, any other?

Critical Pairs: Example

e.g.

Consider the rewrite rules:

Note tht. vars. in the 2 rules do NOT share any var. names

$$\overbrace{f(\underbrace{f(x, y)}_{s}, z)}^{L_1} \Rightarrow \overbrace{f(x, f(y, z))}^{R_1}$$

$$\overbrace{f(i(x_1), x_1)}^{L_2} \Rightarrow \overbrace{e}^{R_2}$$

Unify L_2 w/ s : $L_2 \equiv s$

$f(i(x_1), x_1) \equiv f(x, y)$

The mgu θ, given our choice of non-variable subterm s of L_1, is given by $\theta = \{i(x_1)/x, x_1/y\}$ and by considering:

Apply θ to L_1

$$\boxed{f(\underline{f(i(x_1), x_1)}, z)}$$

$s[\theta]$

$R_1[\theta]$ $\boxed{f(i(x_1), f(x_1, z))}$ $\qquad$ $\boxed{f(e, z)}$ $L_1[\theta]\{R_2[\theta]/s[\theta]\}$

Replace $s[\theta]$ w/ $R_2[\theta]$

$f(i(x_1), x_1) \equiv e$

We get the critical pair $\langle f(i(x_1), f(x_1, z)), f(e, z)\rangle$.

Testing for Local Confluence

If we can conflate (join) all the critical pairs, then have **local confluence**.

Conflation for a critical pair $\langle s_1, s_2\rangle$ is when there is a t such that $s_1 \longrightarrow^* t$ and $s_2 \longrightarrow^* t$.

An algorithm to test for local confluence (assuming termination):

1. Find all the critical pairs in set of rewrite rules R
2. For each critical pair $\langle s_1, s_2\rangle$:
   - 2.1 Find a normal form s_1' of s_1; — try to rewrite s_1, s_2
   - 2.2 Find a normal form s_2' of s_2; / until you can't anymore
   - 2.3 Check $s_1' \equiv s_2'$, if not then fail.

Establishing Local Confluence

- Sometimes a set of rules is not locally confluent

$$X \cdot e \Rightarrow X$$
$$f \cdot X \Rightarrow X$$

is not locally confluent: $\boxed{\langle f, e\rangle}$ does not conflate.

- We can add the rule $f \Rightarrow e$ to make this critical pair joinable.

However, adding new rules requires care:

- Must preserve termination
- Might give rise to *new* critical pairs and so we may need to check local confluence again.

"A rewrite rule may have CPs w/ itself"

Establishing Local Confluence: Example

e.g.

Consider the set R consisting of just one rewrite rule, with x a variable:

$$f(f(x)) \Rightarrow g(x)$$

which has exactly one critical pair (CP) when it is overlapped with a *renamed* copy of itself $f(f(y)) \Rightarrow g(y)$. The lhs $f(f(x))$ unifies with the subterm $f(y)$ of the renamed lhs to produce the $\boxed{\text{mgu } \{f(x)/y\}:}$ θ

$$f(f(f(x)))$$

$R_1[\theta]$ $\boxed{g(f(x))}$ $\boxed{f(g(x))}$ $L_1[\theta]\{R_2[\theta]/s[\theta]\}$

$\langle g(f(x)), f(g(x))\rangle$ is the critical pair.

- This CP is not joinable, so R is not locally confluent.
- Adding the rule $f(g(x)) \Rightarrow g(f(x))$ to R makes the pair joinable.
- The enlarged R is terminating (how?), but NEW!
- (After renaming) new CP: $\boxed{\langle g(g(z)), f(g(f(z)))\rangle}$ arises (how?);
- LC test: it is joinable, $f(g(f(z))) \to g(f(f(z))) \to g(g(z))$.

Apply rule $f(g(x)) \Rightarrow g(f(x))$

$$\overbrace{f(f(x))}^{L_2} \Rightarrow \overbrace{g(x)}^{R_2}$$

!! RENAME

$$\underbrace{f(\overbrace{f(y)}^{s})}_{L_1} \Rightarrow \underbrace{g(y)}_{R_1}$$

The enlarged R is terminating bc.

avg. depth of all g()'s on term tree decreases

Knuth-Bendix (KB) Completion Algorithm

Start with a set R of $\boxed{\text{terminating}}$ rewrite rules

While there are $\boxed{\text{non-conflatable}}$ critical pairs in R:

1. Take a critical pair $\langle s_1, s_2\rangle$ in R
2. Normalise s_1 to s_1' and s_2 to s_2' (and we know $s_1' \neq s_2'$)
3. if $R \cup \{s_1' \Rightarrow s_2'\}$ is terminating then
   $R := R \cup \{s_1' \Rightarrow s_2'\}$
   else if $R \cup \{s_2' \Rightarrow s_1'\}$ is terminating then
   $R := R \cup \{s_2' \Rightarrow s_1'\}$
   else Fail

- If KB succeeds then we have a locally confluent and terminating (and hence confluent) rewrite set (KB may run forever!)
- Depends on the termination check: define a measure and use that to test for termination.

easy to say BUT...

Showing termination is non-trivial

Q: Briefly state adv. & disadv. of using rewriting as an automated reasoning techique

(+)	(−)
1) When confluent & terminating, rewriting is a decision procedure for equational goals in the theory defined by the rules.	1) Rewriting is not complete if the rule set is either non-confluent or non-terminating. (Smthg tht is provable could be unprovable)
2) Quite efficient! (bc. uses matching rather than unification)	2) Knowledge and goals must often be represented in the form of equations (Restricts applications of rewriting)