Rewriting

In mathematical logic and computer science, rewriting is a formalism to describe certain non-deterministic computations. If combined with an algorithm of how to apply the rewrite rules, or in a so-called deterministic rewrite system, a rewrite system can represent an algorithm, as we will discuss below.

For a start, and for most of what we need today, we consider a very simple form of rewrite systems, known as string rewrite systems. A string rewrite system S is a finite set of rewrite rules of the form $l \to r$ where l and r are strings, i.e., words, in alphabet A. Using S, an input word w in alphabet A rewrites to a word u in one step, denoted by

$$w \to u$$

if and only if there exists a rewrite rule $l \to r$ in S such that $w = xly$ and $u = xry$ for some words x and y. We write $w \to^* u$ if w rewrites to u in zero or more steps.

We say that u is an *end-result* of rewriting w by S, or a *normal form*, if $w \to^* u$ and u can not be rewritten further. We say that S provides/is an *algorithm*, if every word can be rewritten to a unique normal form. We say that a rewrite system is *terminating* if there is no infinite rewriting sequence. We also say that a rewrite system is *deterministic* if the rules are such that at any point in time only a single rule is applicable. A rewrite system that is deterministic and terminating provides an algorithm.

Task 1 Consider the following set of rewrite rules, that can be applied on words in alphabet $\{a, b\}$: $aa \to b, \quad ba \to ab, \quad bb \to b$

- Rewrite *aabbab* in three steps.
- Rewrite *aabbab* to a normal form.
- Does this rewrite system provide an algorithm? If yes, what does the provided algorithm compute?

Task 2 We now consider a small puzzle, known as the camel puzzle: Four Tasmanian camels traveling on a very narrow ledge encounter four Tasmanian camels coming the other way.

As everyone knows, Tasmanian camels never go backwards, especially when on a precarious ledge. The camels will climb over each other, but only if there is a camel-sized space on the other side.

The camels did not see each other until there was only exactly one camel's width between the two groups.

How can all camels pass, allowing both groups to go on their way, without any camel reversing?

Formalise the camels puzzle with a string rewrite system, and then present the solution as rewriting to a normal form. Is this string rewrite system an algorithm? Why? ◁

Task 3 Consider the rewrite system that can be applied on words in alphabet $\{a, b, c\}$ consisting of a single rule $a \to aa$

What does this system do ? Is this system an algorithm ? Argument your answer (consider, e.g., the rewriting tree from the words bbc and aba). ◁

Rewrite systems need not be restricted to words. We may also rewrite subsets, or multisets – as in the following task. More generally, one speaks of *term rewriting systems* of which the string and (multi)set rewriting systems are a special case. One can also rewrite graphs, or other complex objects.

Task 4 Mulan is trading on an imaginary market. She sells two apples for a banana, three bananas for a coconut and three apples, and an apple and a coconut for two mangoes and a banana. Imagine she arrives at the market with three apples, a banana, and a coconut.

- Draw the whole tree of possible rewrites and show that she can leave the market with at least one of each of the fruits.
- Show that she can leave the market with an apple, a banana, a coconut, and two mangoes. Try also using backward reasoning / proof search.
- Find the best possible deal that Mulan can make (most fruits to take home, or another criterion for evaluating deals). ◁

Finally, we very briefly touch upon your next topic.

Task 5 Give a string rewrite system that is an algorithm for sorting a string in alphabet $A = \{a, b, c, d\}$ in alphabetic order. That is, for example, $abdccbadd \to^* aabbccddd$.