G6021 Comparative Programming

Reduction and reduction graphs: functional expressions

A reduction graph simply shows all the different ways that an expression can be reduced. In this work we use three words interchangeably: reduce, simplify, and evaluate. We introduce some more terminology:

- A *reducible expression* (abbreviated to *redex*) is an expression that can be simplified. Example: $3 + 4$ can be simplified to 7, so $3 + 4$ is a redex. It is useful in our work that we underline redexes in expressions. Examples: $\underline{3 + 4}$, $3 + \underline{4 * 5}$, $\underline{1 * 2} + \underline{3 * 4}$. Note that when we reduce a redex, we may create a new one: $3 + \underline{4 * 5} \rightarrow \underline{3 + 20}$. We will also see examples below where redexes overlap.
- An expression that does not have a redex is said to be in *normal form*. Example: $3 + 4$ is not a normal form, but 7 is a normal form.
- We say that an expression *has a normal form* if there is a sequence of reductions that leads to a normal form. Example: $3 + 4$ has a normal form, because $3 + 4$ simplifies to 7 which is a normal form. `infinity`, defined in the lecture notes, is an example of an expression that does not have a normal form.

Reduce the following expressions to normal form, showing all alternative reductions with a reduction graph.

1. Draw the reduction graph for the expression $3 + 4 * 5$.
2. Draw the reduction graph for $(3 + 4) * (3 + 4)$.
3. Consider the following program written in Haskell syntax:

```
sq x = x*x
twice f x = f(f(x))
```

   (a) Draw the reduction graph for `sq (3+4)`

   (b) Draw the reduction graph for `twice sq 3`