Quantum Lift

by Sven Nilsen, 2020

In this paper I introduce the `qlift` function, which makes it possible to construct arbitrary quantum functions satisfying Schrödinger equation out of quantum Phi functions using ordinary source code.

The `qlift` function is an imaginary function (its source code can not be written down):

$$\text{qlift} : (T \to ()) \times X \to (T \to X)$$

$$\exists_{pc}\text{qlift}(f, x_0)(t) \text{ <=> } \backslash(x_1 : X) = \text{if } x_0 == x_1 \ \{ \ \text{sqrt}((\exists_p x_0)(x_0)) \cdot (\exists_{pc} f(t))(()) \ \} \text{ else } \{ \ 0 \ \}$$

The probabilistic `$\exists_p$qlift` is undefined, because the functions returned from `qlift` redefines what the complex probabilistic existential path does. Otherwise, it would contradict probability theory.

What `qlift` does is to bind the probability of a program generating a value `$x_0$` to quantum behavior.

Usually, the `qlift` function is combined with `phi` (see paper “Quantum Schrödinger Functions”). The complex probability amplitudes of `f` over time is scaled with the probability of `$x_0$`.

This means, since values generated by a non-deterministic program adds probabilities up to `1`, that multiple qlifts can be used to construct arbitrary quantum functions satisfying Schrödinger equation.

For example:

```
f() = if random() < 0.2 { qlift(phi(1), false) } else { qlift(phi(2), true) }

f : () → (time → bool)
```

Intuitively, `f()` returns a quantum function rotating a complex probability amplitude over time with frequency either `1` or `2`. One can tell which `phi` function that was used from the boolean. However, the identity of this quantum `phi` function is not known before it has been called with a time argument!

When calling `f()(t)`, it returns `false` and `true` with complex probabilities:

false	true
$0.2\varphi(1)(t)$	$0.8\varphi(2)(t)$

Each of these states satisfies the Schrödinger equation. When two solutions of the Schrödinger equation is combined, the new wavefunction also satisfies the Schrödinger equation.

Notice that `f` is order-free, which is important to construct quantum functions implicitly. For more information about order-free quantum functions, see paper “Order-Free Quantum Non-Determinism”.