ssr-bit: from bits to sets, and back

Pierre-Evariste Dagand

Can you tell what the following program (devised by Gosper) does?

```
unsigned snoob(unsigned x)
{
  unsigned smallest = x & -x;
  unsigned ripple = x + smallest;
  unsigned ones = x ^ ripple;
  ones = (ones >> 2) / smallest;
  return ripple | ones;
}
```

I pretend that this is a (very efficient) routine for enumerating the subsets of an ordered set according to their lexicographic order.

To reason about this kind of programs (as well as the ones you wrote in the datalab) and specify them in terms of set-theoretic operations, we are developing a Coq library called ssr-bit[1] that provides a model of bit-level operations and relate them to the mathematical abstractions provided by the math-components[2] library.

This project can be tackled through several angles:

1. one could consolidate and extend the library to enable reasoning about non-trivial programs (such as the above) taken from the bible of bit-twiddling: "Hacker's Delight", by Warren.
2. one could focus on proof automation, transferring techniques from automated theorem provers, such as SMT solvers, and integrating them in the workflow of an interactive theorem prover, such as Coq.
3. one could develop techniques to semi-automatically synthesise efficient, bit-level implementations from high-level, set-theoretic specifications, exploiting the unique facility offered by an interactive theorem prover.

[1] https://github.com/pedagand/ssrbit
[2] https://math-comp.github.io/math-comp/