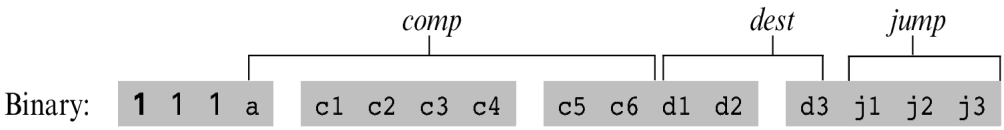
4.2.3 The *C*-Instruction

The *C*-instruction is the programming workhorse of the Hack platform—the instruction that gets almost everything done. The instruction code is a specification that answers three questions: (a) what to compute, (b) where to store the computed value, and (c) what to do next? Along with the *A*-instruction, these specifications determine all the possible operations of the computer.

C-instruction: *dest=comp;jump* // Either the *dest* or *jump* fields may be empty.
// If *dest* is empty, the "=" is omitted;
// If *jump* is empty, the ";" is omitted.



The leftmost bit is the *C*-instruction code, which is 1. The next two bits are not used. The remaining bits form three fields that correspond to the three parts of the instruction's symbolic representation. The overall semantics of the symbolic instruction *dest = comp;jump* is as follows. The *comp* field instructs the ALU what to compute. The *dest* field instructs where to store the computed value (ALU output). The *jump* field specifies a jump condition, namely, which command to fetch and execute next. We now describe the format and semantics of each of the three fields.

The Computation Specification The Hack ALU is designed to compute a fixed set of functions on the D, A, and M registers (where M stands for Memory[A]). The computed function is specified by the `a`-bit and the six `c`-bits comprising the instruction's *comp* field. This 7-bit pattern can potentially code 128 different functions, of which only the 28 listed in figure 4.3 are documented in the language specification.

Recall that the format of the *C*-instruction is `111a cccc ccdd djjj`. Suppose we want to have the ALU compute `D-1`, the current value of the D register minus 1. According to figure 4.3, this can be done by issuing the instruction `111`**`0 0011 10`**`00 0000` (the 7-bit operation code is in bold). To compute the value of `D|M`, we issue the instruction `111`**`1 0101 01`**`00 0000`. To compute the constant −1, we issue the instruction `111`**`0 1110 10`**`00 0000`, and so on.

The Destination Specification The value computed by the *comp* part of the *C*-instruction can be stored in several destinations, as specified by the instruction's 3-bit