Consider the two programs below. The program in Fig-1 is 4.70x faster than the program in Fig-2, when they run with arrays a, b and c having size 4K. Both programs were compiled with clang v4 at -O2.

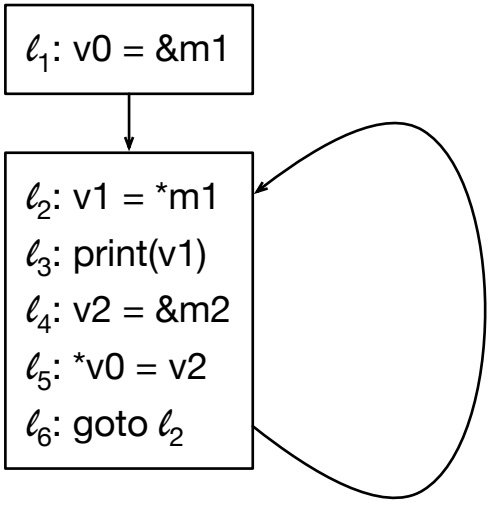

Figure 1: Exemplo of program that manipulates pointers. Consider that m1 and m2 are pre-allocated memory blocks.

Let P(x) be the "points-to set" of variable x, meaning that x might point to any variable within P(x). For instance, P(v0) = {m1} in this example.

1. Could you fill up the points-to set for the following variables:

   P(v0)
   P(v1)
   P(v2)
   P(m1)
   P(m2)

   Assume that the whole program is given on the left; hence, every points-to set should start empty.

2. Can you came up with equations to solve the "points-to analysis"? The points-to analysis is a static code analysis that finds a conservative estimate of the points-to set of each variable. This is a "may" analysis: if it says that P(x) contains {m}, then variable x might point to m during some execution of the program. However, it might also never point to that variable.

3. Given this conservative nature of points-to analysis, would it be wrong to say that P(x) is V, where V is the set of program variables, for any variable `x`?

UFMG
DCC
DEPARTAMENTO DE CIÊNCIA DA COMPUTAÇÃO
Compilers Laboratory