Foundations of Monotone Data-Flow Analysis

Max Hailperin

MCS-388 class notes, 2014-05-02

Suppose there are n blocks, $B_1 \ldots B_n$, and that for each block we've got IN and OUT values, $\text{IN}[B_1], \text{OUT}[B_1], \ldots, \text{IN}[B_n], \text{OUT}[B_n]$ that are each drawn from a finite-height lattice V with top element $\top$ and partial order $\leq$. We can treat all the values taken together, $(\text{IN}[B_1], \text{OUT}[B_1], \ldots, \text{IN}[B_n], \text{OUT}[B_n])$, as being an element of a lattice V^{2n} that has its own top element $\top^{2n} = (\top, \top, \ldots, \top)$ and its own partial order $\sqsubseteq$ defined as follows. We say

$$(\text{IN}_a[B_1], \text{OUT}_a[B_1], \ldots) \sqsubseteq (\text{IN}_b[B_1], \text{OUT}_b[B_1], \ldots)$$

if and only if for all i from 1 to n, $\text{IN}_a[B_i] \leq \text{IN}_b[B_i]$ and $\text{OUT}_a[B_i] \leq \text{OUT}_b[B_i]$. Note that because V has finite height, so does V^{2n}.

If we focus on the outer loop of the iterative data-flow analysis algorithm, we see it is computing a new collection of IN and OUT values from an old collection. That is, it is iterating from one element of V^{2n} to the next. We can abstract all the work done by the inner loop (the loop over the blocks) as being a single monotone function, ϕ, from V^{2n} to V^{2n}, which describes how the whole collection of IN and OUT values is updated by each pass through the outer loop. As such, the iterative data-flow analysis algorithm is really quite straightforward: it starts with $\top^{2n}$ and computes from it $\phi(\top^{2n})$, $\phi(\phi(\top^{2n}))$, ... We can can call the collection of IN and OUT values after k iterations $\phi^k(\top^{2n})$. The algorithm terminates when $\phi^k(\top^{2n}) = \phi^{k+1}(\top^{2n})$.

From the algorithm's termination condition, it follows that if the algorithm does terminate, the final value $\phi^k(\top^{2n})$ is a fixed point of ϕ, that is, a value X such that $\phi(X) = X$. To show that the algorithm terminates, we can use the fact that V^{2n} has finite height together with an additional fact that is provable by induction on k: for all $k \geq 0$,

$$\phi^k(\top^{2n}) \sqsupseteq \phi^{k+1}(\top^{2n}).$$

The base case follows directly from the fact that $\phi^0(\top^{2n}) = \top^{2n}$ is the top element of V^{2n}. Each subsequent case can be derived from the preceding

one by using the monotonicity of ϕ. Thus, we know that the iterates form a descending chain, which (given finite height) must terminate in a fixed point.

Having shown that the algorithm terminates with a fixed point of ϕ, our next goal is to show that this terminating fixed point is the maximum fixed point. Suppose we pick any arbitrary fixed point of, X; that is, $\phi(X) = X$. We need to show that the algorithm's terminating fixed point is $\sqsupseteq X$. We can show an even more general result by induction on k: for all $k \geq 0$,

$$\phi^k(\top^{2n}) \sqsupseteq X.$$

That is, not only is the algorithm's terminating fixed point $\sqsupseteq X$, but all the earlier iterates are as well. The base case again follows directly from the fact that $\phi^0(\top^{2n}) = \top^{2n}$ is the top element of V^{2n}. Each subsequent case can be derived from the preceding one by using the monotonicity of ϕ together with the fact that X is a fixed point of ϕ.

Some remaining questions include:

- How efficient is this algorithm?
- Are there alternative algorithms that compute the same result?
- Why can we be sure that a fixed point conservatively approximates (provides a lower bound for) what would be true along any particular path through the program? (The maximum fixed point in particular, then, is conservative, and is less so than any other fixed point.)