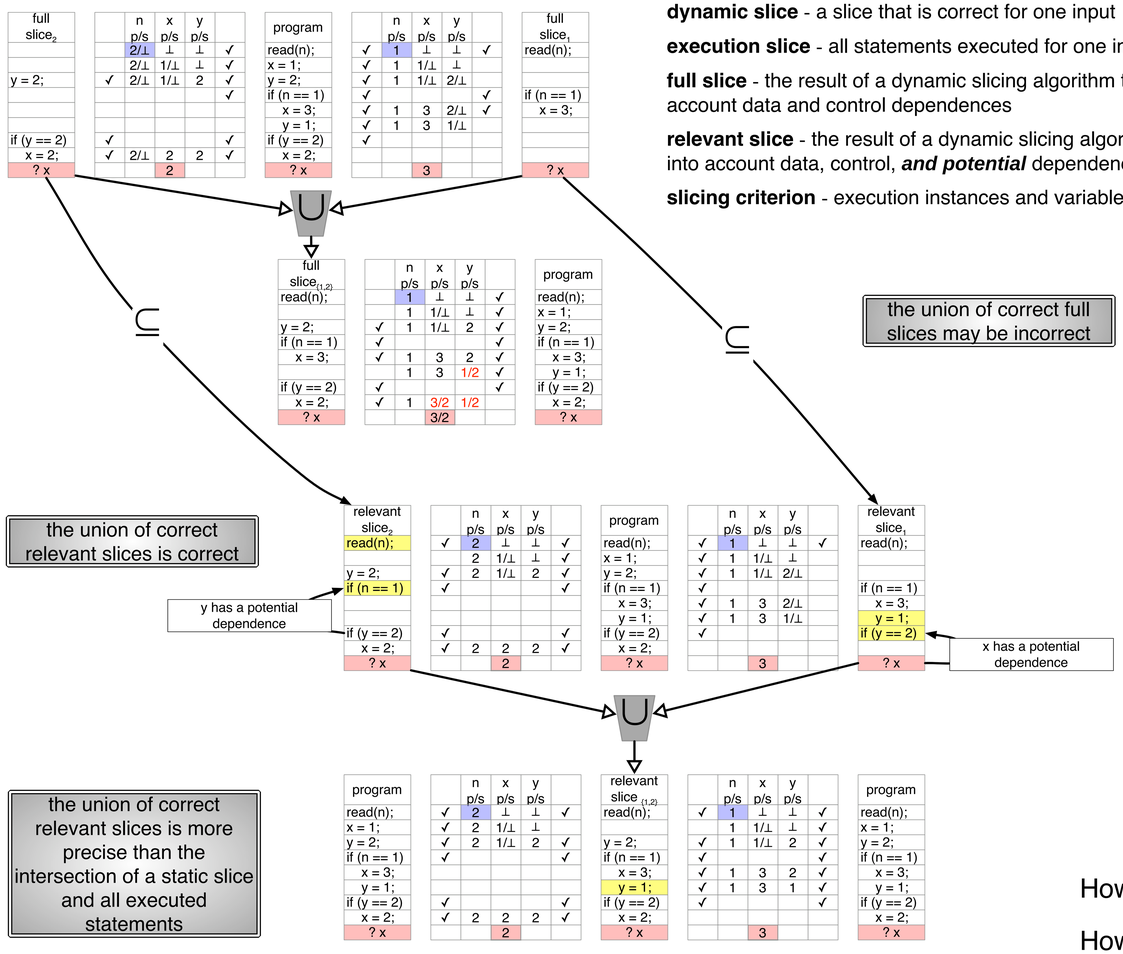
EFFECTIVE SLICING: A GENERALIZATION OF FULL AND RELEVANT SLICING

ANNE MULHERN UNIVERSITY OF WISCONSIN-MADISON



correct unions of dynamic slices are desirable

debugging: multiple inputs cause failure

program understanding: an option to a program may have multiple values, but the programmer is interested in the behavior of the program when the option assumes any value

dynamic slice - a slice that is correct for one input

execution slice - all statements executed for one input

full slice - the result of a dynamic slicing algorithm that takes into account data and control dependences

relevant slice - the result of a dynamic slicing algorithm that takes into account data, control, ***and potential*** dependences

slicing criterion - execution instances and variables of interest

Future Work

How many steps to reach fixpoint?

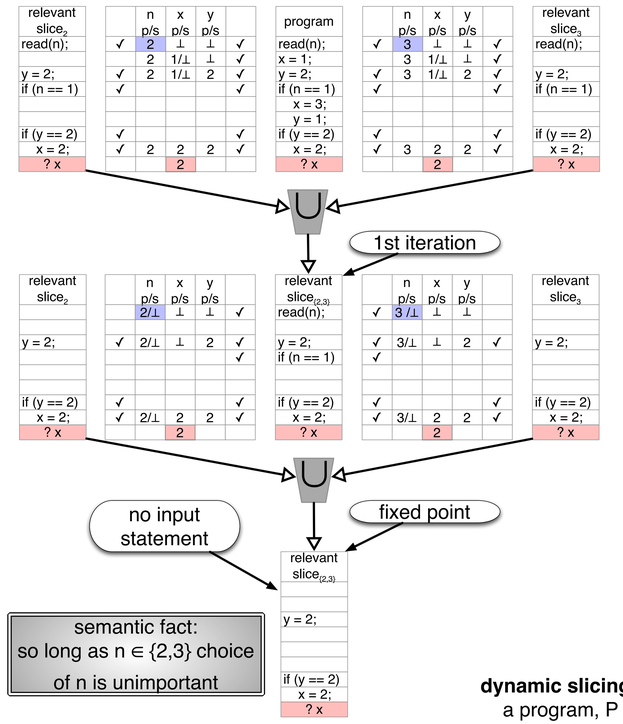
How big are unions of relevant slices?

How good an approximation can effective slicing give?

How many, and what sort, of dynamic semantic facts can be extracted?

What other interesting generalizations of dynamic slicing exist?

Fixed Point Computation



Foundational

Weiser [1981]

Korel and Laski [1988]

Agrawal and Horgan [1990]

Formalization

Binkley, Danicic, Gyimóthy, Harman, Kiss, Lakhota, Ouarbya [2004 - 2006]

Unions of Slices

Beszédes, Csirik, Danicic, Faragó, Gyimóthy, Hall, Harman, Hierons, Krinke, Lucia, Szabó [1995 - 2007]

Practical Dynamic Slicing

Beszédes, Csirik, Forgács, Gergely, Gupta, Gupta, Gyimóthy, He, Szabó, Tallam, Zhang, Zhang [2001 - 2007]

dynamic slicing algorithm - a function of two arguments:
a program, P
an initial state, σ
examples: full(P, σ), relevant(P, σ)

effective slicing algorithm - a relevant slicing algorithm abstracted over a set, Π, of potential statements:
effective(P, σ, Π)
effective(P, σ, $\varnothing$) = full(P, σ)
effective(P, σ, P) = relevant(P, σ)

Π is the union of execution slices