Data Flow Coverage

Control-Flow-Graph-Based Coverage Criteria

- Statement Coverage
- Path Coverage
- Branch Coverage
- Hidden Paths
- Loop Guidelines
  - General
  - Boundary - Interior

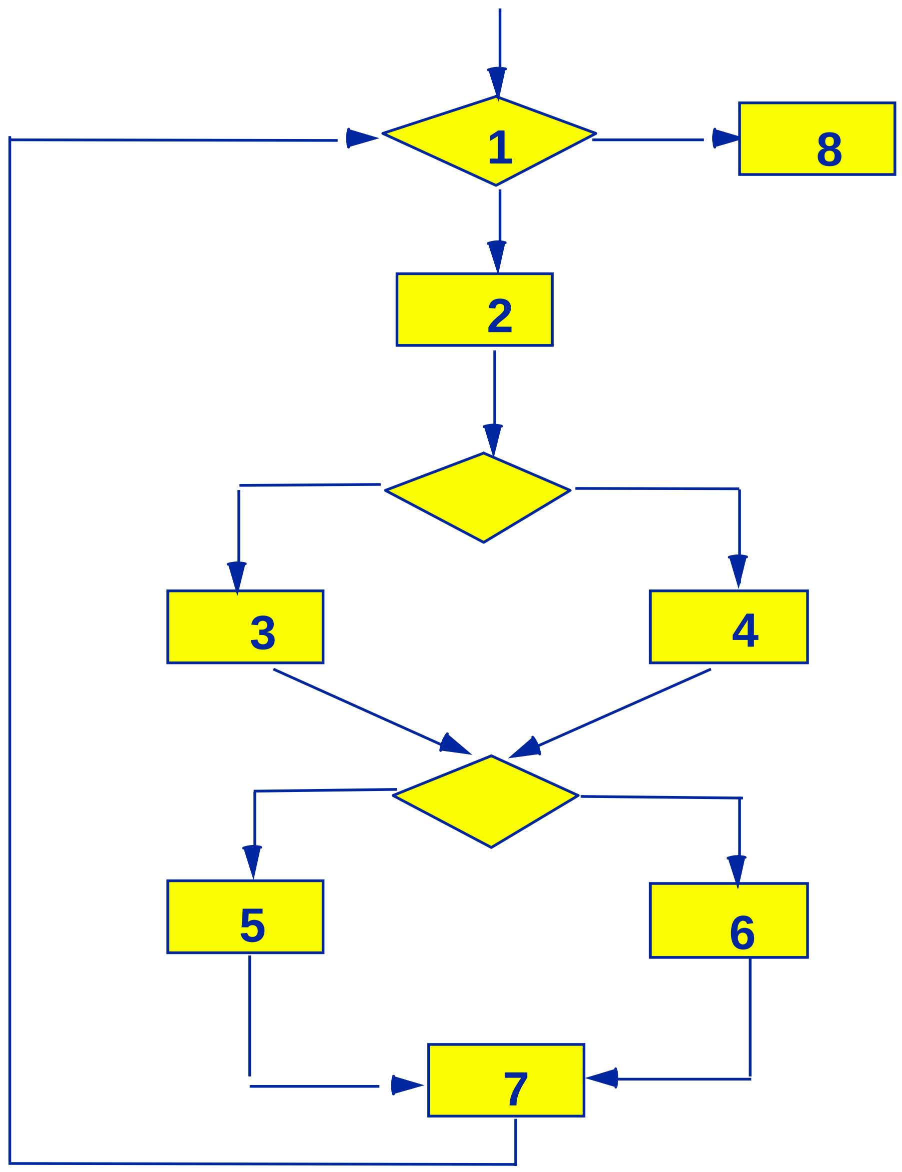
Paths for Example

Boundary paths

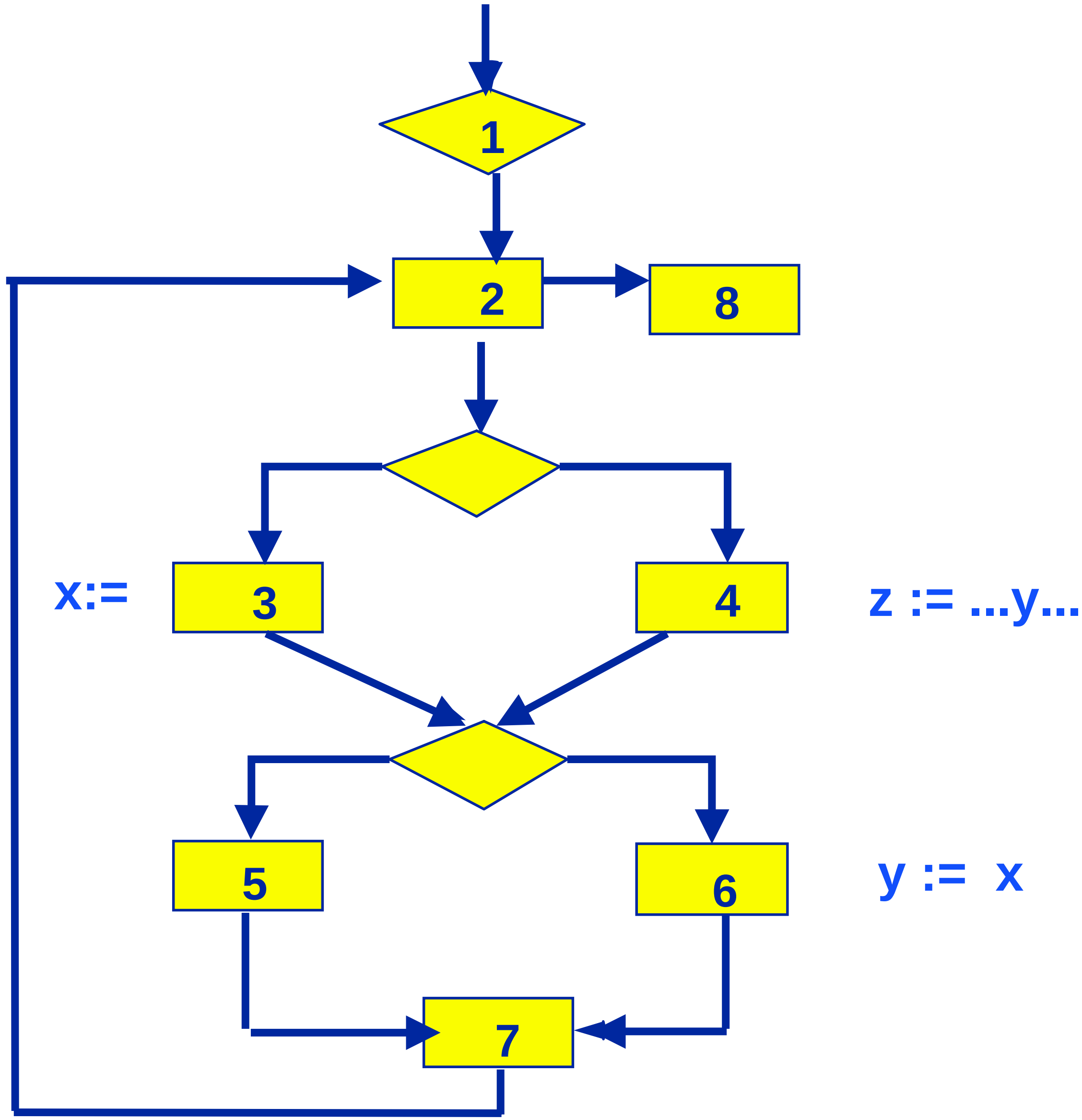
1,2,3,5,7	a
1,2,3,6,7	b
1,2,4,5,7	c
1,2,4,6,7	d

Interior paths
(for 2 executions of the loop)

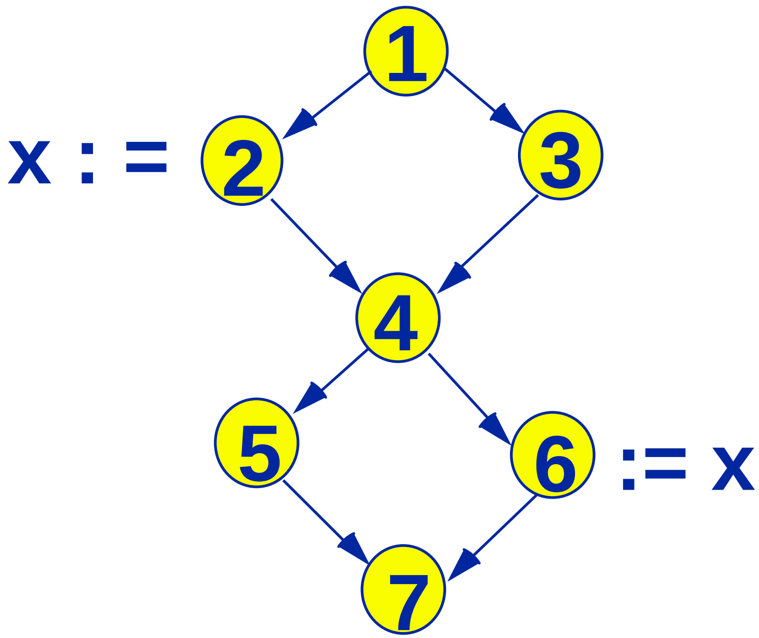
a,a
a,b
a,c
a,d
b,a
b,b
...
x,y for x,y = a, b, c, d



Need Control Flow AND Data Dependence



Non-looping Path Selection Problem



All branches 1, 2, 4, 5, 7
1, 3, 4, 6, 7

does not exercise the relationship between the definition of X in statement 2 and the reference to X in statement 6.

Definitions

- $d_n(x)$ denotes that variable x is assigned a value at node n (defined)
- $u_m(y)$ denotes that variable y is used (referenced at node m)
  - a definition clear path p with respect to (wrt) x is a subpath where x is not defined at any of the nodes in p
  - a definition $d_m(x)$ reaches a use $u_n(x)$ iff there is a subpath (m) • p • (n) such that p is definition clear wrt x

Data Flow Path Selection

- Rapps and Weyuker
  - definition-clear subpaths from definitions to uses
- Ntafos
  - chains of alternating definitions and uses linked by definition-clear subpaths
- Laski and Korel
  - combinations of definitions that reach uses at a node via a subpath

Assumptions

- no edges of the form (n, n_s) or $(n_f\ , n)$
- no edges of the form (n,n)
- there is at most one edge (m,n) for all m,n
- every control graph is well formed
  - Connected
  - Single start and single final node
- every loop has a single entry and a single exit

More assumptions

- at least one variable is associated with a node representing a predicate
- no variable definitions are associated with a node representing a predicate
- every definition of a variable reaches at least one use of that variable
- every use is reached by at least one definition
- every control graph contains at least one variable definition
- no variable uses or definitions are associated with n_s and n_f

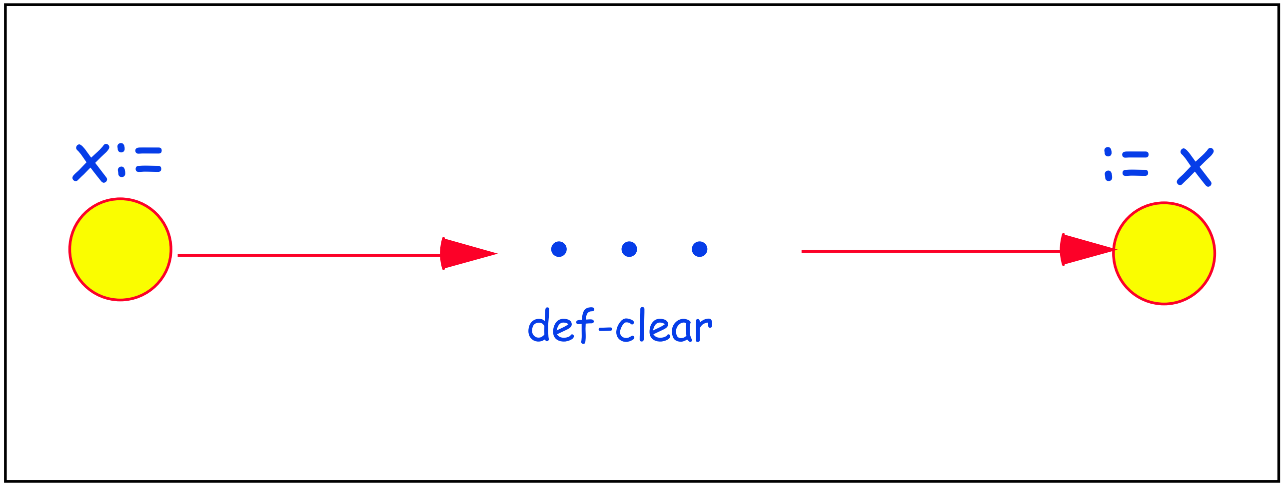
Rapps' and Weyuker's Data Flow Criteria

Foundation:

- Definition-clear subpaths from each definition to {some/all} use(s)

All-Defs

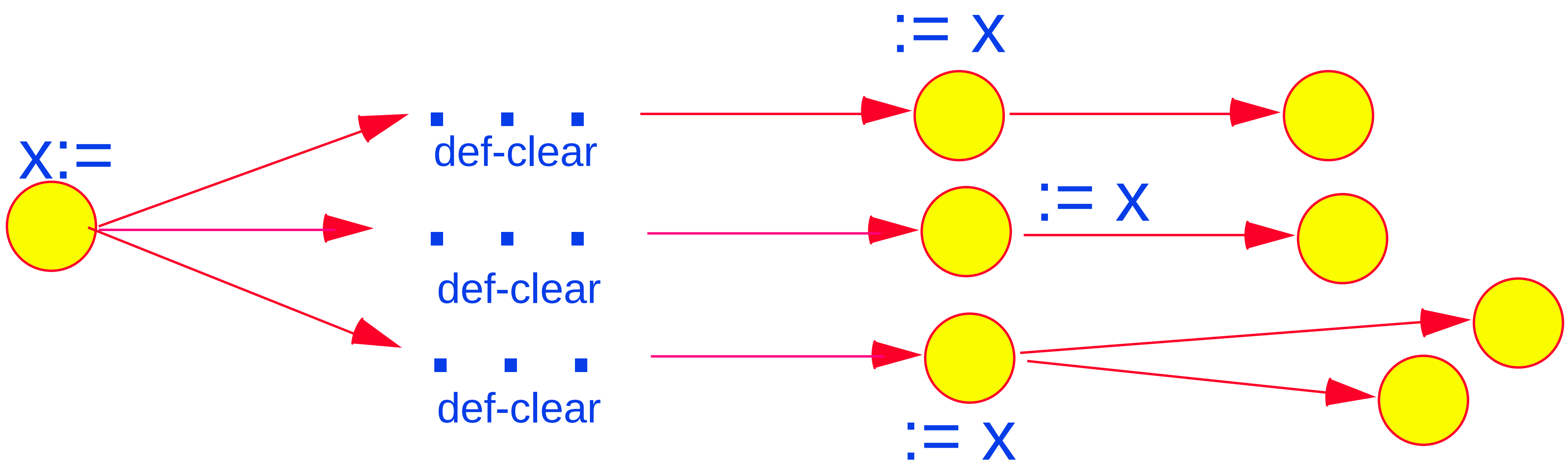
- Some definition-clear subpath from each definition to some use reached by that definition



Rapps' and Weyuker's Data Flow Criteria

All-Uses

- Some definition-clear subpath from each definition to each use reached by that definition and each successor node of the use



Rapps' and Weyuker's Data Flow Criteria

C-use is a "computation use"

P-use is a "predicate use"

All-C-Uses, Some-P-Uses

- either All-C-Uses for $d_m(x)$ or at least one P-Use

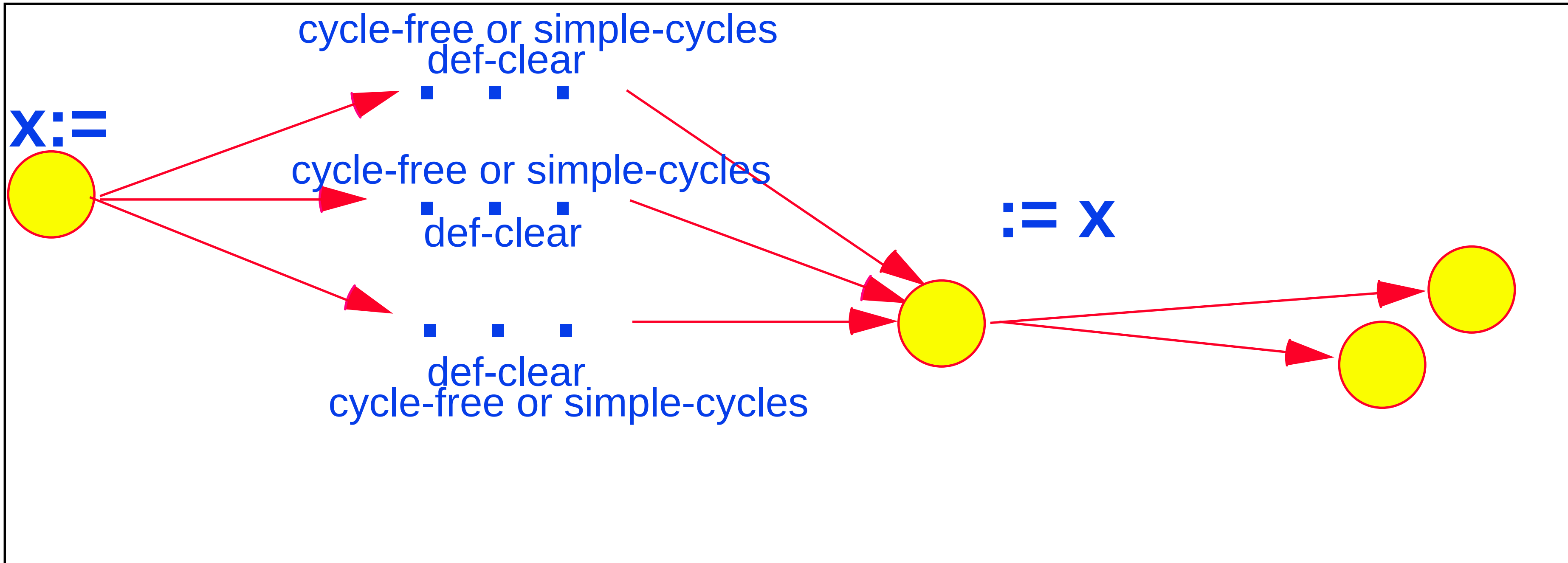
All-P-Uses, Some-C-Uses

- either All-P-Uses for $d_m(x)$ or at least one C-Use

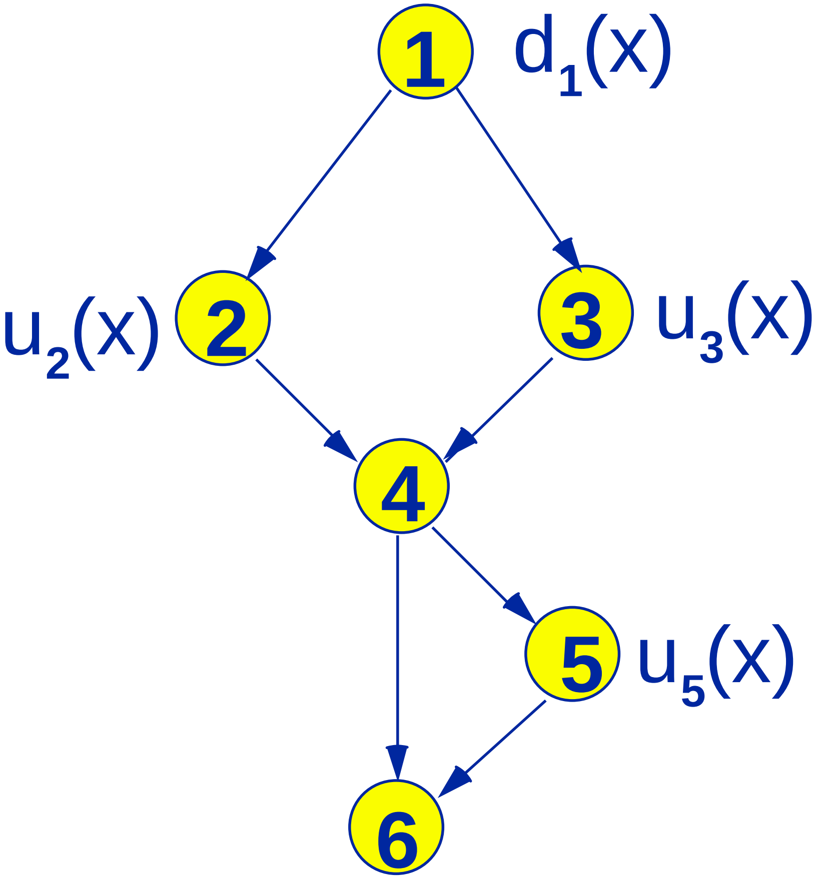
Rapps' and Weyuker's Data Flow Criteria

All-Du-Paths

- All definition-clear subpaths that are cycle-free or simple-cycles from each definition to each use reached by that definition and each successor node of the use



Example



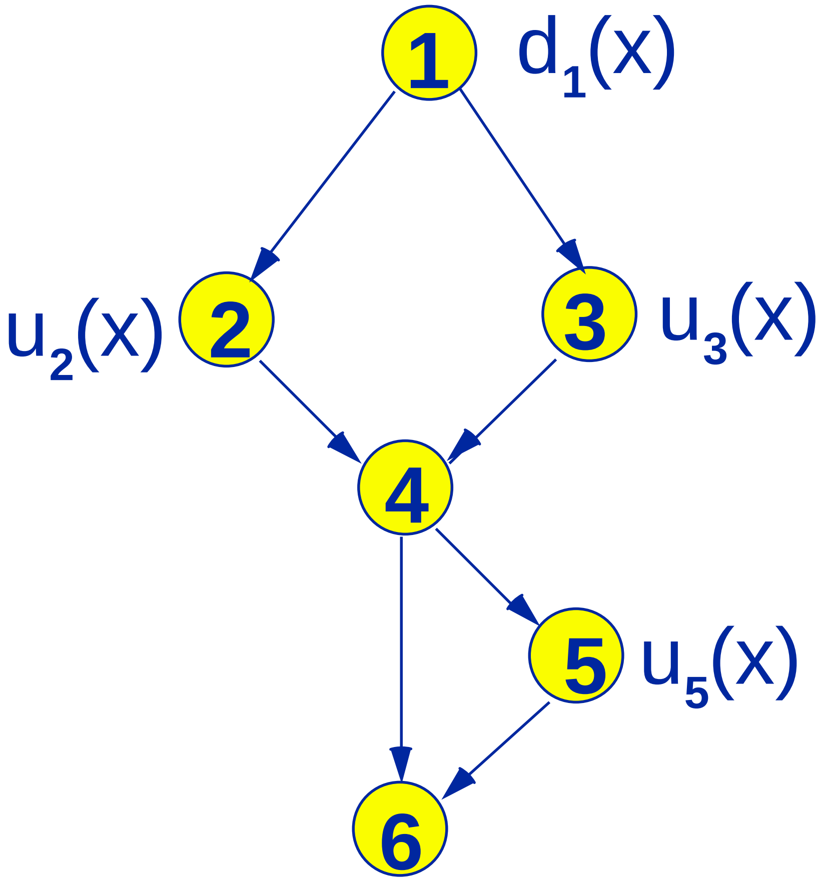
All-Defs

Requires:

$d_1(x)$ to a use

Satisfactory Path:

1, 2, 4, 6



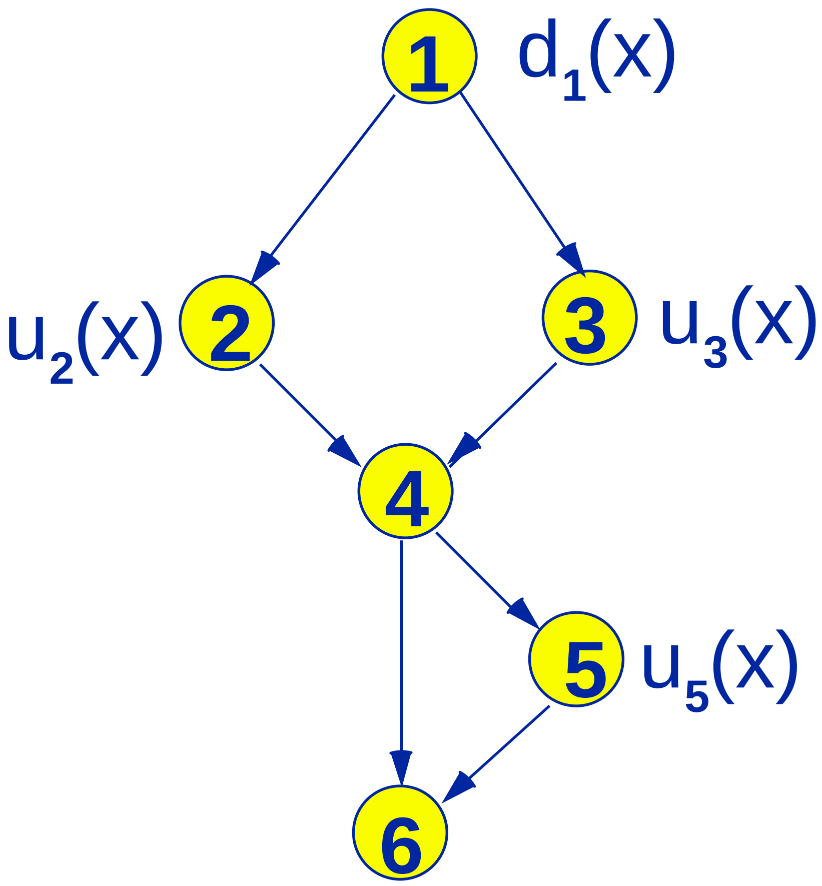
All-Uses

Requires:

- $d_1(x)$ to $u_2(x)$
- $d_1(x)$ to $u_3(x)$
- $d_1(x)$ to $u_5(x)$

Satisfactory Paths:

- 1, 2, 4, 5, 6
- 1, 3, 4, 6



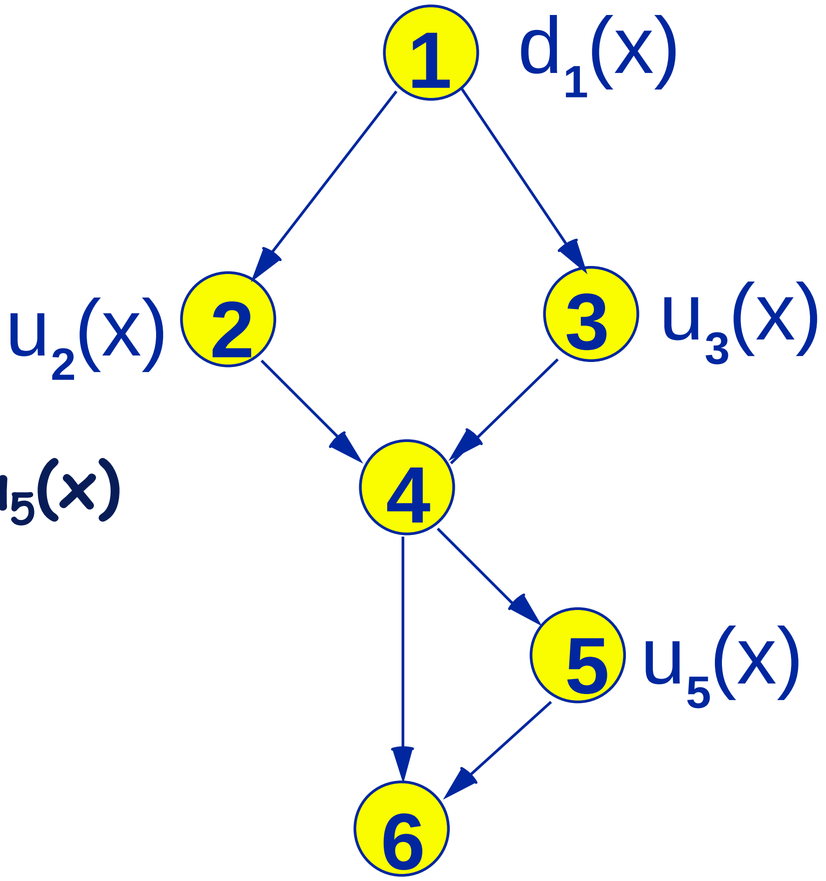
All-Du-Paths

Requires:

- $d_1(x)$ to $u_2(x)$
- $d_1(x)$ to $u_3(x)$
- both paths for $d_1(x)$ to $u_5(x)$

Satisfactory Paths:

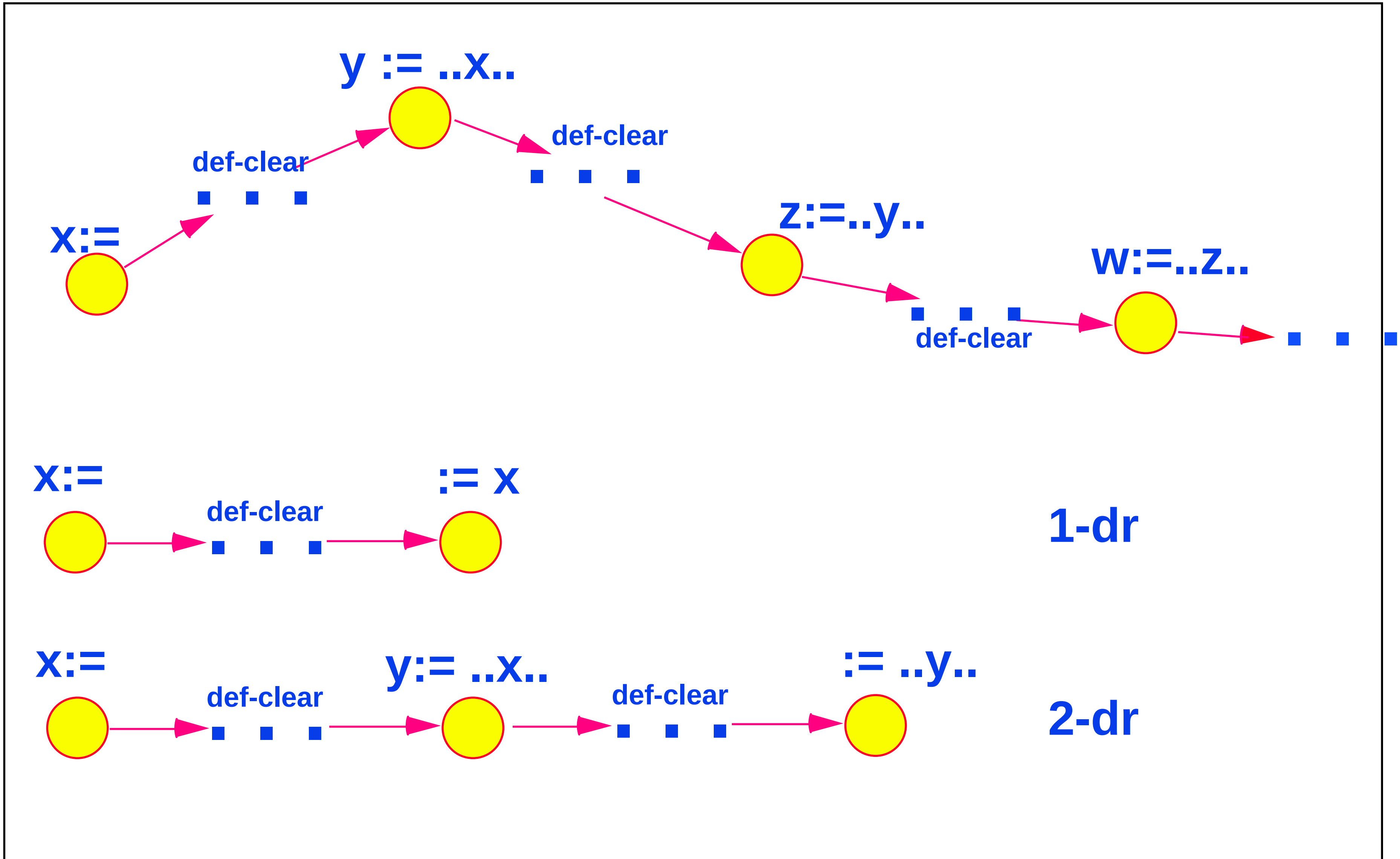
- 1, 2, 4, 5, 6
- 1, 3, 4, 5, 6



Ntafos' Data Flow Criteria

- Foundation:
  - Chains of alternating definitions and uses linked by definition-clear subpaths (k-dr interactions)
  - i^{th} definition reaches i^{th} use,
  - which defines $i^{th}+1$ definition
  - K is number of branches

k-dr interactions



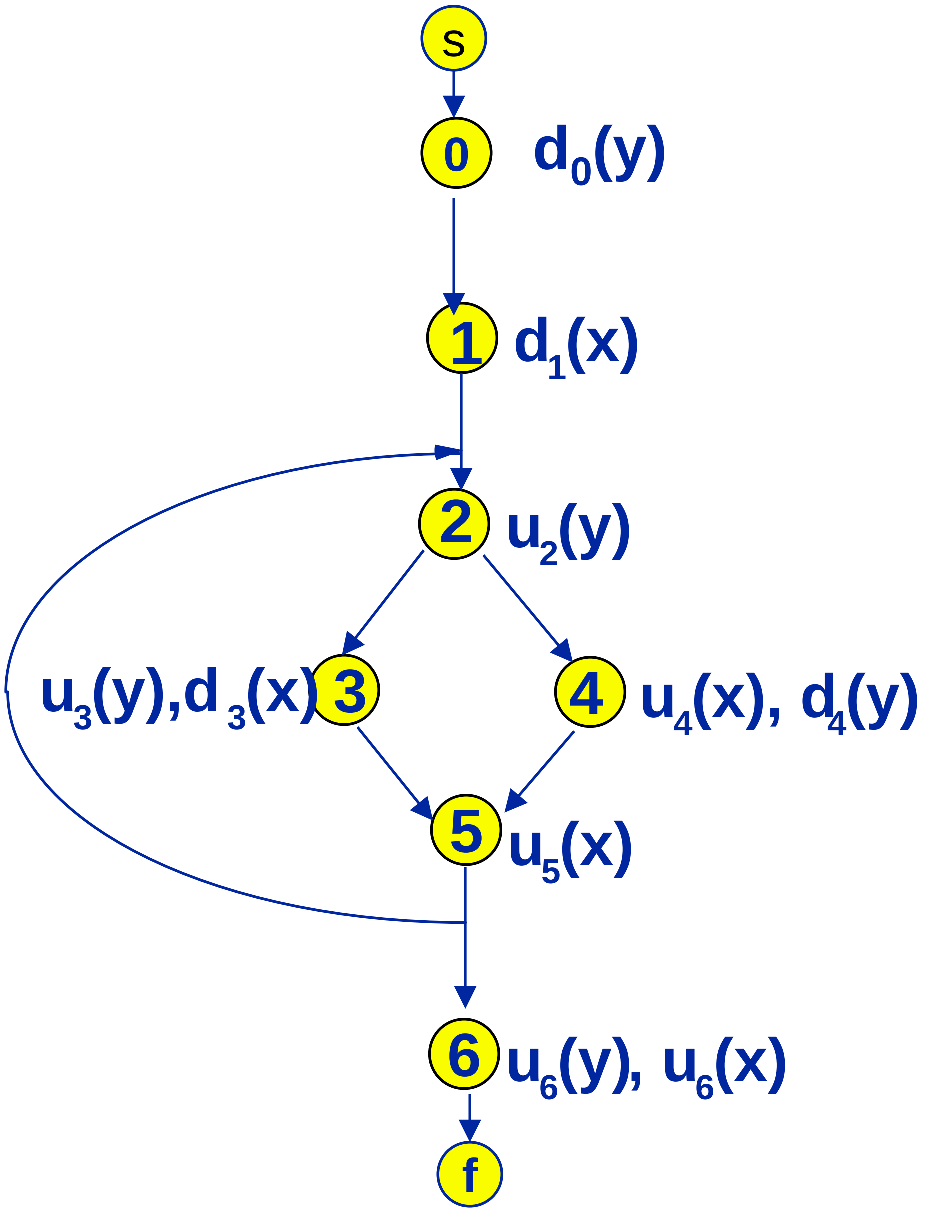
Ntafos' Data Flow Criteria

- Required K-tuples

  Some subpath propagating each k-dr interaction

  + if last use is a predicate, both branches
  + if first definition or last use is in a loop, minimal and some larger number of loop iterations

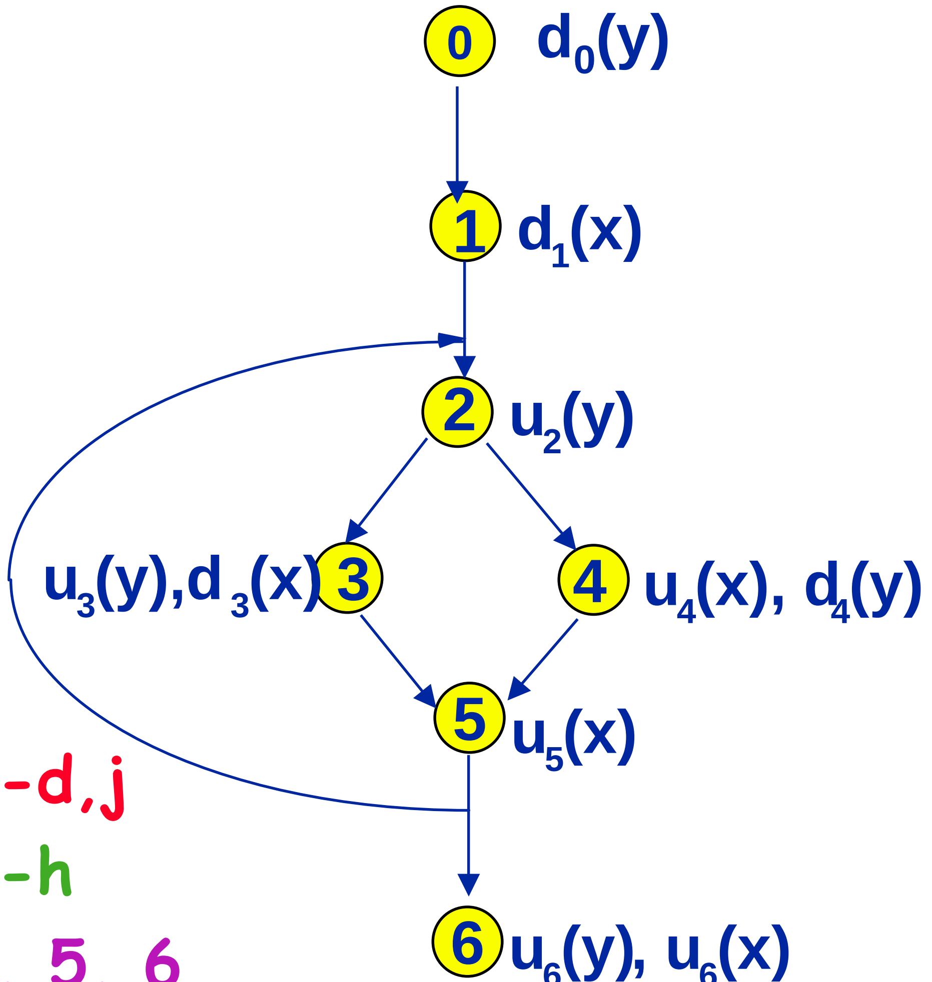
Example



1-DR interaction

a: d1(x) to u4(x)
b: d1(x) to u5(x)
c: d1(x) to u6(x)
d: d0(y) to u2(y)
e: d0(y) to u3(y)
f: d0(y) to u6(y)

g: d3(x) to u5(x)
h: d3(x) to u6(x)
i: d3(x) to u4(x)
j: d4(y) to u6(y)
k: d4(y) to u2(y)
l: d4(y) to u3(y)



PATHS

0,1, 2, 4, 5, 6: satisfies a-d,j

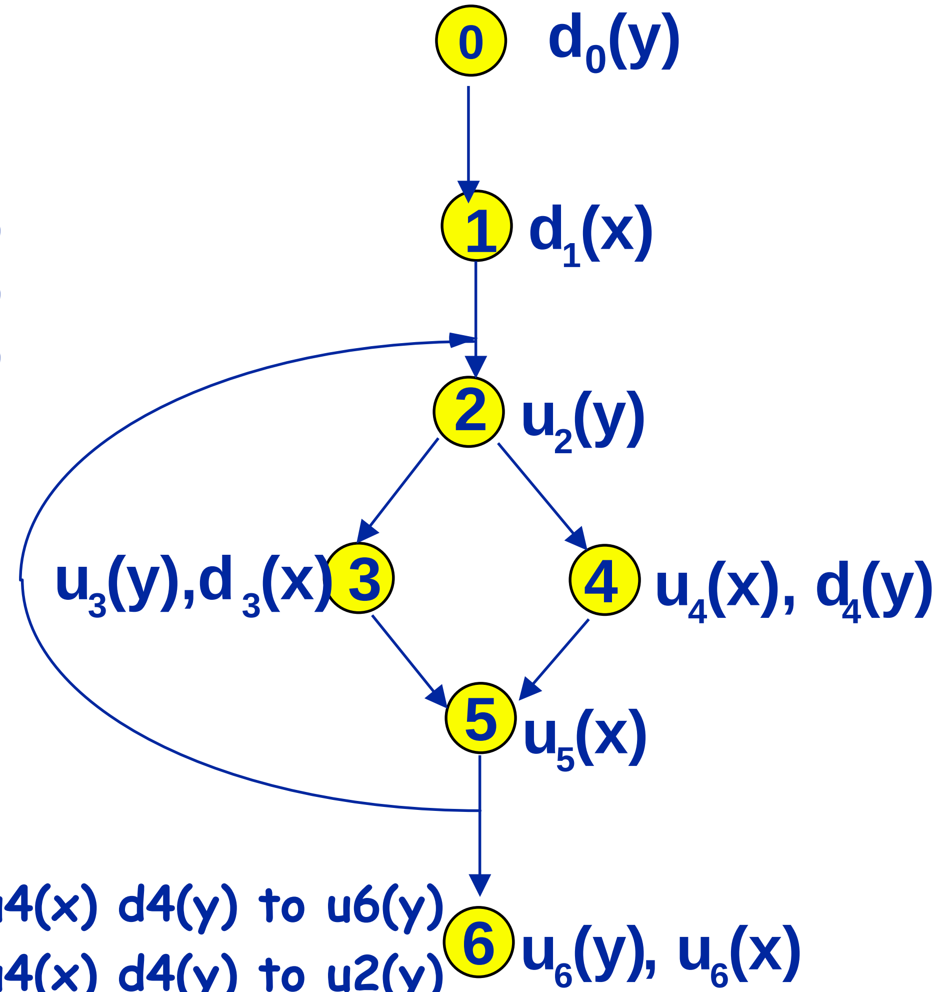
0,1, 2, 3, 5, 6: satisfies e-h

0,1, 2, 3, 5, 2, 4, 5, 2, 3, 5, 6 : satisfies i,k,l

From 1-DR to 2-DR

a: d1(x) to u4(x)
b: d1(x) to u5(x)
c: d1(x) to u6(x)
d: d0(y) to u2(y)
d0(y) to u3(y)
e: d0(y) to u6(y)
f:

g: d3(x) to u5(x)
h: d3(x) to u6(x)
i: d3(x) to u4(x)
j: d4(y) to u6(y)
d4(y) to u2(y)
k: d4(y) to u3(y)
l:



Plus:

d1(x) to u4(x) d4(y) to u6(y)
d1(x) to u4(x) d4(y) to u2(y)
d1(x) to u4(x) d4(y) to u3(y)
d0(y) to u3(y) d3(x) to u5(x)
d0(y) to u3(y) d3(x) to u6(x)
d0(y) to u3(y) d3(x) to u4(x)

d3(x) to u4(x) d4(y) to u6(y)
d3(x) to u4(x) d4(y) to u2(y)
d3(x) to u4(x) d4(y) to u3(y)
d4(y) to u3(y) d3(x) to u5(x)
d4(y) to u3(y) d3(x) to u6(x)
d4(y) to u3(y) d3(x) to u4(x)

2-DR interactions

aj: d1(x), u4(x), d4(y), u6(y)
ak: d1(x), u4(x), d4(y), u2(y)
al: d1(x), u4(x), d4(y), u3(y)
eg: d0(y), u3(y), d3(x), u5(x)
eh: d0(y), u3(y), d3(x), u6(x)
ei: d0(y), u3(y), d3(x), u4(x)
ij: d3(x), u4(x), d4(y), u6(y)
ik: d3(x), u4(x), d4(y), u2(y)
il: d3(x), u4(x), d4(y), u3(y)
lg: d4(y), u3(y), d3(x), u5(x)
lh: d4(y), u3(y), d3(x), u6(x)
li: d4(y), u3(y), d3(x), u4(x)

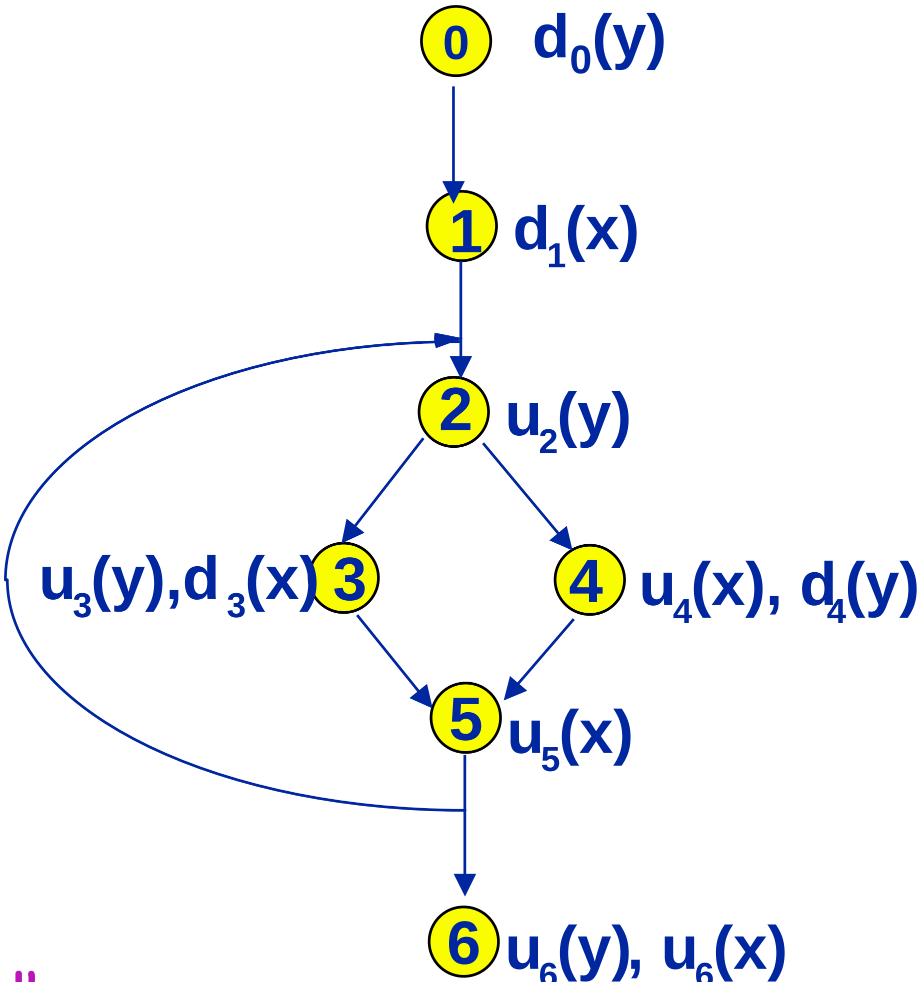
Paths:

0, 1, 2, 4, 5, 6: satisfies aj

0, 1, 2, 3, 5, 6: satisfies eg, eh

0, 1, 2, 3, 5, 2, 4, 5, 2, 3, 5, 6
: satisfies ei, ij, ik, il, lh

0, 1, 2, 4, 5, 2, 3, 5, 6: satisfies ak, al, lg (but not li)



Laski's and Korel's Criteria

- **Foundation:**

  Combinations of definitions that reach uses at some node via a subpath

- **Reach Coverage**

  Some definition-clear subpath from each definition to all uses reached by that definition

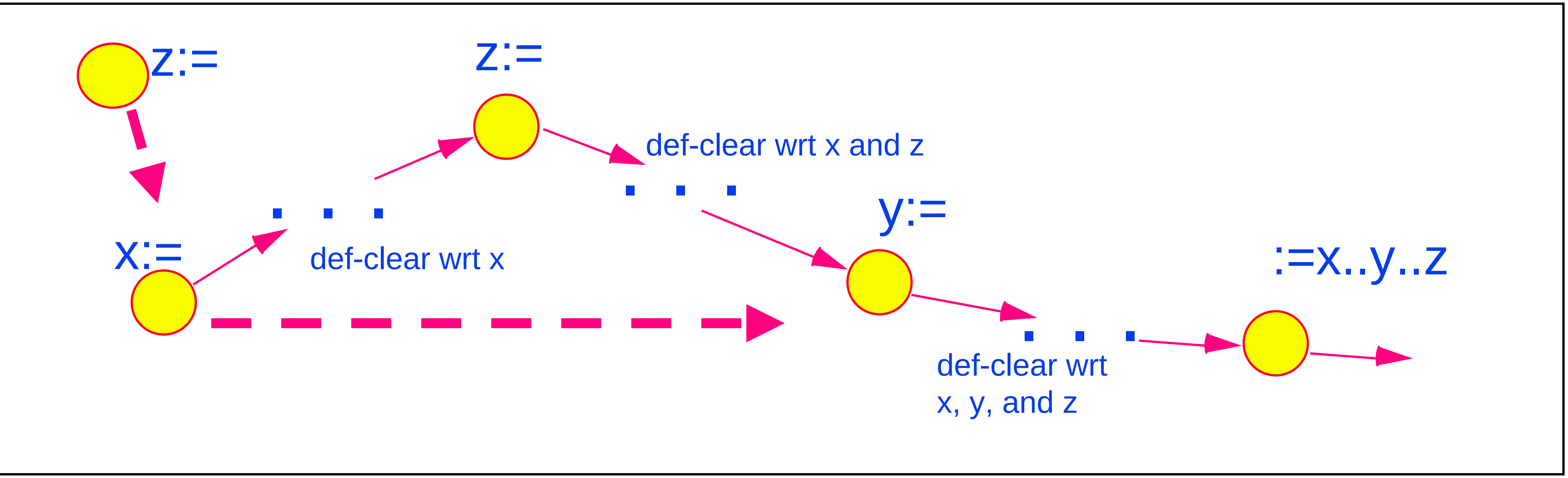
  basically the same as all-uses

Laski's and Korel's Criteria

- **Context Coverage**

  Some subpath along which each set of definitions reach uses at each node

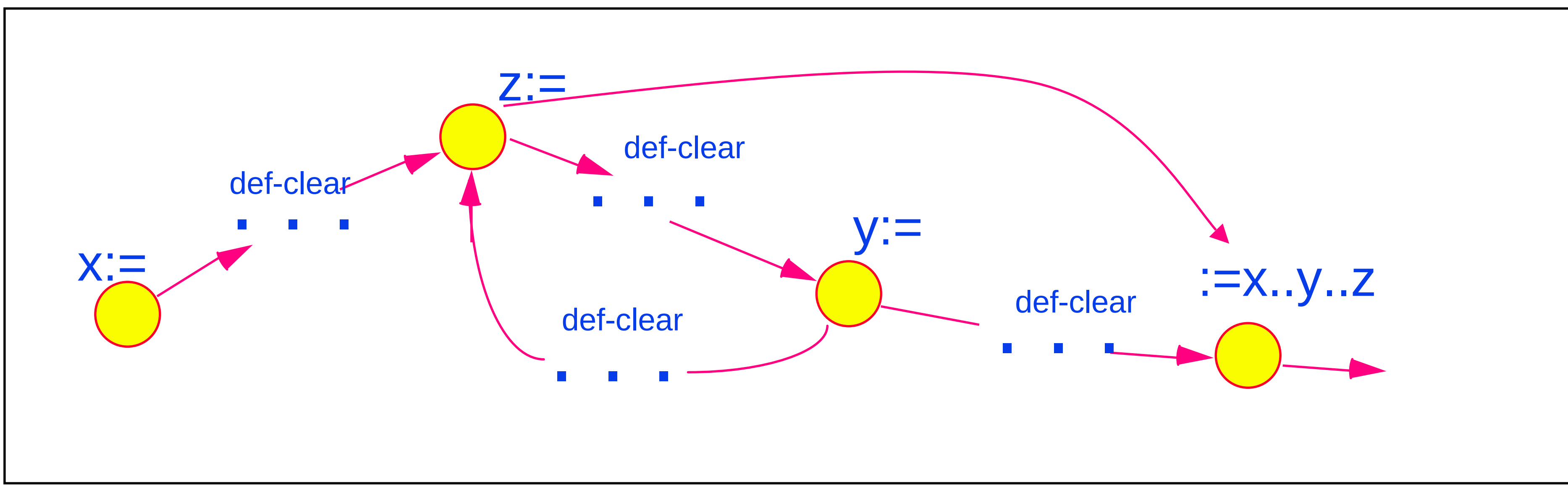
:=x..y..z



Laski's and Korel's Criteria

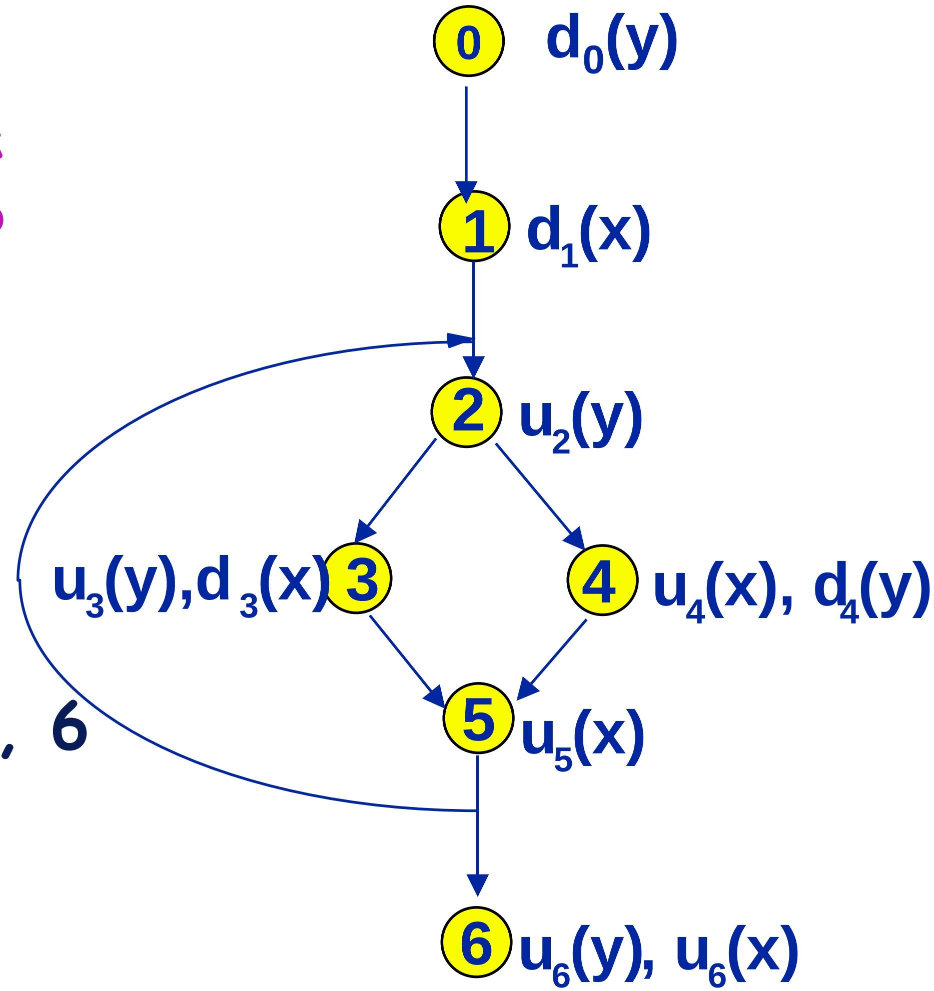
- **Ordered Context Coverage**

  Some subpath along which each sequence of definitions reach uses at each node



Context Coverage

$DC(n_6) = \{d_1(x),\ d_4(y)\},$ a
$\{d_3(x),\ d_0(y)\},$ b
$\{d_3(x),\ d_4(y)\}$ c



Paths

a: 1, 2, 4, 5, 6

b: 1, 2, 3, 5, 6

c: 1, 2, 3, 5, 2, 4, 5, 6

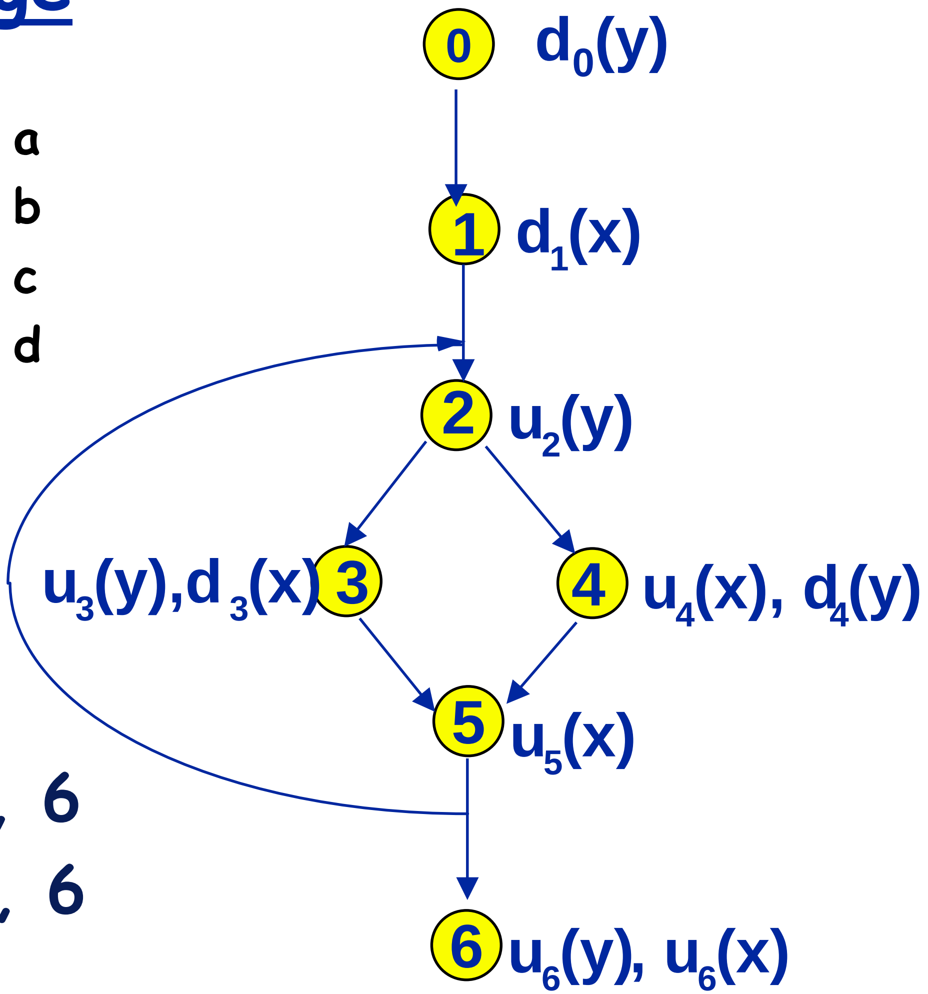
Note: must compute the sets for each node

Ordered Context Coverage

$ODC(n_6) = [d_1(x),\ d_4(y)],$ a
$[d_0(y),\ d_3(x)],$ b
$[d_3(x),\ d_4(y)],$ c
$[d_4(y),\ d_3(x)],$ d

Paths

a: 1, 2, 4, 5, 6
b: 1, 2, 3, 5, 6
c: 1, 2, 3, 5, 2, 4, 5, 6
d: 1, 2, 4, 5, 2, 3, 5, 6



Note: must compute the sequences for each node

How can we compare these criteria?

- all select a set of paths, so compare the paths that they select
  - set of paths that satisfy a criterion are not necessarily unique
  - e.g., s1 or s2 satisfies criterion A
    s1, s2, or s3 satisfy criterion B

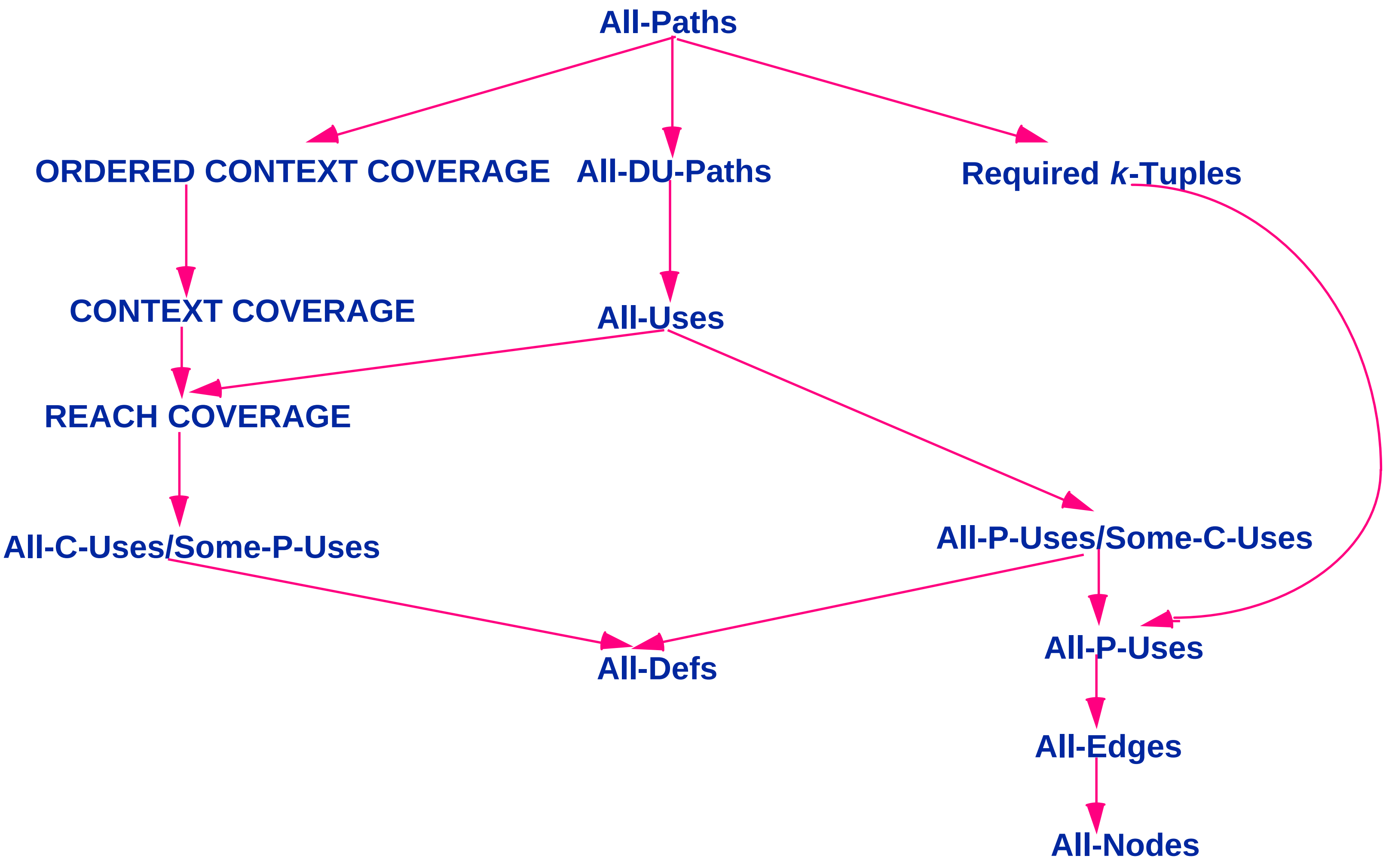
How can we compare these criteria?

- define a subsumption relationship
- criterion A subsumes criterion B iff for any flow graph

  P satisfies A ==> P satisfies B

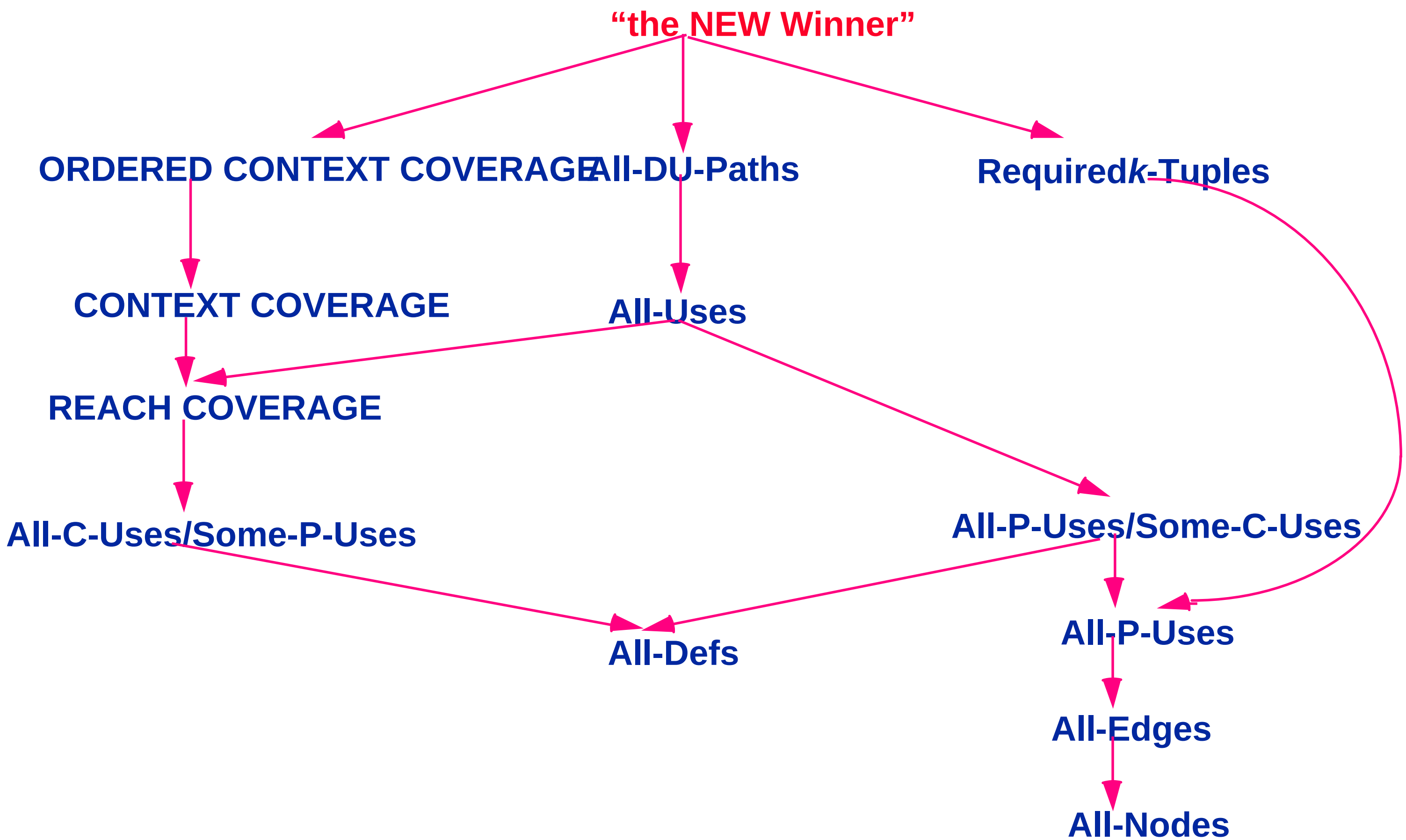
- criterion A is equivalent to criterion B iff A subsumes B and B subsumes A

Relationships among these criteria



Should we define yet another criteria?

- could subsume all the others, (except all paths)?



Problems with data flow coverage criteria

- infeasible paths
  - Don't usually get 100% coverage
- Need to understand fault detection ability
- Artificially combines control with data flow
  - Considering p-uses or all predicate alternatives, tacked on to incorporate control flow

Conclusions

- An improvement over control flow techniques
- Provides a rationale for how many times to iterate a loop or which combinations of subpaths to consider
- Most commonly used criterion is all-uses
- Need more empirical evidence to evaluate effectiveness