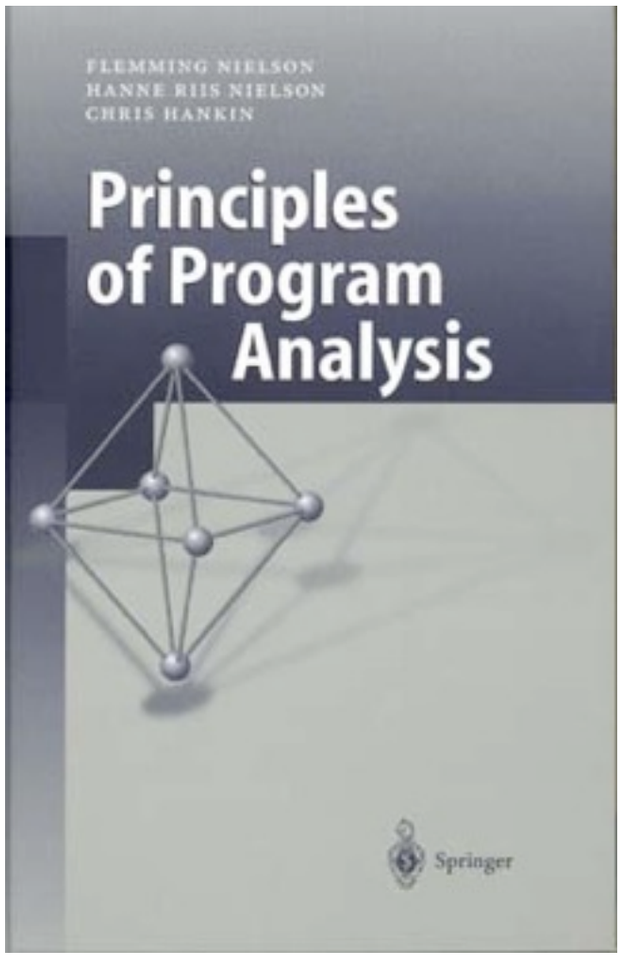
Calculating Graph Algorithms for Dominance and Shortest Path

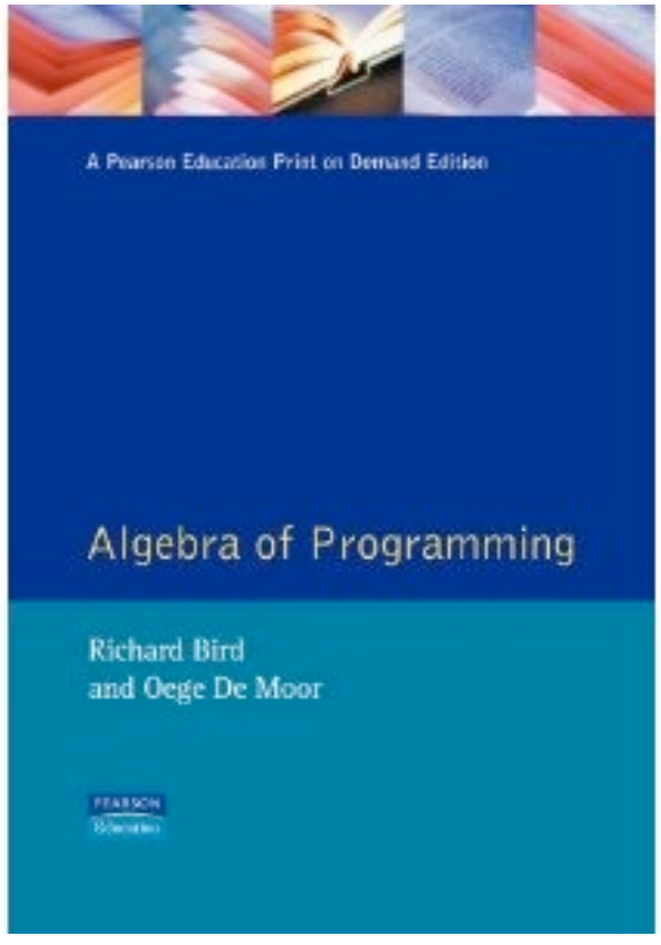
Ilya Sergey Jan Midtgaard Dave Clarke

MPC 2012

How to derive two graph algorithms by means of Abstract Interpretation


FLEMMING NIELSON
HANNE RIIS NIELSON
CHRIS HANKIN
Principles of Program Analysis
Springer

?


A Pearson Education Print on Demand Edition
Algebra of Programming
Richard Bird
and Oege De Moor
PEARSON

SYSTEMATIC DESIGN OF PROGRAM ANALYSIS FRAMEWORKS

Patrick Cousot* and Radhia Cousot**

Laboratoire d'Informatique,U.S.M.G., BP.53X
38041 Grenoble cedex, France

1. INTRODUCTION and SUMMARY

Semantic analysis of programs is essential in optimizing compilers and program verification systems. It encompasses data flow analysis, data type determination, generation of approximate invariant assertions, etc.

Several recent papers (among others Cousot & Cousot[77a], Graham & Wegman[76], Kam & Ullman[76], Kildall[73], Rosen[78], Tarjan[76], Wegbreit[75]) have introduced abstract approaches to program analysis which are tantamount to the use of a *program analysis framework* (A,t,Y) where A is a lattice of (approximate) assertions, t is an (approximate) predicate transformer and Y is an often implicit function specifying the meaning of the elements of A. This paper is devoted to the systematic and correct design of program analysis frameworks with respect to a formal semantics.

Preliminary definitions are given in Section 2 concerning the merge over all paths and (least) fixpoint program-wide analysis methods. In Section 3 we briefly define the (forward and backward) deductive semantics of programs which is later used as a formal basis in order to prove the correctness of the approximate program analysis frameworks. Section 4 very shortly recall the main elements of the lattice theoretic approach to approximate semantic analysis of programs.

In Section 6 we study and examplify various methods which can be used in order to define a space of approximate assertions or equivalently an approximation function. They include the characterization of the least Moore family containing an arbitrary set of assertions, the construction of the least closure operator greater than or equal to an arbitrary approximation function, the definition of closure operators by composition, the definition of a space of approximate assertions by means of a complete join congruence relation or by means of a family of principal ideals.

Section 7 is dedicated to the design of the approximate predicate transformer induced by a space of approximate assertions. First we look for a reasonable definition of the correctness of approximate predicate transformers and show that a local correctness condition can be given which has to be verified for every type of elementary statement. This local correctness condition ensures that the (merge over all paths or fixpoint) global analysis of any program is correct. Since isotony is not required for approximate predicate transformers to be correct it is shown that non-isotone program analysis frameworks are manageable although it is later argued that the isotony hypothesis is natural. We next show that among all possible approximate predicate transformers which can be used with a given space of approximate assertions there exists a best one which provides the maximum information relative to a program-wide analysis method. The best approximate predicate transformer

This paper is devoted to the systematic and correct design of program analysis frameworks with respect to a formal semantics.

Preliminary definitions are given in Section 2 concerning the merge over all paths and (least) fixpoint program-wide analysis methods. In Section 3 approximate program analysis frameworks. Section 4 very shortly recall the main elements of the lattice theoretic approach to approximate semantic analysis of programs.

Fixed-Point Calculus

Mathematics of Program Construction Group*

October 14, 1994

Abstract

The aim of this paper is to present a small calculus of extreme fixed points and to show it in action. The fixed-point theorem that was the main incentive for writing this paper is the fusion theorem presented in Section (3). It exploits the calculational properties of Galois connections.

1 Introduction

Solving equations is fundamental to computing. Yet, rules for doing so are seldomly explicitly taught or used, and certainly not in a calculational way. This paper summarizes a small selection of such rules and shows their use in a series of examples. Most results obtained in these applications are well-known; it is the method —purely equational reasoning— that is novel.

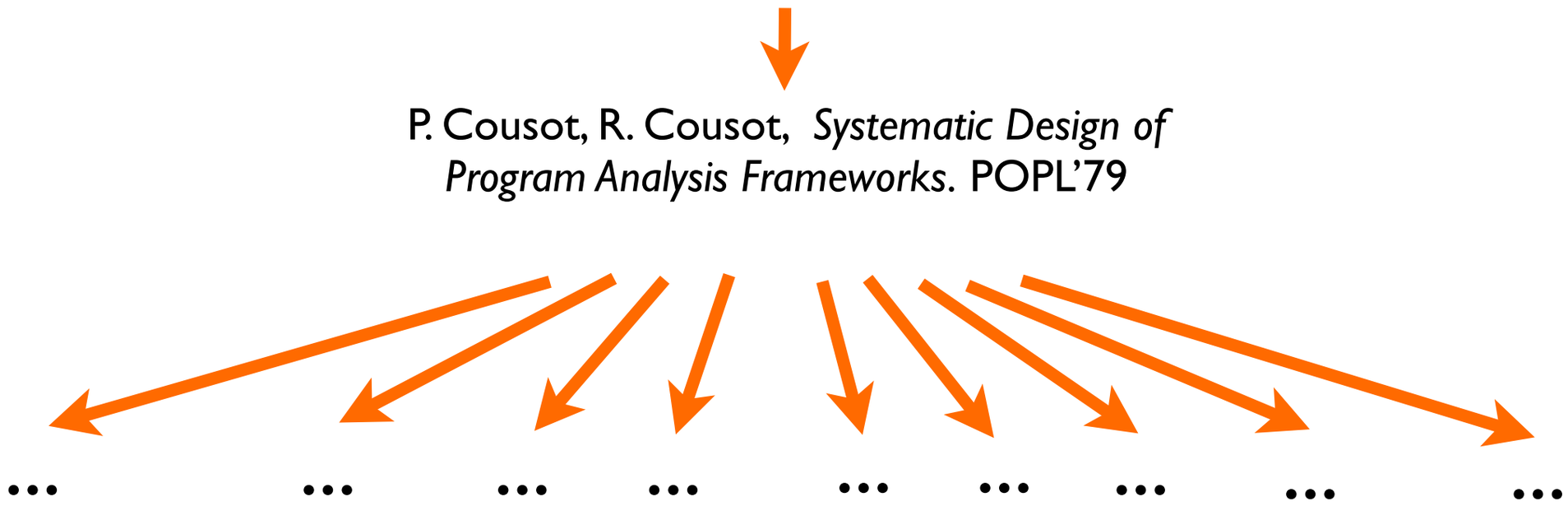
Our universes of discourse are complete lattices — in some applications augmented with a regular-algebra structure — and all functions considered are monotonic. We present a calculus of least fixed points; its counterpart for greatest fixed points follows by dualization.

Our universes of discourse are complete lattices — in some applications augmented with a regular-algebra structure — and all functions considered are monotonic. We present a calculus of least fixed points; its counterpart for greatest fixed points follows by dualization.

Abstract Interpretation

Abstract Interpretation
(a brief history)

P. Cousot, R. Cousot, *Abstract Interpretation: A Unified Lattice Model for Static Analysis of Programs by Construction or Approximation of Fixpoints*. POPL'77



Abstract Interpretation
(a brief list of applications in PL)

- strictness analysis
- type system design
- state reachability
- control-flow analysis
- information flow properties
- static error detection
- constant propagation
- escape analysis
- points-to analysis
- state reachability
- model checking
- data-flow optimizations
- loop detection
- CPR analysis

Abstract Interpretation

(an essence)

Not Interpretation as We Know It

Abstract Interpretation
(an essence)

- Partial orders and lattices
- Monotone functions and fixed points
- Galois connections

Partial Orders and Lattices

Complete lattice $\langle C; \sqsubseteq, \bot, \top, \sqcup, \sqcap \rangle$

Partial order $\langle C; \sqsubseteq \rangle$

Least upper bound $\sqcup$

Greatest lower bound $\sqcap$

Top $\top = \sqcup C$

Bottom $\bot = \sqcap C$

Monotone Functions

$$\langle C, \sqsubseteq \rangle \qquad \langle A, \leq \rangle$$

$$f : C \to A$$

$$[\ x \sqsubseteq y \implies f(x) \leq f(y)\ ]$$

Least Fixed Points

$$f : \langle C, \sqsubseteq \rangle \to \langle C, \sqsubseteq \rangle$$

$$\mathsf{lfp}_{\sqsubseteq} f = \sqcap \{x \mid f(x) \sqsubseteq x\}$$

Least Fixed Points

Kleene iteration

$$\bot \sqsubseteq f(\bot) \sqsubseteq f^2(\bot) \sqsubseteq f^3(\bot) \sqsubseteq \ldots$$

$$\mathsf{lfp}_{\sqsubseteq} f = \bigsqcup_{n \geq 0} f^n(\bot)$$

Semantics:

$$\langle C, \sqsubseteq \rangle, f : C \to C$$

$$f \text{ is monotone}$$

Interpretation:

$$\mathsf{lfp}_{\sqsubseteq} f$$

Concrete Interpretation

(an example)*

Partial trace collecting semantics

* P. Cousot, R. Cousot, *Basic concepts of abstract interpretation*

Partial trace collecting semantics

System states $s \in \Sigma$

Transition relation $t \subset \Sigma \times \Sigma$

Initial states $\Sigma_0 \subseteq \Sigma$

Partial trace $\sigma = s_0 s_1 \ldots s_n, s_i \in \Sigma$

Partial trace collecting semantics

Semantic functional

$$\mathcal{F}(X) = \{s \mid s \in \Sigma_0\} \cup \{\sigma s s' \mid \sigma s \in X \wedge \langle s, s' \rangle \in t\}$$

$$\mathcal{F} : \wp(\Sigma^+) \to \wp(\Sigma^+)$$

$\mathcal{F}$ is monotone

Partial trace collecting semantics

Lattice

$$\langle \wp(\Sigma^+), \subseteq, \emptyset, \Sigma^+, \cup, \cap \rangle$$

Interpretation

$$\mathsf{lfp}_{\subseteq} \mathcal{F} = \bigcup_{n \geq 0} \mathcal{F}^n(\emptyset)$$

Partial trace collecting semantics

(an example)

$$\Sigma = \{s_0, s_1, s_2\} \qquad \Sigma_0 = \{s_0\}$$

$$t = \{\langle s_0, s_1\rangle, \langle s_1, s_1\rangle, \langle s_1, s_2\rangle\}$$

$$\mathcal{F}^0(\emptyset) = \emptyset$$
$$\mathcal{F}^1(\emptyset) = \{s_0\}$$
$$\mathcal{F}^2(\emptyset) = \{s_0, s_0 s_1\}$$
$$\mathcal{F}^3(\emptyset) = \{s_0, s_0 s_1, s_0 s_1 s_2, s_0 s_1 s_1\}$$
$$\mathcal{F}^4(\emptyset) = \{s_0, s_0 s_1, s_0 s_1 s_2, s_0 s_1 s_1, s_0 s_1 s_1 s_1, s_0 s_1 s_1 s_2\}$$
$$\ldots$$

$$\mathsf{lfp}_{\subseteq}\,\mathcal{F}$$

is undecidable

$$\alpha\left(\mathbf{lfp}_{\sqsubseteq}\widehat{\mathcal{F}}\right)$$

is undecidable

α - property, “abstraction”

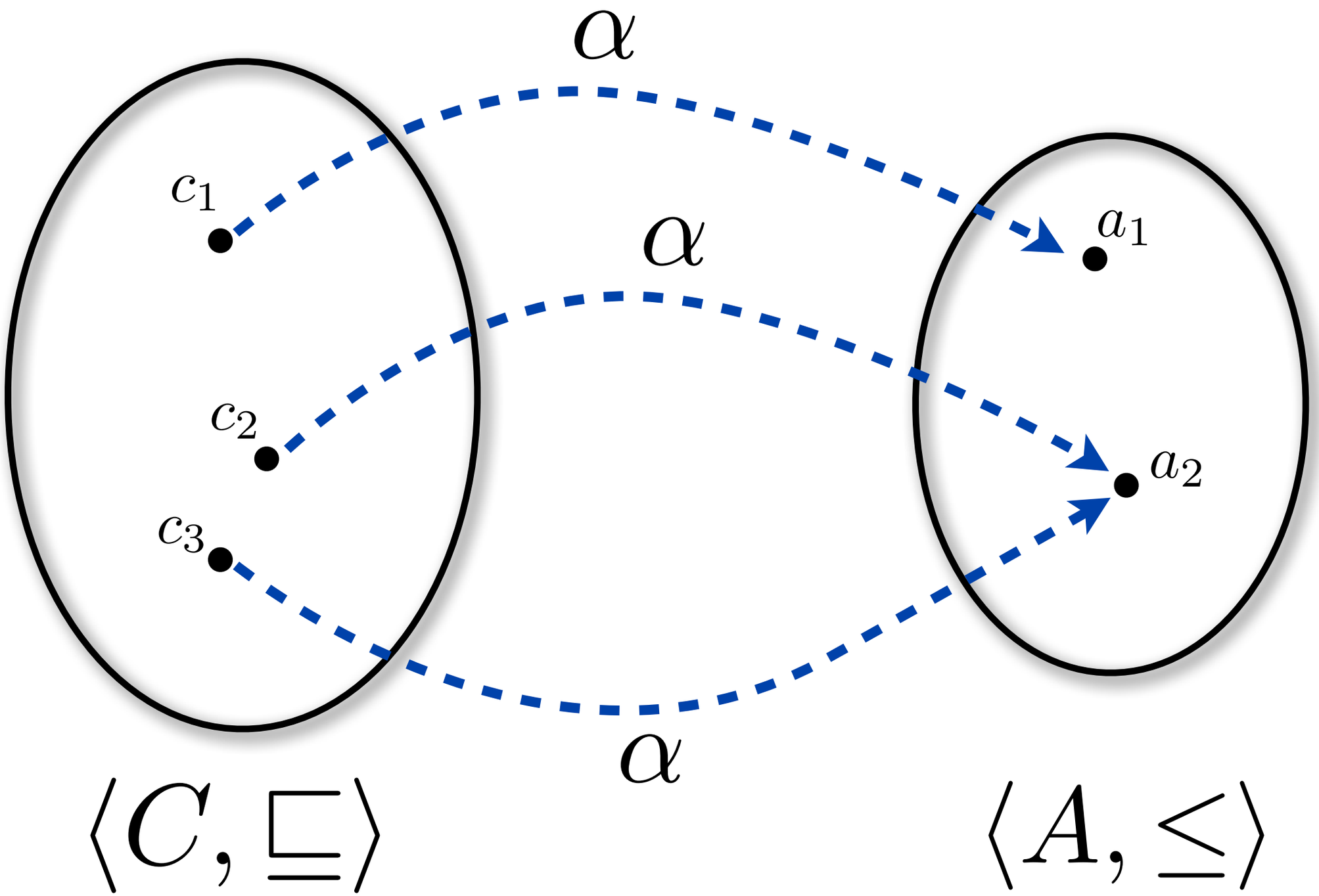
$\widehat{\mathcal{F}}$ is defined by α and $\mathcal{F}$

Abstraction

$$\alpha : \langle C, \sqsubseteq \rangle \to \langle A, \leq \rangle$$

α is monotone

Abstraction

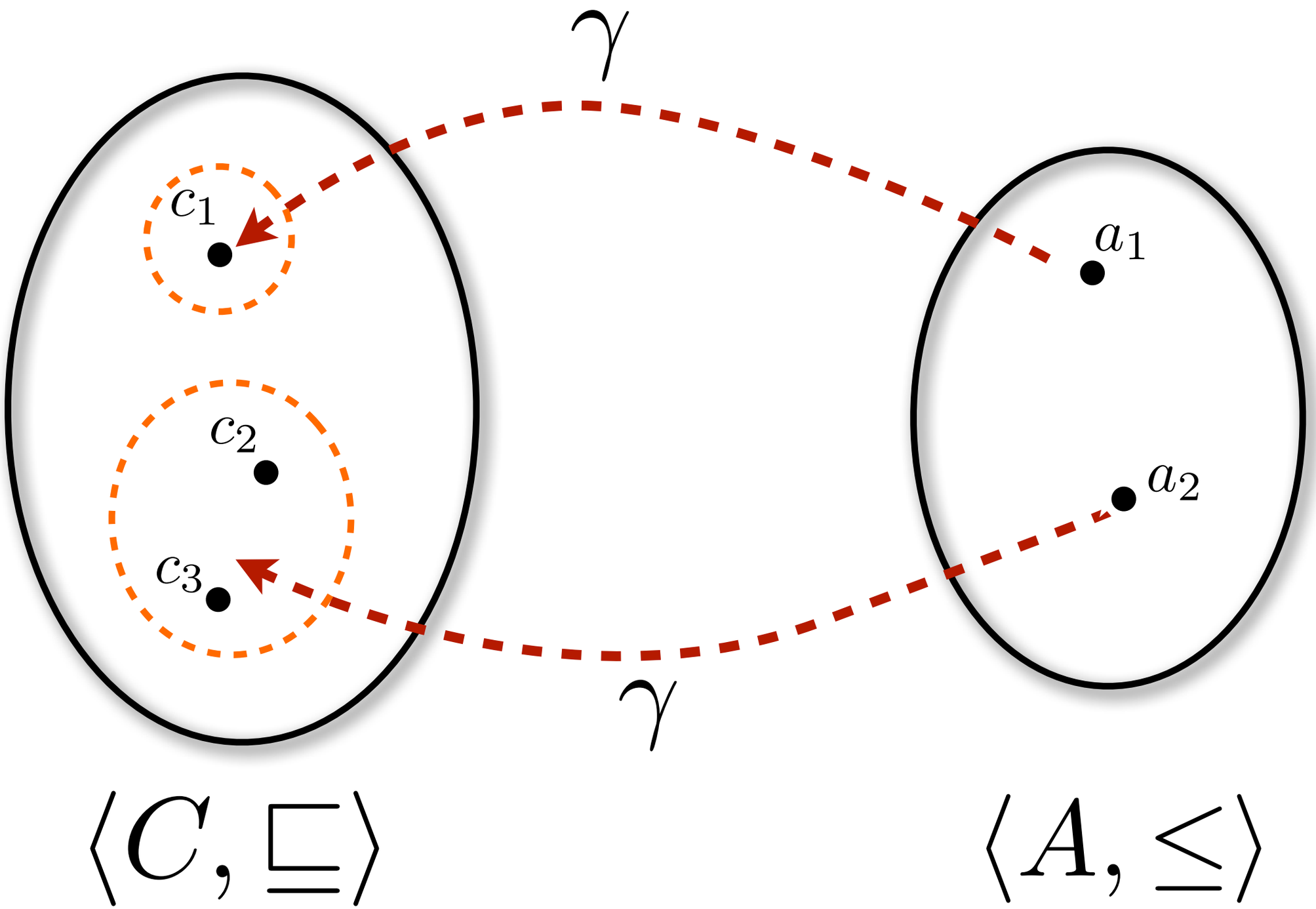


Concretization

$$\gamma : \langle A, \leq \rangle \rightarrow \langle C, \sqsubseteq \rangle$$

γ is monotone

Concretization



Galois Connections

$$\langle C, \sqsubseteq \rangle \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} \langle A, \leq \rangle$$

iff

$$[\ \alpha(c) \leq a \iff c \sqsubseteq \gamma(a)\ ]$$

Galois Connection Properties

Compositional construction

$$\langle C, \sqsubseteq \rangle \underset{\alpha_1}{\overset{\gamma_1}{\leftrightarrows}} \langle A_1, \leq_1 \rangle \qquad \langle A_1, \leq_1 \rangle \underset{\alpha_2}{\overset{\gamma_2}{\leftrightarrows}} \langle A_2, \leq_2 \rangle$$

$$\langle C, \sqsubseteq \rangle \underset{\alpha_2 \circ \alpha_1}{\overset{\gamma_1 \circ \gamma_2}{\leftrightarrows}} \langle A_2, \leq_2 \rangle$$

Galois Connection Properties

Fixed point fusion

$$F_c : \langle C, \sqsubseteq \rangle \to \langle C, \sqsubseteq \rangle \qquad F_a : \langle \underbrace{A, \leq}_{\text{usually finite}} \rangle \to \langle A, \leq \rangle$$

F_c, F_a **are monotone**

$$\alpha \circ F_c \dot{\leq} F_a \circ \alpha \implies \alpha(\mathsf{lfp}\ F_c) \leq \mathsf{lfp}\ F_a$$

$$\alpha \circ F_c = F_a \circ \alpha \implies \boxed{\alpha(\mathsf{lfp}\ F_c) = \mathsf{lfp}\ F_a}$$

$$\alpha(\mathsf{lfp}\ F_c) = \underbrace{\mathsf{lfp}\ F_a}_{\text{decidable}}$$

How to find an abstract functional

(a recipe)

$$\alpha \circ F_c$$

$$= \ldots$$

$$\equiv \ldots$$

$$\sqsubseteq \ldots$$

$$= F_a \circ \alpha$$

A property of trace-based interpretation

Partial trace collecting semantics
(abstracted)

$$\alpha(X) = \text{last states of all traces in } X$$

$$\alpha(X) = \{s \mid \sigma s \in X \text{ for some } \sigma\}$$

$$\underbrace{\langle \wp(\Sigma^+), \subseteq, \emptyset, \Sigma^+, \cup, \cap \rangle}_{\text{Partial traces}} \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} \underbrace{\langle \wp(\Sigma), \subseteq, \emptyset, \Sigma, \cup, \cap \rangle}_{\text{Reachable states}}$$

Partial trace collecting semantics
(abstracted)

$$\underbrace{\langle \wp(\Sigma^+), \subseteq, \emptyset, \Sigma^+, \cup, \cap \rangle}_{\text{Partial traces}} \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} \underbrace{\langle \wp(\Sigma), \subseteq, \emptyset, \Sigma, \cup, \cap \rangle}_{\text{Reachable states}}$$

$$\mathcal{F}(X) = \{s \mid s \in \Sigma_0\} \cup \{\sigma s s' \mid \sigma s \in X \wedge \langle s, s' \rangle \in t\}$$

$$\widehat{\mathcal{F}}(X) = \{s \mid s \in \Sigma_0\} \cup \{s' \mid s \in X \wedge \langle s, s' \rangle \in t\}$$

$$\alpha \circ \mathcal{F} = \widehat{\mathcal{F}} \circ \alpha^* \Longrightarrow \boxed{\alpha(\mathsf{lfp}_{\subseteq} \mathcal{F}) = \mathsf{lfp}_{\subseteq} \widehat{\mathcal{F}}}$$

* Proof: P. Cousot, R. Cousot, *Basic concepts of abstract interpretation*

Reachable states semantics

(an example)

$$\Sigma = \{s_0, s_1, s_2\} \qquad \Sigma_0 = \{s_0\}$$

$$t = \{\langle s_0, s_1\rangle, \langle s_1, s_1\rangle, \langle s_1, s_2\rangle\}$$

$$\widehat{\mathcal{F}}^0(\emptyset) = \emptyset$$

$$\widehat{\mathcal{F}}^1(\emptyset) = \{s_0\}$$

$$\widehat{\mathcal{F}}^2(\emptyset) = \{s_0, s_1\}$$

$$\widehat{\mathcal{F}}^3(\emptyset) = \{s_0, s_1, s_2\}$$

$$\widehat{\mathcal{F}}^4(\emptyset) = \{s_0, s_1, s_2\} = \widehat{\mathcal{F}}^3(\emptyset) = \mathsf{lfp}_{\subseteq}(\widehat{\mathcal{F}})$$

Graphs

Directed Rooted Graph

$$G = \langle V, E, v_0 \rangle$$

Edges: $E \subseteq V \times V$

Root node: $v_0 \in V$

Paths in Rooted Graphs

$$(u \to v) \iff \langle u, v \rangle \in E$$

Finite non-empty paths

$$\sigma \in V^+$$

$$\sigma = u_0 \ldots u_n \quad \forall i \in 1 \ldots n, (u_{i-1} \to u_i)$$

Enumerating All Paths

A finite path functional

$$p_G : \wp(V^+) \to \wp(V^+)$$

$$p_G(\mathcal{X}) = \{\sigma, v : \sigma \in \mathcal{X} \wedge (\texttt{last}(\sigma) \to v) : \sigma v\}$$

$$\texttt{last} : V^+ \to V$$

$$\texttt{last}(\sigma u) = u$$

All Paths in a Fixpoint Form

A lattice $\langle \wp(V^+), \subseteq \rangle$

$$\mathrm{P}_G = \mathsf{lfp}(\lambda \mathcal{X}.\{v_0\} \cup p_G(\mathcal{X}))$$

Dominance

A node u dominates node v if u belongs to *every* path from the initial node v_0 to v.

Preliminary Ideas

- Dominance is a property of *all* paths in a graph
- Dominance is a relation on graph nodes
  - Dominance is a finite (i.e., computable) property

Goal

Compute a dominance algorithm from the definition of dominance

Formal Definition of Dominance

$$\texttt{dom} : \wp(V^+) \to \wp(V \times V)$$

$$[\ u\ \texttt{dom}(\mathcal{X})\ v = \langle \forall \sigma : \sigma \in \mathcal{X} \wedge \texttt{last}(\sigma) = v : u\ \texttt{in}\ \sigma \rangle\ ]$$

where

$$\texttt{in} \subseteq V \times V^+$$
$$\texttt{in} = \mathsf{lfp}(\lambda \mathcal{X}.\texttt{last} \cup \mathcal{X} \circ \texttt{pre})$$

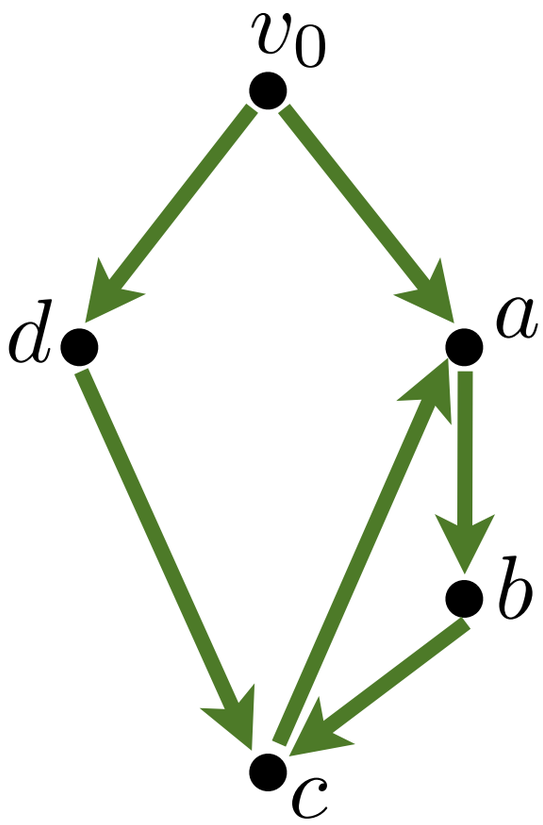
and

$$\texttt{pre} \subseteq V^+ \times V^+$$
$$\texttt{pre} = \{\sigma, v : \sigma \in V^+ \wedge \sigma v \in V^+ : \langle \sigma, \sigma v \rangle\}$$

Formal Definition of Dominance

$$[\ u\ \overbrace{\mathtt{dom}(\mathcal{X})}^{\mathcal{X}\text{ is a set of paths}}\ v = \langle \underbrace{\forall \sigma : \sigma \in \mathcal{X}}_{\text{for every}\quad \sigma \in X} \wedge \overbrace{\mathtt{last}(\sigma) = v}^{v\text{ is a last node of }\sigma} : \underbrace{u\ \mathtt{in}\ \sigma}_{u \in \sigma} \rangle\ ]$$

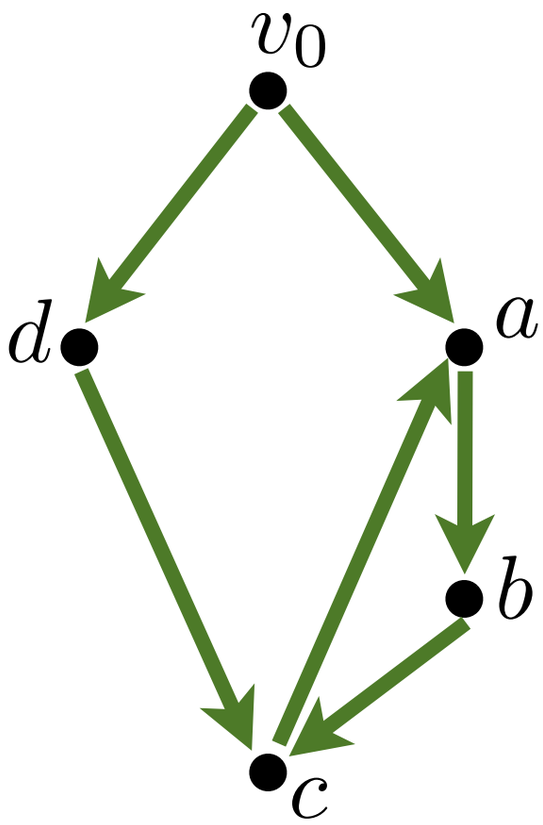
Dominance in a Graph



$$p_G^0(\emptyset) = \emptyset$$

$$\begin{aligned}\texttt{dom}\left(p_G^0(\emptyset)\right) = \{&\langle v_0, v_0\rangle, \langle v_0, a\rangle, \langle v_0, b\rangle, \langle v_0, c\rangle, \langle v_0, d\rangle,\\ &\langle a, v_0\rangle, \langle a, a\rangle, \langle a, b\rangle, \langle a, c\rangle, \langle a, d\rangle,\\ &\langle b, v_0\rangle, \langle b, a\rangle, \langle b, b\rangle, \langle b, c\rangle, \langle b, d\rangle,\\ &\langle c, v_0\rangle, \langle c, a\rangle, \langle c, b\rangle, \langle c, c\rangle, \langle c, d\rangle,\\ &\langle d, v_0\rangle, \langle d, a\rangle, \langle d, b\rangle, \langle d, c\rangle, \langle d, d\rangle\}\end{aligned}$$

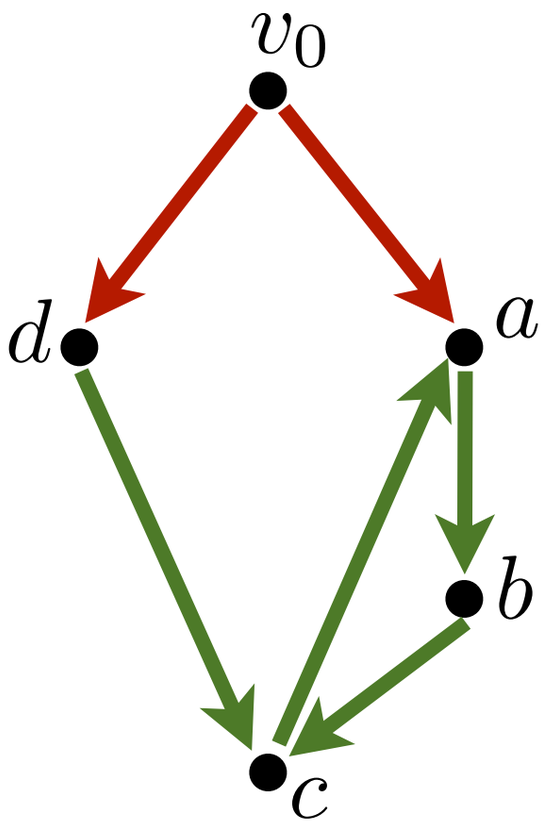
Dominance in a Graph



$$p_G^1(\emptyset) = \{v_0\}$$

$$\begin{aligned}\texttt{dom}\left(p_G^1(\emptyset)\right) = \{&\langle v_0, v_0\rangle, \langle v_0, a\rangle, \langle v_0, b\rangle, \langle v_0, c\rangle, \langle v_0, d\rangle,\\ &\langle a, a\rangle, \langle a, b\rangle, \langle a, c\rangle, \langle a, d\rangle,\\ &\langle b, a\rangle, \langle b, b\rangle, \langle b, c\rangle, \langle b, d\rangle,\\ &\langle c, a\rangle, \langle c, b\rangle, \langle c, c\rangle, \langle c, d\rangle,\\ &\langle d, a\rangle, \langle d, b\rangle, \langle d, c\rangle, \langle d, d\rangle\}\end{aligned}$$

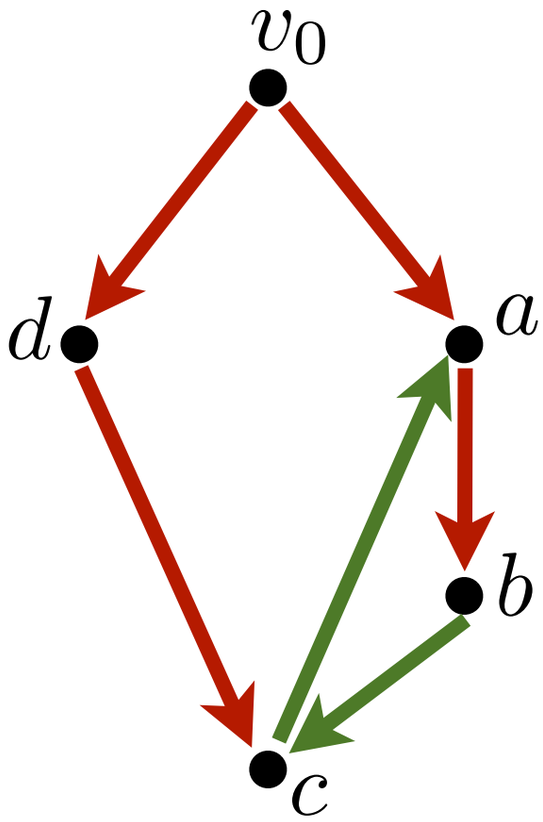
Dominance in a Graph



$$p_G^2(\emptyset) = \{v_0, v_0a, v_0d\}$$

$$\texttt{dom}\left(p_G^2(\emptyset)\right) = \{\langle v_0, v_0\rangle, \langle v_0, a\rangle, \langle v_0, b\rangle, \langle v_0, c\rangle, \langle v_0, d\rangle,$$
$$\langle a, a\rangle, \langle a, b\rangle, \langle a, c\rangle,$$
$$\langle b, b\rangle, \langle b, c\rangle,$$
$$\langle c, b\rangle, \langle c, c\rangle,$$
$$\langle d, b\rangle, \langle d, c\rangle, \langle d, d\rangle\}$$

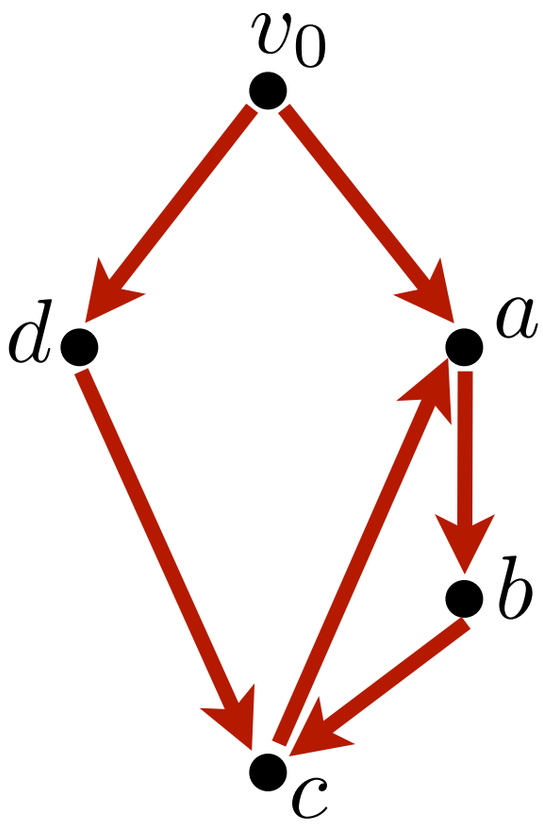
Dominance in a Graph



$$p_G^3(\emptyset) = \{v_0, v_0a, v_0d, v_0ab, v_0dc\}$$

$$\texttt{dom}\left(p_G^3(\emptyset)\right) = \{\langle v_0, v_0\rangle, \langle v_0, a\rangle, \langle v_0, b\rangle, \langle v_0, c\rangle, \langle v_0, d\rangle,$$
$$\langle a, a\rangle, \langle a, b\rangle,$$
$$\langle b, b\rangle,$$
$$\langle c, c\rangle,$$
$$\langle d, c\rangle, \langle d, d\rangle\}$$

Dominance in a Graph



$$p_G^4(\emptyset) = \{v_0, v_0a, v_0d, v_0ab, v_0dc, v_0abc, v_0dca\}$$

$$\texttt{dom}\left(p_G^4(\emptyset)\right) = \{\langle v_0, v_0\rangle, \langle v_0, a\rangle, \langle v_0, b\rangle, \langle v_0, c\rangle, \langle v_0, d\rangle,$$
$$\langle a, a\rangle, \langle a, b\rangle,$$
$$\langle b, b\rangle,$$
$$\langle c, c\rangle,$$
$$\langle d, d\rangle\}$$

A fixed point for dominance is reached

Idea

- As a set of paths increases, a dominance can only decrease
- A possible case to establish a Galois Connection

A Galois Connection for Dominance

$$\langle \underbrace{\wp(V^+)}, \subseteq \rangle \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} \langle \underbrace{\wp(V \times V)}, \supseteq \rangle$$

Finite paths

Dominance relations

What is α?

What about dom?

Recall:

$$\texttt{dom} : \wp(V^+) \to \wp(V \times V)$$

$$[\ u\ \texttt{dom}(\mathcal{X})\ v = \langle \forall \sigma : \sigma \in \mathcal{X} \wedge \texttt{last}(\sigma) = v : u\ \texttt{in}\ \sigma \rangle\ ]$$

Goals

1. Prove a Galois Connection **To do**

$$\langle \wp(V^+), \subseteq \rangle \underset{\mathtt{dom}}{\overset{\overline{\mathtt{dom}}}{\leftrightarrows}} \langle \wp(V \times V), \supseteq \rangle$$

2. Find $\mathscr{F}_{\mathscr{D}}$ such that

$$\mathtt{dom} \circ p_G = \mathscr{F}_{\mathscr{D}} \circ \mathtt{dom}$$

then

$$\mathtt{dom}(\mathrm{P}_G) = \mathsf{lfp}_{\supseteq}(\lambda \mathscr{X}.\mathtt{dom}(\{v_0\}) \cap \mathscr{F}_{\mathscr{D}}(\mathscr{X}))$$

1. A Galois Connection for Dominance

Decomposing Dominance

$$\mathtt{dom} = g \circ f$$

Establish a connection compositionally

Intermezzo:

Compositions and Factors*

*With gratitude to Reviewer #1

Compositions

$$R \subseteq A \times B \qquad S \subseteq B \times C$$

$$R \circ S \equiv \{x, y, z : \langle x, y \rangle \in R \wedge \langle y, z \rangle \in S : \langle x, z \rangle\}$$

Factors

$$R \subseteq A \times B \qquad S \subseteq B \times C \qquad T \subseteq A \times C$$

Left factor:

$$[\ x\ T/S\ y \equiv \langle \forall z : y\ S\ z : x\ T\ z \rangle\ ]$$

Right factor:

$$[\ x\ R\backslash T\ y \equiv \langle \forall z : z\ R\ x : z\ T\ y \rangle\ ]$$

A Nice Thing about Factors

$$R \subseteq A \times B \qquad S \subseteq B \times C \qquad T \subseteq A \times C$$

$$[\ \alpha(S) \leq\ R \iff\ S \sqsubseteq\ \gamma(R)\ ]$$

$$[\ T/S \supseteq R \iff\ S \subseteq R \backslash T\ ]$$

$$\alpha(\mathcal{X}) = T/\mathcal{X} \qquad\qquad \gamma(\mathcal{X}) = \mathcal{X}\backslash T$$

A Nice Thing about Factors

$$R \subseteq A \times B \qquad S \subseteq B \times C \qquad T \subseteq A \times C$$

$$(T/) = \lambda\mathcal{X}.\, T/\mathcal{X} \qquad (\backslash T) = \lambda\mathcal{X}.\, \mathcal{X}\backslash T$$

$$\left[\, \alpha(S) \leq R \iff S \sqsubseteq \gamma(R) \,\right]$$

$$\left[\, T/S \supseteq R \iff S \subseteq R\backslash T \,\right]$$

$$\langle \wp(B \times C), \subseteq \rangle \underset{(T/)}{\overset{(\backslash T)}{\leftrightarrows}} \langle \wp(A \times B), \supseteq \rangle$$

$$\alpha(\mathcal{X}) = T/\mathcal{X} \qquad \gamma(\mathcal{X}) = \mathcal{X}\backslash T$$

Back to Dominance

Decomposing Dominance

$$[u \ \mathtt{dom}(\mathcal{X}) \ v = \underbrace{\langle \forall \sigma : \sigma \in \mathcal{X} \wedge f(\mathcal{X}) \mathtt{last}(\sigma) = v : u \ \mathtt{in} \ \sigma \rangle}_{\mathtt{in}/f(\mathcal{X})}]$$

$$f(\mathcal{X}) = \{\sigma : \sigma \in \mathcal{X} : \langle \mathtt{last}(\sigma), \sigma \rangle\}$$

Decomposing Dominance

$$[u\ \mathtt{dom}(\mathcal{X})\ v = \underbrace{\langle \forall \sigma \mathtt{dom} = (\mathtt{in}/f(\mathcal{X})\ \sigma : u\ \mathtt{in}\ \sigma \rangle}_{\mathtt{in}/f(\mathcal{X})}]$$

$$f(\mathcal{X}) = \{\sigma : \sigma \in \mathcal{X} : \langle \mathtt{last}(\sigma), \sigma \rangle\}$$

Decomposing Dominance

$$\mathtt{dom} = (\mathtt{in}/) \circ f$$

Factors

$$\langle \wp(\mathtt{last}), \subseteq \rangle \underset{(\mathtt{in}/)}{\overset{(\backslash \mathtt{in})}{\leftrightarrows}} \langle \wp(V \times V), \supseteq \rangle$$

$$f(\mathcal{X}) = \{\sigma : \sigma \in \mathcal{X}\}$$

In the paper

$$\langle \wp(V^+), \subseteq \rangle \underset{\mathtt{last}}{\overset{\overline{f}}{\leftrightarrows}} \langle \wp(\mathtt{last}), \subseteq \rangle$$

$$\overline{f}(\mathcal{X}) = \{\langle \mathtt{last}(\sigma), \sigma \rangle : \sigma \in \mathcal{X}\}$$

$$\langle \wp(V^+), \subseteq \rangle \underset{\mathtt{dom}}{\overset{\overline{\mathtt{dom}}}{\leftrightarrows}} \langle \wp(V \times V), \supseteq \rangle$$

Goals

1. Prove a Galois Connection **Done**

$$\langle \wp(V^+), \subseteq \rangle \underset{\mathtt{dom}}{\overset{\overline{\mathtt{dom}}}{\leftrightarrows}} \langle \wp(V \times V), \supseteq \rangle$$

2. Find $\mathcal{F}_{\mathscr{D}}$ such that **To do**

$$\mathtt{dom} \circ p_G = \mathcal{F}_{\mathscr{D}} \circ \mathtt{dom}$$

2. Computing a Dominance Functional

$$\mathtt{dom} \circ p_G = (\mathtt{in}/) \circ f \circ p_G$$

$$k(\mathcal{X}) = \{\sigma, u, v : \langle u, \sigma \rangle \in \mathcal{X} \wedge (u \to v) : \langle v, \sigma v \rangle\}$$

$$f \circ p_G = k \circ f$$

$$\mathtt{dom} \circ p_G = (\mathtt{in}/) \circ \not{k} \circ \not{p}_G$$

$$k(\mathcal{X}) = \{\sigma, u, v : \langle u, \sigma \rangle \in \mathcal{X} \wedge (u \to v) : \langle v, \sigma v \rangle\}$$

$$f \circ p_G = k \circ f$$

Details in the paper

$$\mathtt{dom} \circ p_G = (\mathtt{in}/) \circ k \circ f$$

$$k(\mathcal{X}) = \{\sigma, u, v : \langle u, \sigma \rangle \in \mathcal{X} \wedge (u \Longrightarrow v) : \langle v, \sigma v \rangle\}$$

$$\mathcal{F}_{\mathscr{D}}(\mathcal{X}) = \mathtt{id} \uplus \mathcal{X}/\mathtt{pred}$$

$$\left[\begin{array}{c} v \ \mathtt{pred}\ u \equiv u \longrightarrow v \\ f \circ p_G = k \circ f \end{array}\right]$$

$$(\mathtt{in}/) \circ k = \mathcal{F}_{\mathscr{D}} \circ (\mathtt{in}/)$$

Details in the paper

$$\mathtt{dom} \circ p_G = (\mathtt{fin}/) \circ (\mathtt{in}/) \circ f \circ f$$

$$\mathcal{F}_{\mathscr{D}}(\mathcal{X}) = \mathtt{id} \cup \mathcal{X}/\mathtt{pred}$$

$$[\ v\ \mathtt{pred}\ u \equiv u \longrightarrow v\ ]$$

$$(\mathtt{in}/) \circ k = \mathcal{F}_{\mathscr{D}} \circ (\mathtt{in}/)$$

Details in the paper

$$\mathtt{dom} \circ p_G = \mathcal{F}_{\mathscr{D}} \circ (\mathtt{dom}/) \circ f$$

$$\mathcal{F}_{\mathscr{D}}(\mathcal{X}) = \mathtt{id} \cup \mathcal{X}/\mathtt{pred}$$

$$[\ v\ \mathtt{pred}\ u \equiv u \longrightarrow v\ ]$$

$$(\mathtt{in}/) \circ k = \mathcal{F}_{\mathscr{D}} \circ (\mathtt{in}/)$$

Details in the paper

$$\mathtt{dom} \circ p_G = \mathcal{F}_{\mathscr{D}} \circ \mathtt{dom}$$

$$\mathcal{F}_{\mathscr{D}}(\mathcal{X}) = \mathtt{id} \cup \mathcal{X}/\mathtt{pred}$$

$$\left[\; v \;\mathtt{pred}\; u \equiv u \longrightarrow v \;\right]$$

$$\left(\mathtt{in}/\right) \circ k \Downarrow \mathcal{F}_{\mathscr{D}} \circ \left(\mathtt{in}/\right)$$

$$\mathtt{dom}(\mathrm{P}_G) = \mathsf{lfp}_{\supseteq}(\lambda\mathcal{X}.\mathtt{dom}(\{v_0\}) \cap \mathcal{F}_{\mathscr{D}}(\mathcal{X}))$$

Goals

1. Prove a Galois Connection **Done**

$$\langle \wp(V^+), \subseteq \rangle \underset{\mathtt{dom}}{\overset{\overline{\mathtt{dom}}}{\leftrightarrows}} \langle \wp(V \times V), \supseteq \rangle$$

2. Find $\mathcal{F}_{\mathscr{D}}$ such that **Done**

$$\mathtt{dom} \circ p_G = \mathcal{F}_{\mathscr{D}} \circ \mathtt{dom}$$

$$\mathtt{dom}(\mathrm{P}_G) = \mathsf{lfp}_{\supset}(\lambda\mathcal{X}.\mathtt{dom}(\{v_0\}) \cap \mathcal{F}_{\mathscr{D}}(\mathcal{X}))$$

Algorithm

$$\mathrm{Dom}(v) = \{u : u \ \mathtt{dom}(\mathtt{P}_G) \ v : u\}$$

A Straightforward Algorithm

$$\bot \left\{ \begin{array}{l} \textbf{for } v \in V \textbf{ do} \\ \quad \mathrm{Dom}[v] \leftarrow V \end{array} \right.$$

$$\texttt{dom}(\{v_0\}) \sqcup \mathsf{lfp}(\ldots) \left\{ \begin{array}{l} \mathrm{Dom'} \leftarrow \texttt{dom}(\{v_0\}) \cap \mathcal{F}_{\mathscr{D}}(\mathrm{Dom}) \\ \textbf{while } \mathrm{Dom} \neq \mathrm{Dom'} \textbf{ do} \\ \quad \mathrm{Dom} \leftarrow \mathrm{Dom'} \\ \quad \mathrm{Dom'} \leftarrow \texttt{dom}(\{v_0\}) \cap \mathcal{F}_{\mathscr{D}}(\mathrm{Dom}) \end{array} \right.$$

Can we do better?

Dominance Equivalences

$$[\ u\ \mathtt{dom}(\mathrm{P}_G)\ v_0 \iff u = v_0\ ]$$

$$[\ u\ \mathtt{dom}(\mathrm{P}_G)\ v \iff u = v \vee \langle \forall w : w \to v : u\ \mathtt{dom}(\mathrm{P}_G)\ w \rangle\ ]$$

Dominance Equations

$$\mathrm{Dom}(v_0) = \{v_0\}$$

$$\mathrm{Dom}(v) = \bigcap_{w \in \mathtt{pred}(v)} \mathrm{Dom}(w) \cup \{v\}$$

An Optimized Algorithm*

for $v \in V$ **do**
    $\mathrm{Dom}[v] \leftarrow V$
$\mathrm{Dom}[v_0] \leftarrow \{v_0\}$
$\mathrm{Changed} \leftarrow \mathrm{true}$
while Changed **do**
    $\mathrm{Changed} \leftarrow \mathrm{false}$
    for $v \in V$ **do**
        $\mathrm{newSet} \leftarrow \left(\bigcap_{w \in \mathtt{pred}(v)} \mathrm{Dom}[w]\right) \cup \{v\}$
        if $\mathrm{newSet} \neq \mathrm{Dom}[v]$ **then**
            $\mathrm{Dom}[v] \leftarrow \mathrm{newSet}$
            $\mathrm{Changed} \leftarrow \mathrm{true}$

* K. D. Cooper, T. J. Harvey, and K. Kennedy. *A simple, fast dominance algorithm.*

Complexity

$\mathcal{O}(|V|^2)$
```
for v ∈ V do
    Dom[v] ← V
Dom[v_0] ← {v_0}
Changed ← true
while Changed do
    Changed ← false
    for v ∈ V do
        newSet ← (⋂_{w ∈ pred(v)} Dom[w]) ∪ {v}
        if newSet ≠ Dom[v] then
            Dom[v] ← newSet
            Changed ← true
```

$$\mathcal{O}(|V|^2)\left\{\begin{array}{l}\textbf{for } v \in V \textbf{ do}\\ \quad \mathrm{Dom}[v] \leftarrow V\\ \mathrm{Dom}[v_0] \leftarrow \{v_0\}\\ \mathrm{Changed} \leftarrow \mathrm{true}\\ \textbf{while } \mathrm{Changed} \textbf{ do}\\ \quad \mathrm{Changed} \leftarrow \mathrm{false}\end{array}\right.$$

$$\mathcal{O}(|V| \times |E|)\left\{\begin{array}{l}\textbf{for } v \in V \textbf{ do}\\ \quad \mathrm{newSet} \leftarrow \left(\bigcap_{w \in \texttt{pred}(v)} \mathrm{Dom}[w]\right) \cup \{v\}\\ \quad \textbf{if } \mathrm{newSet} \neq \mathrm{Dom}[v] \textbf{ then}\\ \quad\quad \mathrm{Dom}[v] \leftarrow \mathrm{newSet}\\ \quad\quad \mathrm{Changed} \leftarrow \mathrm{true}\end{array}\right.$$

$$
\mathcal{O}(|V|^2)\left\{
\begin{array}{l}
\textbf{for } v \in V \textbf{ do} \\
\quad \mathrm{Dom}[v] \leftarrow V \\
\mathrm{Dom}[v_0] \leftarrow \{v_0\} \\
\mathrm{Changed} \leftarrow \mathrm{true}
\end{array}
\right.
$$

$$
\mathcal{O}(|V|^3 \times |E|)\left\{
\begin{array}{l}
\textbf{while } \mathrm{Changed} \textbf{ do} \\
\quad \mathrm{Changed} \leftarrow \mathrm{false} \\
\textbf{for } v \in V \textbf{ do} \\
\quad \mathrm{newSet} \leftarrow \left(\bigcap_{w \in \texttt{pred}(v)} \mathrm{Dom}[w]\right) \cup \{v\} \\
\quad \textbf{if } \mathrm{newSet} \neq \mathrm{Dom}[v] \textbf{ then} \\
\quad\quad \mathrm{Dom}[v] \leftarrow \mathrm{newSet} \\
\quad\quad \mathrm{Changed} \leftarrow \mathrm{true}
\end{array}
\right.
$$

$$O(|V|^3 \times |E|)$$

Bottleneck: naïve iteration

for $v \in V$ **do**
    $\mathrm{Dom}[v] \leftarrow V$
$\mathrm{Dom}[v_0] \leftarrow \{v_0\}$
$\mathrm{Changed} \leftarrow \mathrm{true}$
while Changed **do**
    $\mathrm{Changed} \leftarrow \mathrm{false}$
    for $v \in V$ **do**
        $\mathrm{newSet} \leftarrow \left(\bigcap_{w \in \mathtt{pred}(v)} \mathrm{Dom}[w]\right) \cup \{v\}$
        if newSet $\neq \mathrm{Dom}[v]$ **then**
            $\mathrm{Dom}[v] \leftarrow \mathrm{newSet}$
            $\mathrm{Changed} \leftarrow \mathrm{true}$

Usual Hacks for Performance*

Reverse postorder

$$\mathcal{O}(|V| \times |E|)$$

Using priority queue

$$\mathcal{O}(|V|^2)$$

* K. D. Cooper, T. J. Harvey, and K. Kennedy. *A simple, fast dominance algorithm.*

The Pattern Summarized

1. A concrete domain $\langle \wp(V^+), \subseteq \rangle$

2. A concrete functional $p_G : \wp(V^+) \to \wp(V^+)$

3. An abstract domain $\langle \wp(V \times V), \supseteq \rangle$

4. An abstraction $\mathtt{dom} = (\mathtt{in}/) \circ f$

5. A Galois connection $\langle \wp(V^+), \subseteq \rangle \underset{\mathtt{dom}}{\overset{\overline{\mathtt{dom}}}{\leftrightarrows}} \langle \wp(V \times V), \supseteq \rangle$

6. "Pushing alphas" $\mathtt{dom} \circ p_G = \mathcal{F}_{\mathscr{D}} \circ \mathtt{dom}$

7. An iterative algorithm $\mathsf{lfp}_{\supseteq} \mathcal{F}_{\mathscr{D}}$

Shortest Path Algorithm (an overview)

1. A concrete domain $\langle \wp(V_w^+), \subseteq \rangle$

2. A concrete functional $p_{G_w} : \wp(V_w^+) \to \wp(V_w^+)$

3. An abstract domain $\langle V \to (\mathbb{N} \cup \{\infty\}), \dot{\geq} \rangle$

4. An abstraction $\texttt{dist}(\mathcal{X}) = \lambda v. \min\{\tau : \tau \in \mathcal{X} \ \wedge \ \texttt{last}(\tau) = v : \|\tau\|\}$

5. A Galois connection $\langle \wp(V_w^+), \subseteq \rangle \underset{\texttt{dist}}{\overset{\overline{\texttt{dist}}}{\leftrightarrows}} \langle V \to (\mathbb{N} \cup \{\infty\}), \dot{\geq} \rangle$

6. "Pushing alphas" $\texttt{dist} \circ p_{G_w} = \mathcal{F}_\delta \circ \texttt{dist}$

7. An iterative algorithm $\mathsf{lfp}_{\dot{\geq}} \mathcal{F}_\delta$

Details in the paper

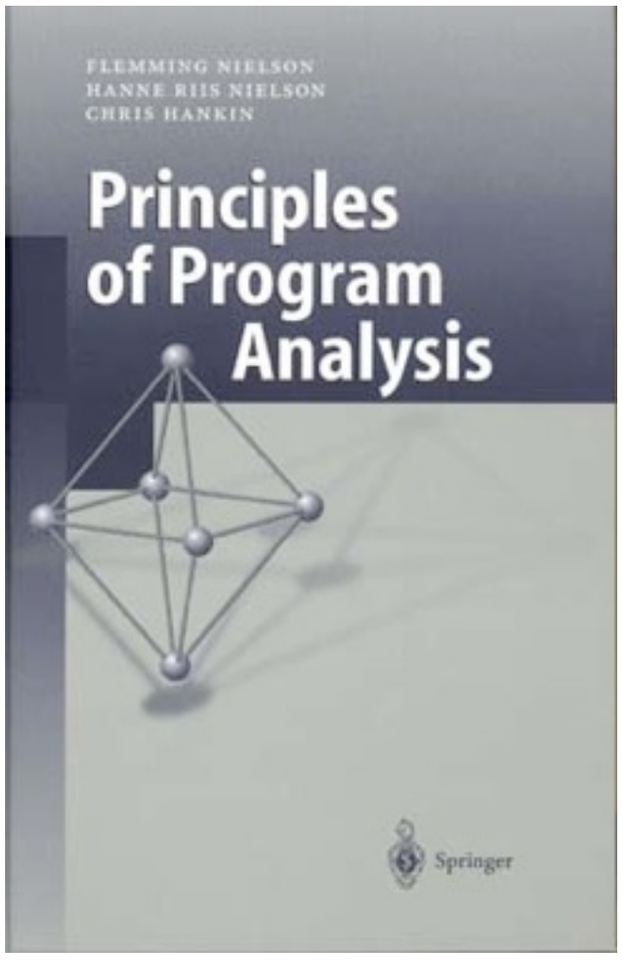
An algorithm*

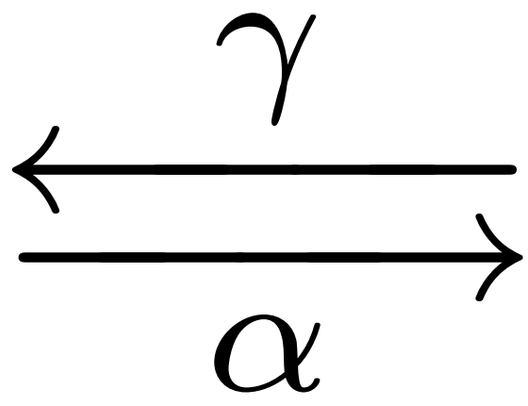
for $u \in V$ **do**
    $\delta[u] \leftarrow \infty$
$\delta[v_0] \leftarrow 0$
Changed $\leftarrow$ true
while Changed **do**
    Changed $\leftarrow$ false
    for $v \in V$ **do**
        for $u \in \texttt{pred}(v)$ **do**
            if $\delta[u] + W[u, v] < \delta[v]$ **then**
                $\delta[v] \leftarrow \delta[u] + W[u, v]$
                Changed $\leftarrow$ true

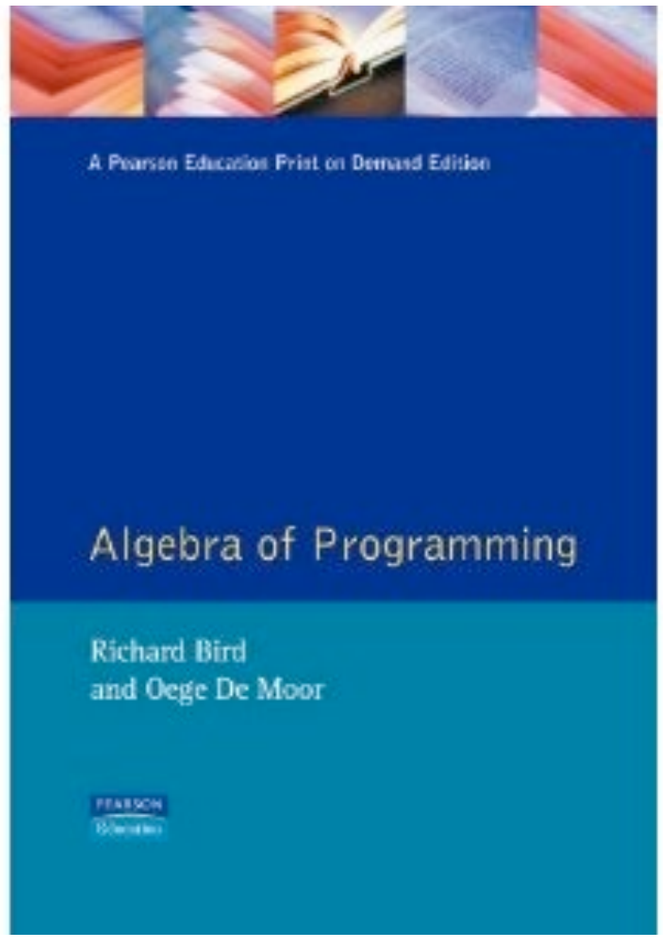
* R. Bellman. *On a routing problem.* 1958

Conclusion

- Various graph problems can be formulated as properties of sets of finite paths
- Sets of paths can be taken as a concrete domain
- A property can be taken as an abstract domain
- A mapping from paths to a property can be proved to be a lower adjoint in a Galois connection
- Fixpoint fusion law to obtain an algorithm
- Correctness is by construction


FLEMMING NIELSON
HANNE RIIS NIELSON
CHRIS HANKIN
Principles of Program Analysis
Springer


γ
α


A Pearson Education Print on Demand Edition
Algebra of Programming
Richard Bird
and Oege De Moor
PEARSON

¡Gracias!