Efficient Deductive Methods for Program Analysis

Harald Ganzinger

Max-Planck-Institut für Informatik

Introduction

- program analysis from high-level inference rules
- complexity analysis through general meta-complexity theorems
- logical aspects of fundamental algorithmic paradigms (dynamic programming, union-find, congruence closure)
- treatment of transitive relations: implication, equivalence, congruence, quasi-orderings
- avoiding the cubic-time bottleneck
- variable-free specializations of fundamental first-order methods: resolution, Knuth/Bendix-completion, ordered chaining
- closely related to McAllester's SAS'99 talk and paper

Contents

Linear-time analyses

Example: interprocedural reachability

Logic background: linear-time bottom-up deduction

Analyses for type congruences

Examples:

Steensgaard's pointer analysis ($O(n \log n)$)

Henglein's subtype analysis ($O(n^2)$)

Logic background: congruence closure for Horn clauses

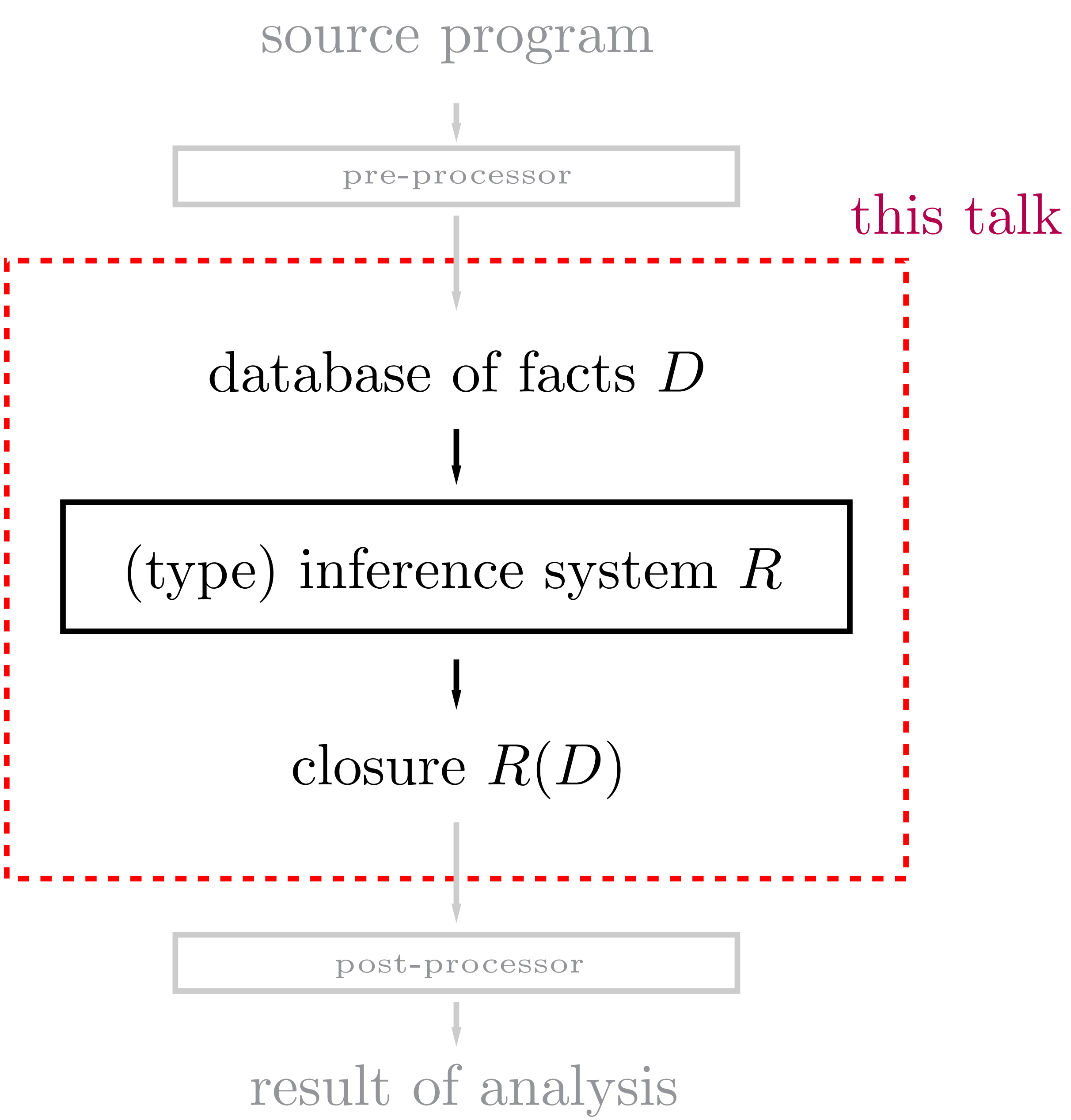
Dynamic transitive closure

Example: Andersen's pointer analysis via atomic set contraints

Logic background: ordered chaining

I. Linear-Time Analyses

Paradigm


source program
pre-processor
this talk
database of facts D
(type) inference system R
closure $R(D)$
post-processor
result of analysis

Example

program

```
1  procedure main
2  begin
3    declare x: int
4    read(x)
5    call p(x)
6  end

7  procedure p(a:int)
8  begin
9    if a>0 then
10     read(g)
11     a:=a-g
12     call p(a)
13     print(a)
14   fi
15 end
```

facts

```
proc(main,2,6)
next(main,2,5)
call(main,p,5,6)

proc(p,8,15)
next(p,8,12)
call(p,p,12,13)
next(p,13,15)
next(p,8,15)
```

Interprocedural Reachability IPR

Read "$L \Rightarrow L'$ in P" as "L' can be reached from L in procedure P".

$$\frac{\begin{array}{c} next(Q, L, L') \\ X \Rightarrow L \text{ in } Q \end{array}}{X \Rightarrow L' \text{ in } Q} \qquad \frac{\begin{array}{c} call(Q, P, L_c, L_r) \\ proc(P, L_0, L_f) \\ L_0 \Rightarrow L_f \text{ in } P \\ X \Rightarrow L_c \text{ in } Q \end{array}}{X \Rightarrow L_r \text{ in } Q} \qquad \frac{proc(P, L_0, L_f)}{L_0 \Rightarrow L_0 \text{ in } P}$$

THEOREM 1.1 $IPR(D)$ can be computed in time $O(|D|)$.

[$|D|$ = size of D = number of nodes in tree representation]

THEOREM 1.2 (MCALLESTER 1999) Let R be an inference system such that $R(D)$ is finite. Then $R(D)$ can be computed in time $O(|R(D)| + \mathsf{pf}_R(R(D)))$.

$\mathsf{pf}_R(R(D))$ is the number of prefix firings of R on $R(D)$:

$$\mathsf{pf}_R(D) = |\{(r, i, \sigma) \mid r = A_1 \wedge \ldots \wedge A_i \wedge \ldots \wedge A_n \supset A_0 \in R \\ A_j\sigma \in D, \text{ for } 1 \leq j \leq i\}|$$

COROLLARY 1.3 (DOWLING, GALLIER 1984) If R is ground, $R(D)$ can be computed in time $O(|D| + |R|)$.

Let $n = |D|$.

$$\frac{proc(P, L_0, L_f)}{L_0 \Rightarrow L_0 \text{ in } P}$$

has $O(n)$ (prefix) firings.[a]

$$\frac{\begin{array}{ll} next(Q, L, L') & O(n) * \\ X \Rightarrow L \text{ in } Q & O(1) \end{array}}{X \Rightarrow L' \text{ in } Q} \qquad \frac{\begin{array}{ll} call(Q, P, L_c, R_r) & O(n) * \\ proc(P, L_0, L_f) & O(1) * \\ L_0 \Rightarrow L_f \text{ in } P & O(1) * \\ X \Rightarrow L_c \text{ in } Q & O(1) \end{array}}{X \Rightarrow L_r \text{ in } Q}$$

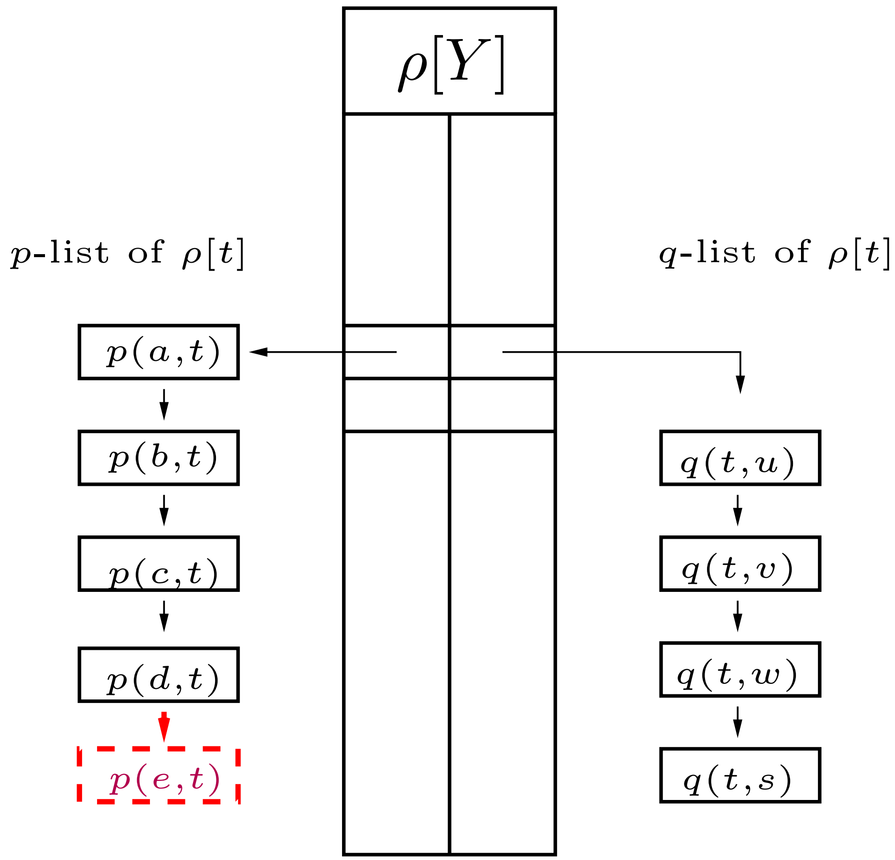
THEOREM 1.4 $IPR(D)$ can be computed in time $O(|D|)$.

Beweis. Both $|IPR(D)|$ and $\mathsf{pf}_{IPR}(IPR(D))$ are in $O(|D|)$. □

[a] Only facts $X \Rightarrow Y$ in P where X is the start label in P can be derived.

Proof of the Meta-Complexity Theorem

Data structure for rules ρ of the form $p(X, Y) \wedge q(Y, Z) \supset r(X, Y, Z)$



Upon adding a fact $p(e, t)$, fire all $r(e, t, z)$, for z on the q-list of $A[t]$.

The inference system can be transformed (maintaining pf) so that it contains unary rules and binary rules of the form ρ.

Problems

- if $R(D)$ infinite, consider $R(D) \cap \mathrm{atoms}(\mathrm{subterms}(D))$

  $\Rightarrow$ concept of local inferences (Givan, McAllester 1993)
- in the presence of transitive relations, complexity is in $\Omega(n^3)$

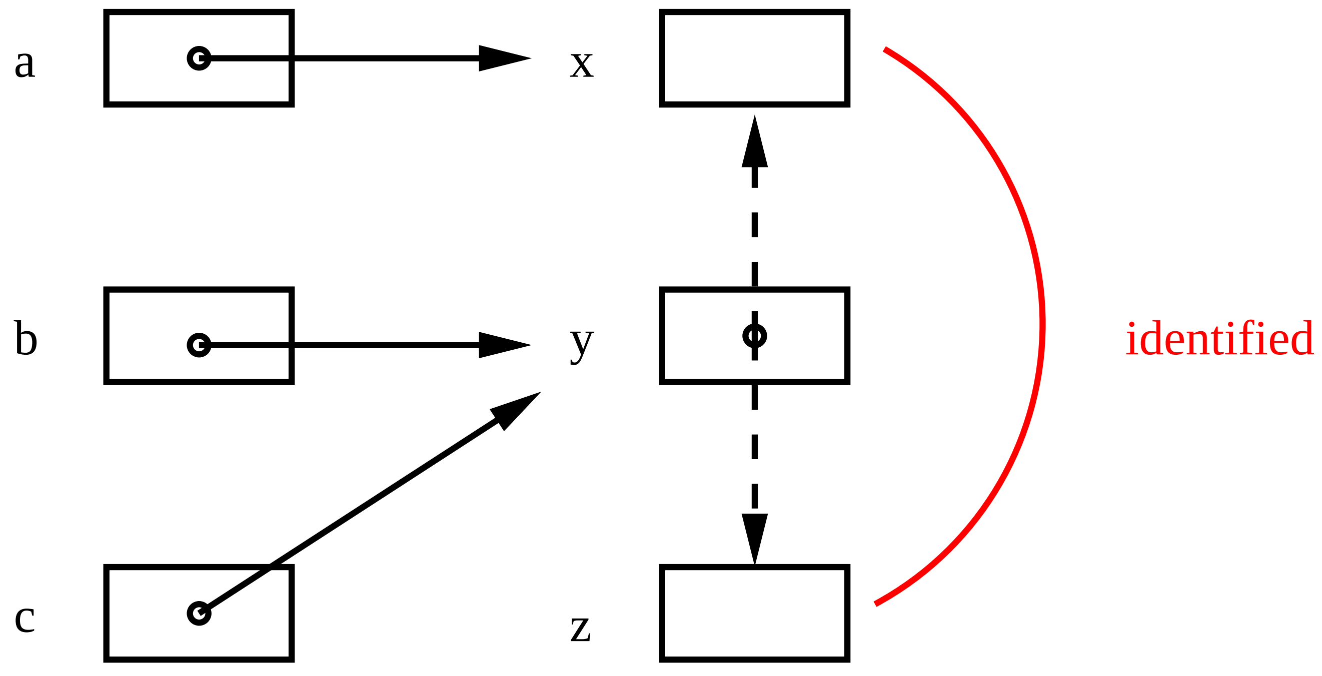
II. Equivalence and Congruence

Steensgaard's (1996) Pointer Analysis

program

```
a = &x
b = &y
if ... then
   y = &x;
else
   y = &z
fi
c = &y
```

shape graph



THEOREM 2.5 (STEENSGAARD 1996) Shape graphs can be computed in time $O(n\alpha(n, n))$.

assignments

$$\frac{\mathsf{input}(X = \&Y) \quad X : \mathsf{ref}(T_x) \quad Y : T_y}{T_x \doteq T_y} \qquad \frac{\mathsf{input}(X = Y) \quad X : \mathsf{ref}(T_x) \quad Y : \mathsf{ref}(T_y)}{T_y \leq T_x}$$

subtyping rules

$$\frac{}{\bot \leq T} \qquad \frac{\mathsf{ref}(T) \leq T'}{\mathsf{ref}(T) \doteq T'} \qquad \frac{\mathsf{ref}(T) \doteq \mathsf{ref}(T')}{T \doteq T'}$$

type equality

$$\frac{}{T \doteq T} \qquad \frac{T \doteq T' \quad T \doteq T''}{T'' \doteq T'} \qquad \frac{T \doteq T' \quad T' \leq T'' \quad T'' \doteq T'''}{T \leq T'''}$$

In the Example

facts from the program

$$a : \mathsf{ref}(\tau_a) \quad b : \mathsf{ref}(\tau_b) \quad c : \mathsf{ref}(\tau_c)$$

$$x : \mathsf{ref}(\tau_x) \quad y : \mathsf{ref}(\tau_y) \quad z : \mathsf{ref}(\tau_z)$$

derived equations from the assignments

$$\tau_a \doteq \mathsf{ref}(\tau_x) \quad \tau_b \doteq \mathsf{ref}(\tau_y) \quad \tau_y \doteq \mathsf{ref}(\tau_z)$$

$$\tau_y \doteq \mathsf{ref}(\tau_x) \quad \tau_c \doteq \mathsf{ref}(\tau_y)$$

additionally, after computing the closure

$$\mathsf{ref}(\tau_z) \doteq \mathsf{ref}(\tau_x) \quad \tau_z \doteq \tau_x$$

Meta-Complexity Theorem for Horn Clauses with Equality

THEOREM 2.6 (DOWNEY, SETHI, TARJAN 1980) Let $\mathcal{E}$ be a set of ground equations over terms in $\mathcal{T}$. Then $\mathcal{T}/\mathcal{E}$ is computable in time $O(n + m \log m)$, with $n = |\mathcal{E}|$ and $m = |\mathcal{T}|$.

THEOREM 2.7 (G, MCALLESTER 2001) Let $\mathcal{E}$ be a set of ground Horn clauses with equality[a] over terms in $\mathcal{T}$. Then $\mathcal{T}/\mathcal{E}$ is computable in time $O(n + \min(n \log m, m^2))$, with $n = |\mathcal{E}|$ and $m = |\mathcal{T}|$.

COROLLARY 2.8 $SPA(D)$ can be computed in time $O(|D|^2)$.

With some more work we can get it down to $O(n \log n)$.

---

[a] equivalences with some/all compatibility axioms

Henglein's (1996) Quadratic Subtype Analysis

Language with record types

$$\sigma = [l_1 : \sigma_1; \ldots ; l_n : \sigma_n]$$

and subtyping $\sigma \leq \tau$.

Main requirement to check: if $\sigma \leq \tau$ and τ accepts l, then σ accepts l.

Data base contains facts

- $accepts(\sigma, l)$ giving the field labels
- equations $\sigma.l_i \doteq \sigma_i$ for describing component types
- subtype facts of the form $\sigma \leq \tau$

Typing rules:

$$\frac{}{\sigma \sqsubseteq \sigma} \qquad \frac{\sigma \leq \tau \quad \tau \sqsubseteq \rho}{\sigma \sqsubseteq \rho} \qquad \frac{accepts(\sigma, l) \quad accepts(\tau, l) \quad \sigma \sqsubseteq \tau}{\sigma.l \doteq \tau.l}$$

Type equality is an equivalence, plus compatibility axioms:

$$\frac{\sigma \doteq \tau}{\sigma.l \doteq \tau.l} \qquad \frac{\sigma \doteq \sigma' \quad \sigma' \sqsubseteq \tau' \quad \tau' \doteq \tau}{\sigma \sqsubseteq \tau}$$

THEOREM 2.9 (HENGLEIN 1997) Subtype constraints can be checked in quadratic time.

Beweis. $STA(D)$ can be computed in time $O(|D|^2)$. □

Proof of 2nd Meta-Complexity Theorem

- extend the Downey, Sethi, Tarjan (1980) algorithm
- alternatively,
  - extend the first meta-complexity theorem to inference systems with priorities and deletion

    THEOREM 2.10 (G, McALLESTER 2001) Let R be an inference system with priorities and deletion such that all closures $R(D)$ are finite. Then one closure $R(D)$ can be computed in time $O(|R(D)| + \mathsf{pf}_R(R(D)))$.
  - define conditional congruence closure by inferences with priorities and deletion based on ideas by (Bachmair, Tiwari 2000)

Union-Find as Inferences with Priorities and Deletion

Inference system UF (priorities from left to right; premises in $[\ldots]$ are deleted after the rule has fired)[a]:

$$\frac{[x \doteq x]}{\top} \qquad \frac{\begin{array}{c}[x \rightarrow y] \\ y \rightarrow z\end{array}}{x \rightarrow z} \qquad \frac{\begin{array}{c}[x \doteq y] \\ x \rightarrow z\end{array}}{x \doteq z} \qquad \frac{\begin{array}{c}[x \doteq y] \\ [\mathit{weight}(x, w_1)] \\ \mathit{weight}(y, w_2) \\ w_1 \geq w_2\end{array}}{(y \rightarrow x) \wedge \mathit{weight}(x, w_1 + w_2)}$$

Theorem 2.11 Let $\mathcal{E}$ be a set of ground equations over terms in $\mathcal{T}$. Then $\mathsf{pf}_{UF}(UF(\mathcal{E}))$ is in $O(n \log m)$, with $n = |\mathcal{E}|$ and $m = |\mathcal{T}|$.

With a slightly more sophisticated system we obtain $O(n + m \log m)$.

[a] We also need the symmetric variants of the last two rules, and we assume that initial data bases initialize *weight* by 1.

III. Dynamic Transitive Closure

Quasi-Orderings with Monotone Functions

Basic axioms QO

$$\frac{}{x \Rightarrow x} \qquad \frac{x \Rightarrow x' \quad x' \Rightarrow x''}{x \Rightarrow x''} \qquad \frac{x \Rightarrow x'}{f(x) \Rightarrow f(x')} \text{ for certain } f$$

optionally exploiting the induced congruence

$$\frac{x \Rightarrow y \quad y \Rightarrow x}{x \doteq y}$$

additionally, for atomic set constraints (Melski, Reps 1997):

$$\frac{f(x) \Rightarrow f(y)}{x \Rightarrow y}$$

additionally, from pointer analysis:

$$\frac{\mathsf{input}(X = Y) \quad X : \mathsf{ref}(T) \quad Y : \mathsf{ref}(T')}{T' \Rightarrow T}$$

Ground Monadic Reachability

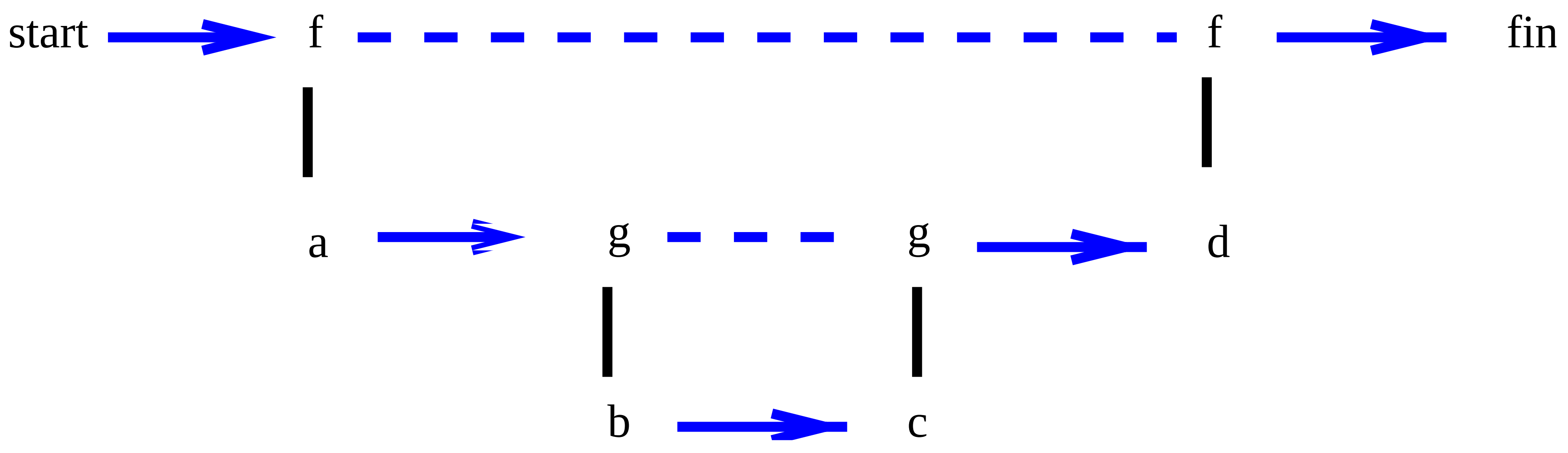
Decision problem:

$$QO \models (s_1 \Rightarrow t_1) \wedge \ldots \wedge (s_n \Rightarrow t_n) \supset (s_0 \Rightarrow t_0) \quad (s_i,\, t_i \text{ ground})$$

Example:

$$(\mathsf{start} \Rightarrow f\,a) \wedge (a \Rightarrow gb) \wedge (b \Rightarrow c) \wedge (gc \Rightarrow d) \wedge (f\,d \Rightarrow \mathsf{fin}) \supset (\mathsf{start} \Rightarrow \mathsf{fin})$$

Graphically:



Results about Ground Monadic Reachability

- GMR is 2NPDA-complete (Neal 1989)[a]
- 2NPDA acceptance is in $O(n^3)$ (Aho, Hopcroft, Ullman 1968)
- no subcubic algorithm known
- QO (also non-monadic) is a local theory, that is,
  $QO \models C$ iff QO[subterms in C] $\models C$,
  thus in $O(n^3)$ by (Dowling, Gallier 1980)

$$\frac{\dfrac{\mathsf{start} \Rightarrow fa \qquad \dfrac{\dfrac{a \Rightarrow gb \qquad \dfrac{\dfrac{b \Rightarrow c}{gb \Rightarrow gc} \qquad gc \Rightarrow d}{gb \Rightarrow d}}{a \Rightarrow d}}{fa \Rightarrow fd}}{\mathsf{start} \Rightarrow fd} \qquad fd \Rightarrow \mathsf{fin}}{\mathsf{start} \Rightarrow \mathsf{fin}}$$

---

[a] This holds for flat terms already.

Many Data Flow Problems are Equivalent with GMR

- atomic set constraints (Melski, Reps 1997)
- interprocedural reachability for higher-order languages (Heintze, McAllester 1997)
- Amadio/Cardelli typability (Heintze, McAllester 1997)
- Andersen's (1994) pointer analysis (Aiken et al 1998)

Ordered Chaining

Issue: better balancing of forward and backward computation

History:
- Bledsoe, Kunen, Shostak (1985), Hines (1992): limes theorems, set theory
- Levy, Agustí (1993): bi-rewriting for distributive lattices
- Bachmair, G (1996): ordered chaining for binary relations

Assumption: ground terms are ordered by $\succ$ (total, well-founded, ...)

Ordered Chaining OC:

$$\frac{y \Rightarrow x \quad u[x] \Rightarrow v}{u[y] \Rightarrow v} \text{ if } x \succ y \text{ and } u \succ v$$

(Ground) reachability through rewrite proofs: [a]

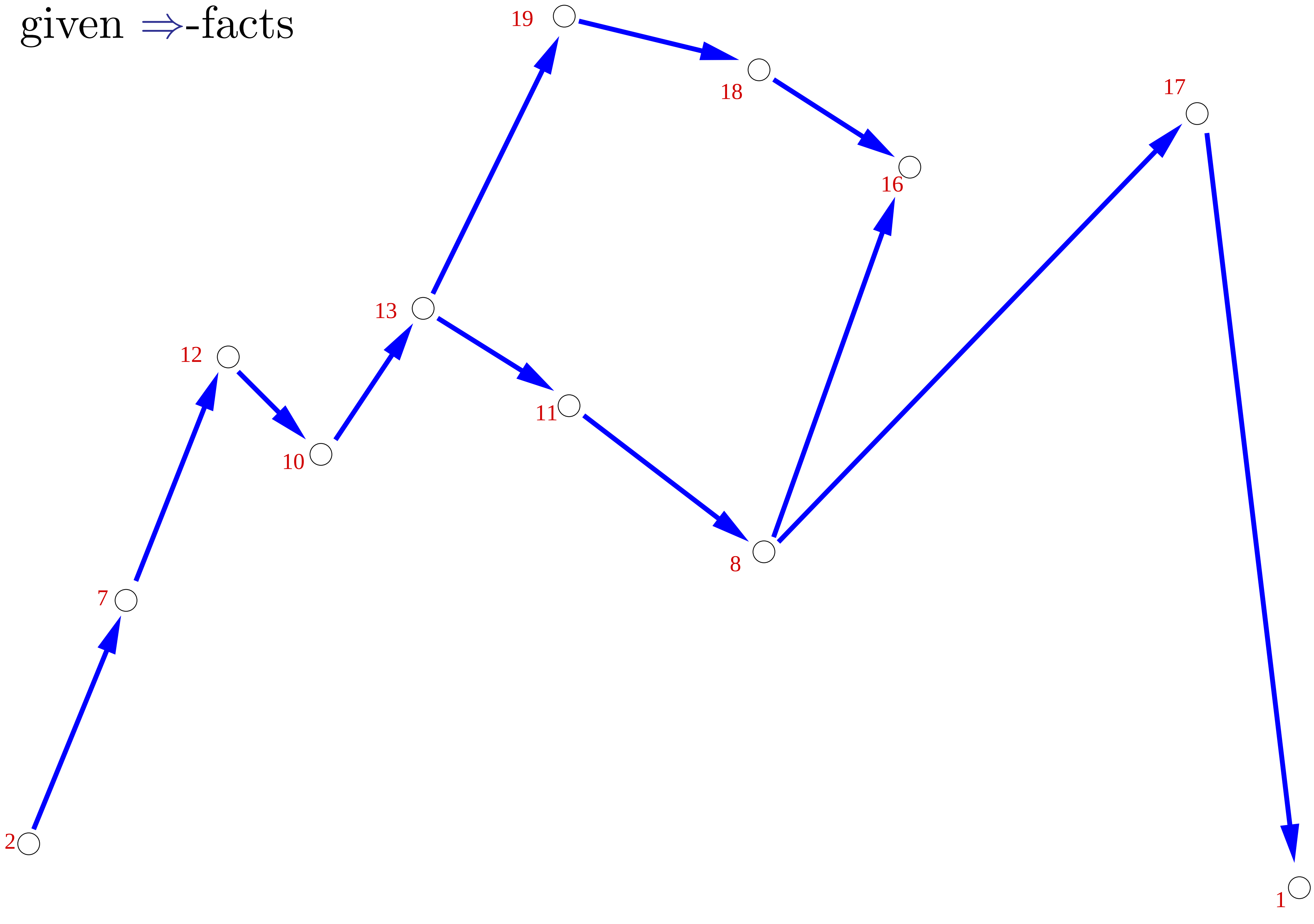
$QO \models D \supset (s \Rightarrow t)$ iff $s \overset{\vee}{\Rightarrow} t$ in $OC(D)$, that is,

$$s \underset{\succ}{\Rightarrow} \ldots \underset{\succ}{\Rightarrow} w \underset{\prec}{\Rightarrow} \ldots \underset{\prec}{\Rightarrow} t$$

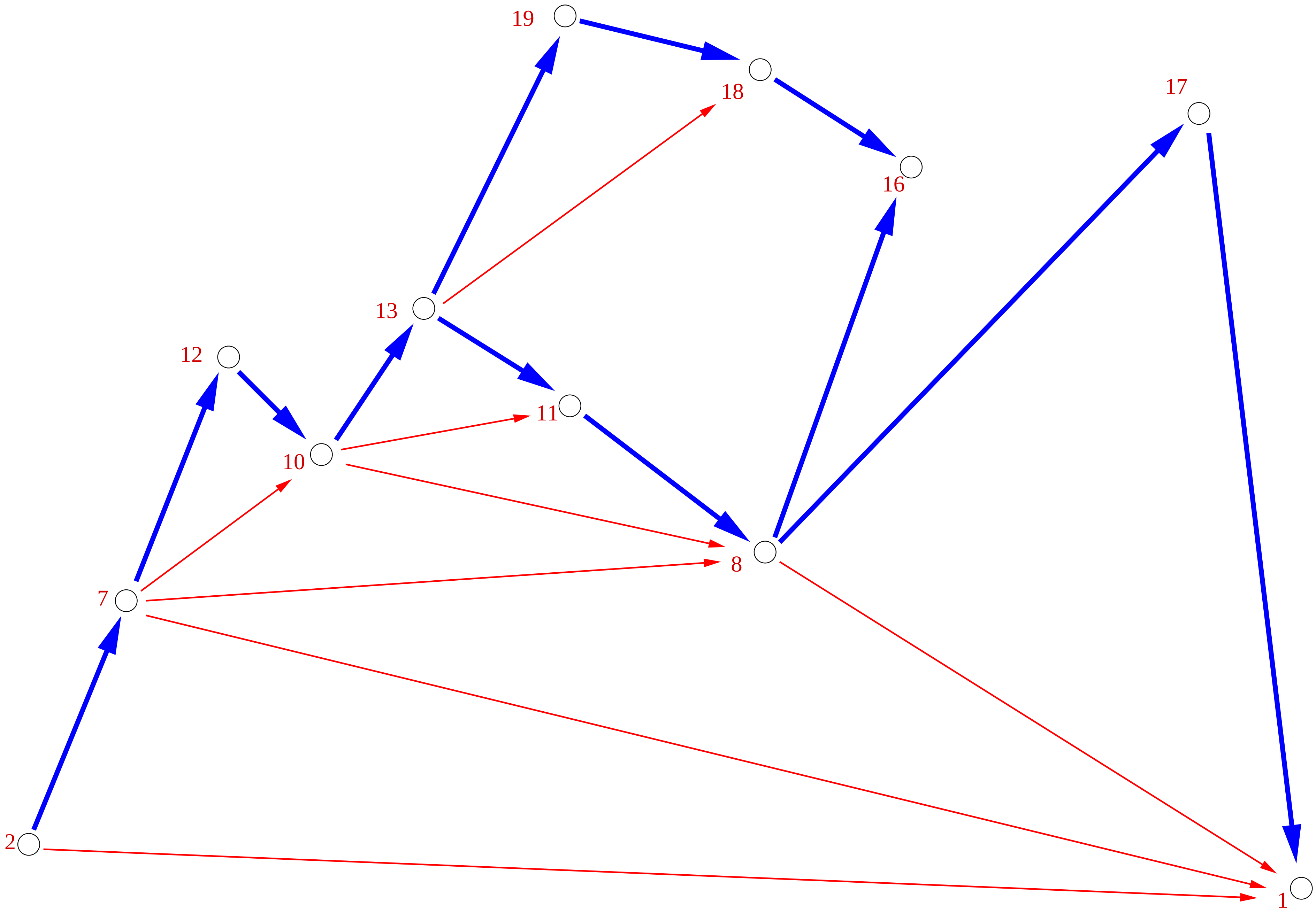
[a] for flat terms decidable in $O(|D|^2)$ since $|OC(D)|$ is in $O(|D|^2)$.

Chaining Diagram (Terms Ordered by Number)

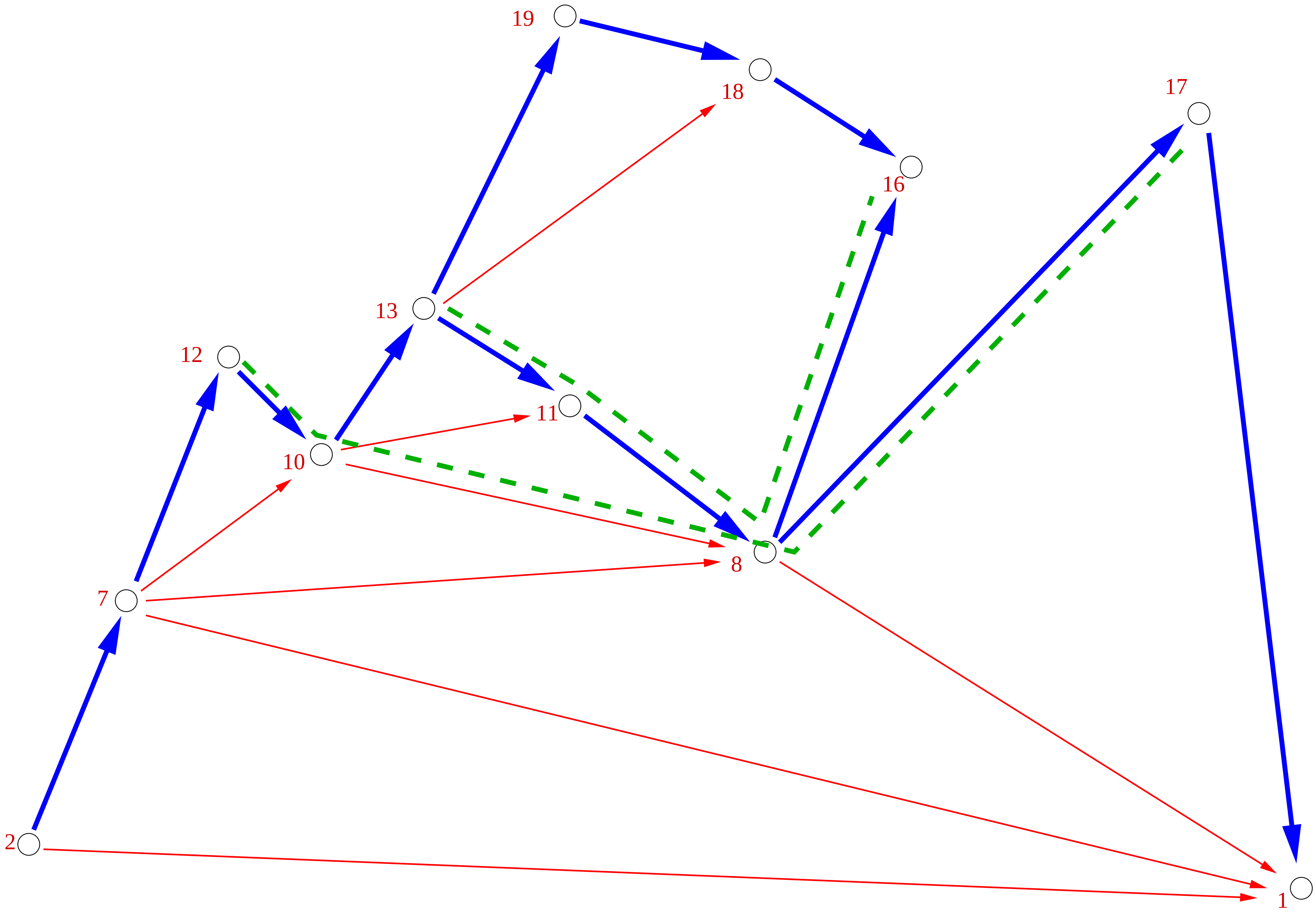
given $\Rightarrow$-facts



Adding Peak Facts



Reachability Through Rewrite Proofs



Adding Equality and Set Constraints

Deriving equations from inequations is optional. Using them for simplification collapses cycles. Premises in parenthesis become redundant and can be deleted.

$$\frac{[x \overset{\vee}{\Rightarrow} y] \quad [y \overset{\vee}{\Rightarrow} x]}{x \doteq y} \text{ (whenever you like)} \qquad \frac{x \doteq y \quad [A(x)]}{A(y)} \text{ (if } x \succ y\text{)}$$

Negative inequations in inference rules have to be replaced by rewrite provability, e.g., for set constraints we may add:

$$\frac{f(x) \overset{\vee}{\Rightarrow} f(y)}{x \Rightarrow y}$$

Theoretical Results and Open Questions

- completeness
- worst-case complexity not better than $O(n^3)$
- for which classes of data bases quadratic?
- how to choose a good ordering?

Practical Results

Encouraging results by Aiken, Fähndrich, Foster, Su (1998, 2000) for Andersen's pointer analysis via atomic set constraints:

- flat inequations $\mathcal{X} \Rightarrow \mathcal{Y}$, $\mathsf{ref}(\mathcal{X}) \Rightarrow \mathcal{Y}$, and $\mathcal{X} \Rightarrow \mathsf{ref}(\mathcal{Y})$
- $\mathsf{ref}(\mathcal{X})$ minimal in $\succ$, therefore, $O(1)$ test for injectivity
- if $\succ$ on set variables is random, then relatively few variable-variable edges are added
- partial cycle elimination according to

$$\frac{x \underset{\succ}{\Rightarrow} \ldots \underset{\succ}{\Rightarrow} y \quad y \underset{\prec}{\Rightarrow} x}{x \doteq y}$$

- analytical model: $O(1)$ for partial cycle test; ordered chaining adds only 40% of the transitive edges
- transformation to delay peak computation that eventually collapse

Very long programs can be analysed in reasonable time

Conclusions

Fundamental problem: efficient deduction for transitive relations in algebraic structures

Logical view: clarifies the issues and provides general efficient methods

Advice to the PL community: adopt that view and obtain almost optimal complexity results and prototype implementations for free

Advice to the ATP community:

- make first-order provers work well on these near-propositional cases
- find more meta-complexity theorems for the general case
- implement the algorithms behind the meta-complexity theorems
- analytical models for ordered chaining: when is GMR sub-cubic?