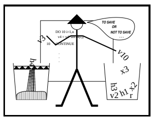
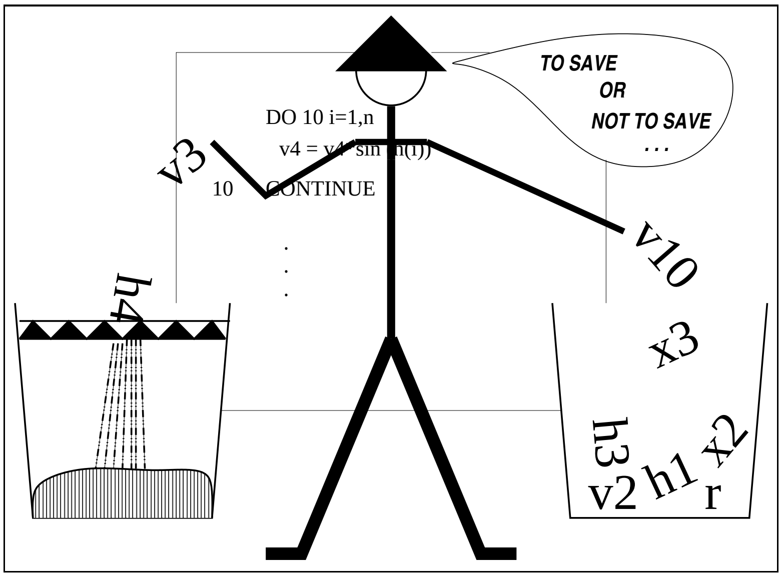


The Selective Storage Problem



Uwe Naumann

INRIA Sophia-Antipolis
Action TROPICS

Uwe.Naumann@sophia.inria.fr

- Automatic Differentiation
- **O∂yssée** and the Selective Storage Problem
- Trajectory Analysis
- Outlook: Reversal Modes and advanced TA

What is Automatic Differentiation?

1. Alternative to *symbolic differentiation* and *divided differences* (a good one!)
2. Systematic application of the **chain rule** to an algorithmic specification of a function
3. **Automatic** application on computers
4. Directional derivatives to machine precision

Two philosophies of implementing AD:

1. Source Code Transformation (O∂yssée, Padre2, ADIFOR)
   → derivative code
2. Operator Overloading (ADOL-C/F, AD01)
   → compile-time solution

Forward and **Reverse** modes of AD

→ two special cases of interpreting the chain rule.

X-country

→ the general case

Rall and Corliss: *An Introduction to Automatic Differentiation*,
in Computational Differentiation: Techniques, Applications, and Tools,
M. Berz, C. Bischof, G. Corliss, and A. Griewank, eds, SIAM, Philadelphia, PA, 1996.

Evaluation program of

$$y = F(x), \qquad F : \mathbb{R}^n \to \mathbb{R}^m$$

Algorithmic specification of F as e.g. Fortran or C program

$\Rightarrow$ Sequence of scalar assignments

$$v_j = \varphi_j(v_i)_{i \in \mathcal{P}_j} \quad \in \mathbb{R}$$

$\Rightarrow$ **Code list**

$X \equiv \{v_{1-n}, v_{2-n}, \ldots, v_{-1}, v_0\}$	n independent variables
$Z \equiv \{v_1, v_2, \ldots, v_p\}$	p intermediate variables
$Y \equiv \{v_{p+1}, v_{p+2}, \ldots, v_{p+m}\}$	m dependent variables

Differentiability

For some fixed argument $x \in \mathbb{R}^n$ the elementary functions

$$\varphi_j, \qquad j = 1, 2, \ldots, p+m,$$

are well defined and have jointly continuous partial derivatives

$$c_{ji}(v_k)_{k \in \mathcal{P}_j} \equiv \frac{\partial}{\partial v_i} \varphi_j(v_k)_{k \in \mathcal{P}_j} \quad \text{for} \quad i \in \mathcal{P}_j$$

on some neighborhood

$$\mathcal{D}_j \subset \mathbb{R}^{n_j} \qquad \text{with} \quad n_j \equiv \mid \mathcal{P}_j \mid$$

of their respective arguments $(v_i)_{i \in \mathcal{P}_j}$.

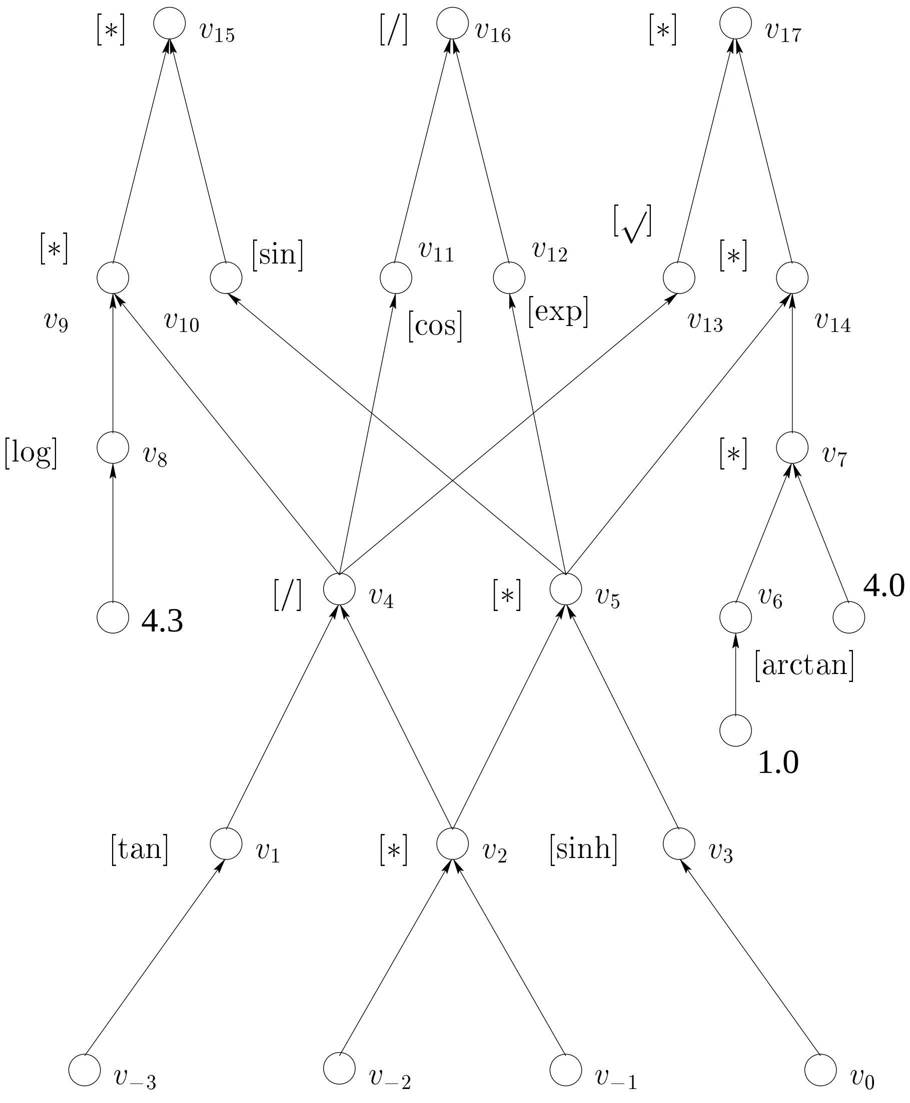
Computational Graph

```
REAL x(1:4), y(1:3), h1, h2, pi
...
h1 = TAN(x(0))/(x(1)*x(2))
h2 = x(1)*x(2)*SINH(x(3))
pi = 4.0*ATAN(1.0)
y(0) = LOG(4.3)*h1*SIN(h2)
y(1) = COS(h1)/EXP(h2)
y(2) = SQRT(h1)*pi*h2
...
```

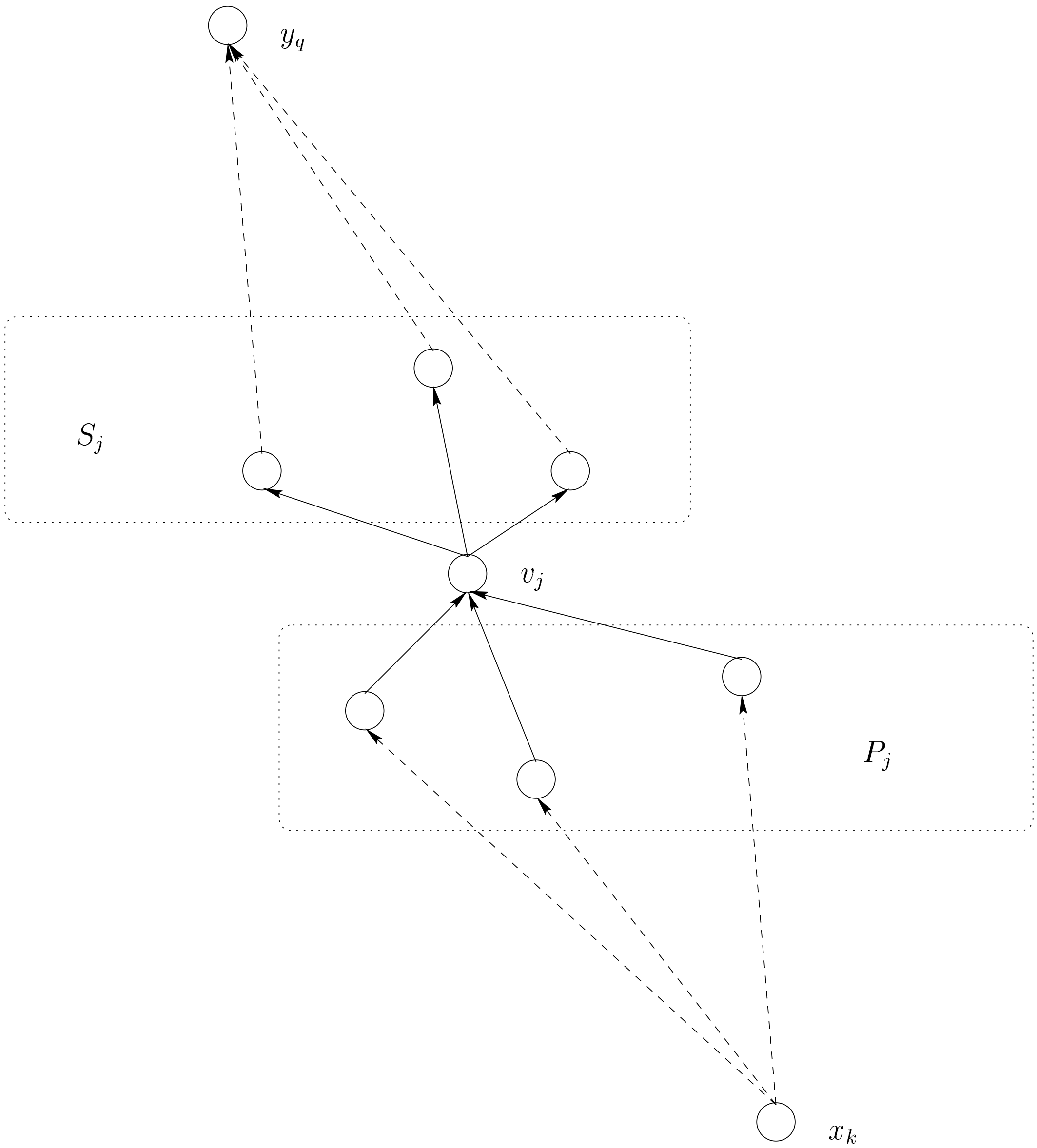
$$\ldots$$

$$v_5 = v_2 v_3 \quad \rightarrow \quad \dot{v}_5 = v_3 \dot{v}_2 + v_2 \dot{v}_3$$

$$\ldots$$



Chain Rule



Forward Mode $F : \mathbb{R}^{(n+n_z)} \to \mathbb{R}^m, \quad y = F(x,z)$

$$v_j = \varphi_j(\{v_l\}) \quad l \in \mathcal{P}_j, \qquad \varphi_j : \mathbb{R}^{|\mathcal{P}_j|} \to \mathbb{R}$$

$$\text{Chain Rule} \qquad \frac{\partial v_j}{\partial x_k} = \sum_{l \in \mathcal{P}_j} c_{jl} \cdot \frac{\partial v_l}{\partial x_k}$$

Forward propagation of tangents using

$$\dot{v}_j = \sum_{l \in \mathcal{P}_j} c_{jl} \cdot \dot{v}_l \qquad \text{set} \quad \underline{\dot{x}} \quad \Rightarrow \quad J \cdot \underline{\dot{x}} \quad \text{(columns)}$$

(J by n forward sweeps with $\underline{\dot{x}}_i = \underline{e}_i \in \mathbb{R}^n$)

Vector version

$$Y = F(X,z), \quad Y = [y|\dot{Y}], \quad X = [x|\dot{X}], \quad \dot{X} \in \mathbb{R}^{n\times s}, \dot{Y} \in \mathbb{R}^{m\times s}$$

$$(\mathbb{R}^{n\times n} \ni) \quad \dot{X} = I \quad \Rightarrow \quad \dot{Y} = F'_x(x,z)$$

Overloading: $[\overset{\uparrow}{y}|\overset{\uparrow}{\dot{Y}}] = F([\underset{\uparrow}{x}|\underset{\uparrow}{\dot{X}}], \underset{\uparrow}{z})$ ("natural information flow")

Reverse Mode $\quad F : \mathbb{R}^{(n+n_z)} \to \mathbb{R}^m, \quad y = F(x,z)$

$$\mathcal{S}_j := \{ l \ \mid \exists v_l = \varphi_l(V_l) : \quad v_j \in V_l \}$$

$$\text{Chain Rule} \qquad \frac{\partial y_q}{\partial v_j} = \sum_{l \in \mathcal{S}_j} \frac{\partial y_q}{\partial v_l} \cdot c_{l,j}$$

Backward propagation of gradients using

$$\bar{v}_j = \sum_{l \in \mathcal{S}_j} c_{lj} \cdot \bar{v}_l \qquad \text{set} \quad \underline{\bar{y}} \quad \Rightarrow \quad \underline{\bar{y}}^T \cdot J \quad \text{(rows)}$$

J by m reverse sweeps with $\underline{\bar{y}}_i = \underline{e}_i \in \mathbb{R}^m$

Vector version

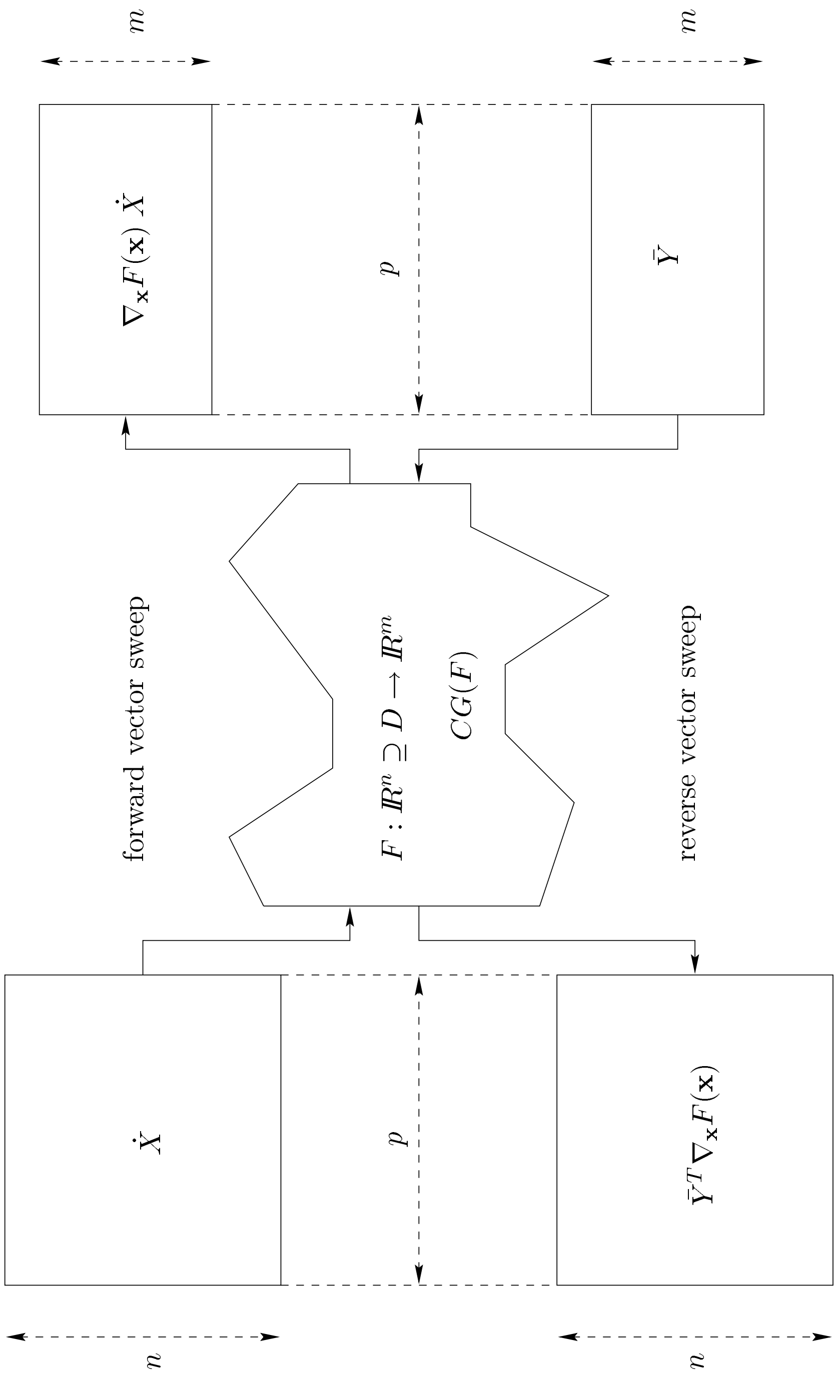
$$Y = F(X,z), \quad Y = [y|\bar{Y}], \quad X = [x|\bar{X}], \quad \bar{X} \in \mathbb{R}^{n \times s}, \bar{Y} \in \mathbb{R}^{m \times s}$$

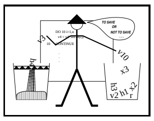
$$(\mathbb{R}^{m \times m} \ni) \quad \bar{Y} = I \quad \Rightarrow \quad \bar{X} = F'_x(x,z)$$

$$\text{Problem:} \qquad [\overset{\uparrow}{y}|\underset{\uparrow}{\bar{Y}}] = F([\underset{\uparrow}{x}|\underset{\uparrow}{\overset{\uparrow}{\bar{X}}}], \underset{\uparrow}{z})$$

("inverted information flow" ... Overloading???)

Interpreting the Trajectory





O∂yssée and the SSP

- save before overwritten
- local trajectory

⇒ **Problems!**

e.g.

- y saved before initialized
- x saved as linear argument
- both restored without being used

```
program problem2
real a,x,y
call p2s1(a,x,y)
end
```

```
subroutine p2s1(a,x,y)
real a,x,y
y=a*x
x=y**2
end
```

Trajectory

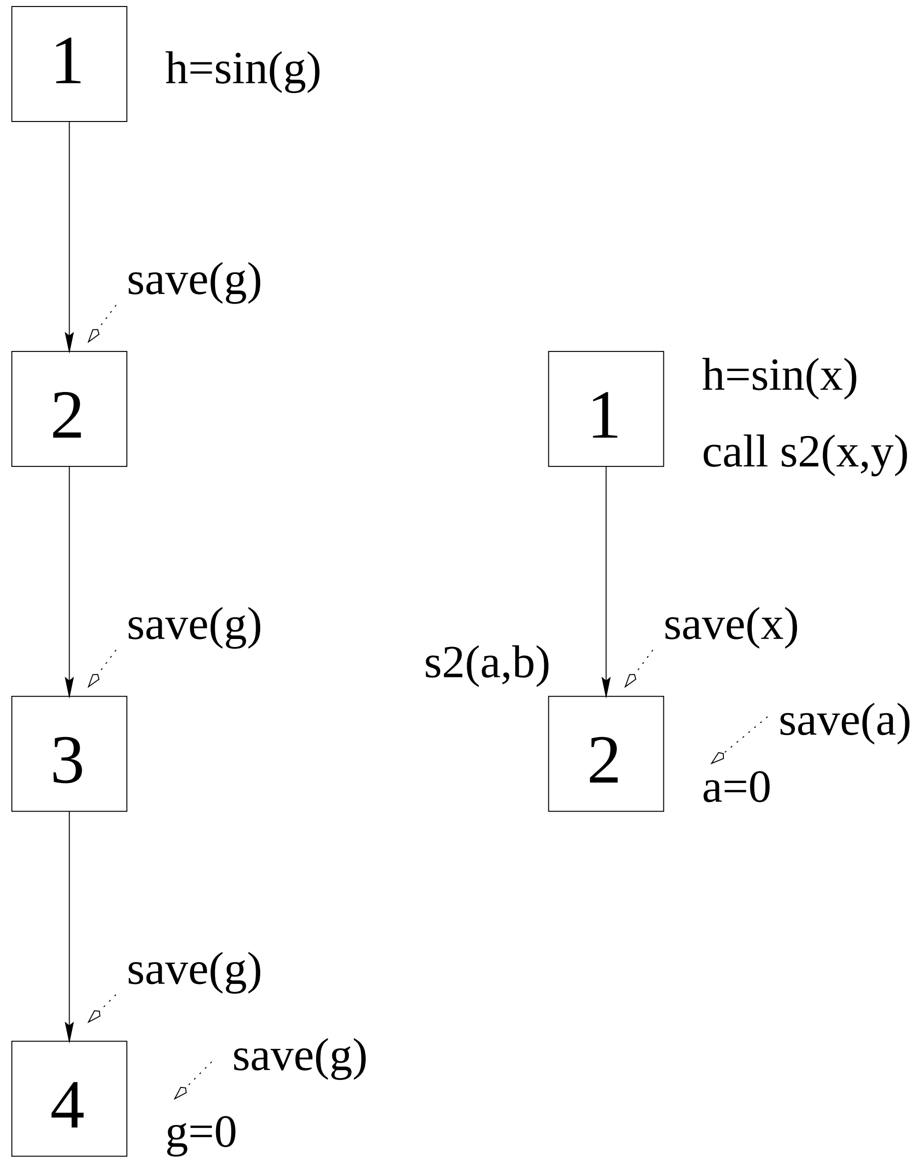
```
save1 = y
y = a*x
save2 = x
x = y**2
```

Transposed linear forms

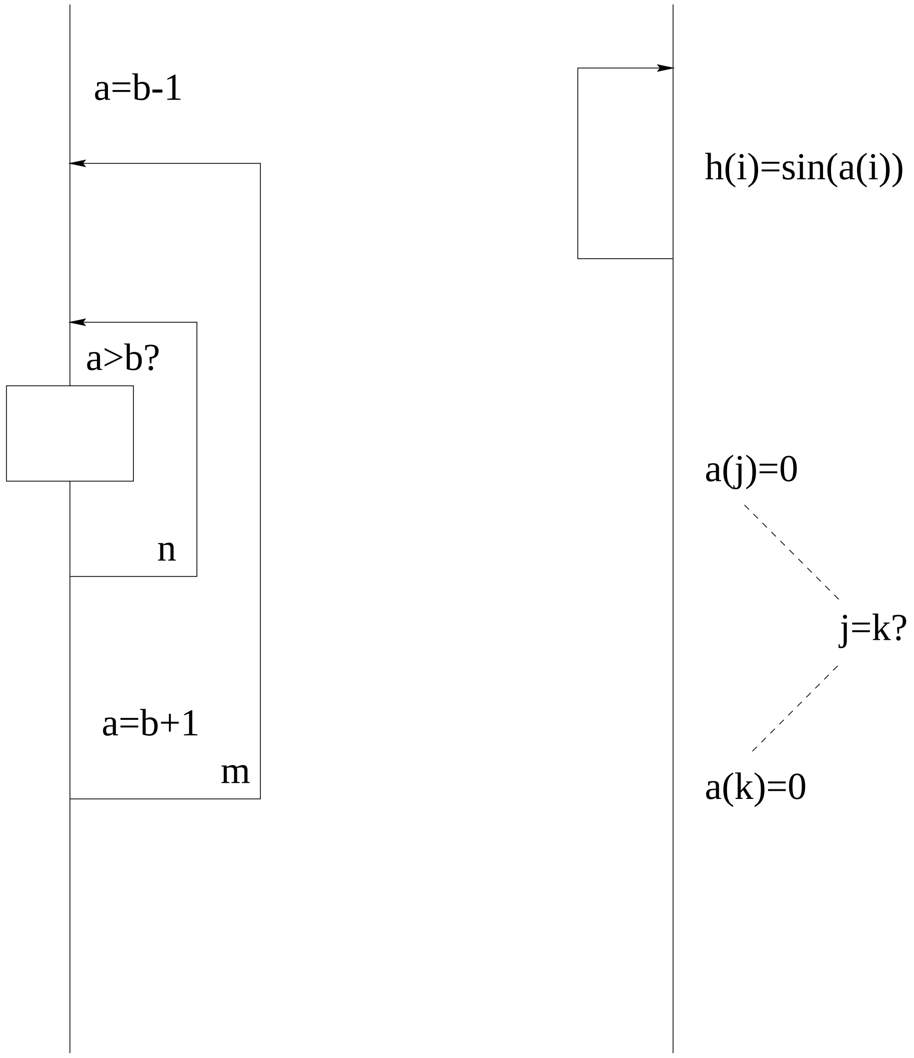
```
x = save2
yccl = yccl+xccl*(2*y)
xccl = 0.
y = save1
xccl = xccl+yccl*a
yccl = 0.
```

Signature of subroutines and functions

- global g
- argument x



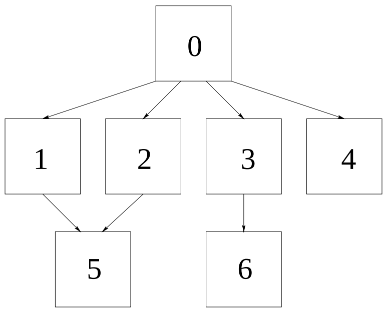
Control and Arrays



SSP Scenario

- **Given:**

  $\Pi = \Pi(f_i)_{i=0..n_f-1}$ evaluation program, d-unit hierarchy
  $f_0 = (\mathbf{x}, \mathbf{y}, \mathbf{CG}_\Pi)$ head routine
  $\mathbf{G}$ d-unit call graph



- **Find:**

  1. **Trajectory (TRJ)** of Π
     TRJ(Π) contains all values which are necessary (and useful) for the generation of a feasible (and efficient) derivative code for computing $Y^T \mathbf{y}_\mathbf{x}$.
  2. $\bar{X} = \bar{Y}^T \mathbf{y}_\mathbf{x}$

- LIFO $\Rightarrow$ `PUSH`, `POP`

- **To-Be-Saved** status (**TBS**) to each symbol (memory location)

Uwe Naumann (Uwe.Naumann@sophia.inria.fr)
Action TROPICS (http://www-sop.inria.fr/tropics)

Activity

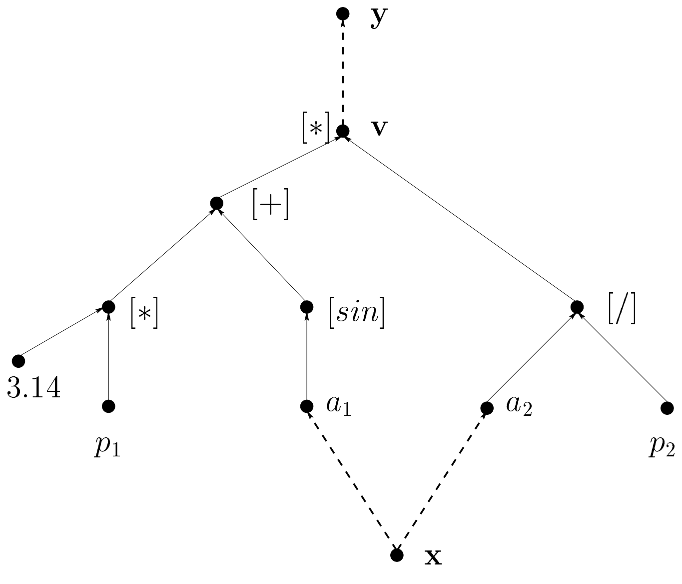
- $f(\mathbf{u}) \equiv (\mathbf{u}, \mathbf{v}, \mathbf{CG}) \rightarrow$ differentiate $\mathbf{v}_a \subseteq \mathbf{v}$ wrt. $\mathbf{u}_a \subseteq \mathbf{u}$
- forward activity (passive output: $\mathbf{v}_p = \mathbf{v} \setminus \mathbf{v}_a$)

$$FAct(+f) = (FAct(-f) \cup \mathbf{v}_a) \setminus \mathbf{v}_p$$

- backward activity ($\bar{f} \equiv (\mathbf{v} \equiv \mathbf{u}(\bar{f}), \mathbf{u} \equiv \mathbf{v}(\bar{f}), \overline{\mathbf{CG}})$)

$$BAct(-f) = BAct(+f) \cup \mathbf{v}_a(\bar{f})$$

- (required) activity $\Lambda_{\{\mathcal{A},\mathcal{P}\}}$: $Act(\pm f) = FAct(\pm f) \cap BAct(\pm f)$
- active (A) and passive (P) expressions in expression graph
- Example: $\mathbf{v} = (3.14 * p_1 + sin(a_1)) * (a_2/p_2)$



  - $a_1, a_2, \mathbf{v} \in \Lambda_{\mathcal{A}}$, $p_1, p_2 \in \Lambda_{\mathcal{P}}$
  - $3.14 * p_1 \in \mathcal{P}$;
  - $a_2/p_2 \in \mathcal{A}$

Uwe Naumann (Uwe.Naumann@sophia.inria.fr)
Action TROPICS (http://www-sop.inria.fr/tropics)

Formalisation

Every d-unit $f : I\!R^{n_\mathbf{u}} \supseteq D_\mathbf{u} \rightarrow I\!R^{n_\mathbf{v}} : \mathbf{u} \mapsto \mathbf{v} = f(\mathbf{u})$ can be regarded as

$$\mathbf{v} = f(\mathbf{u}) \equiv \varphi(\mathbf{p}_1, \mathbf{a}_2, \mathbf{p}_2) \equiv \varphi(\xi(\mathbf{p}_0), \psi(\mathbf{a}_1), \mathbf{p}_2)$$

where

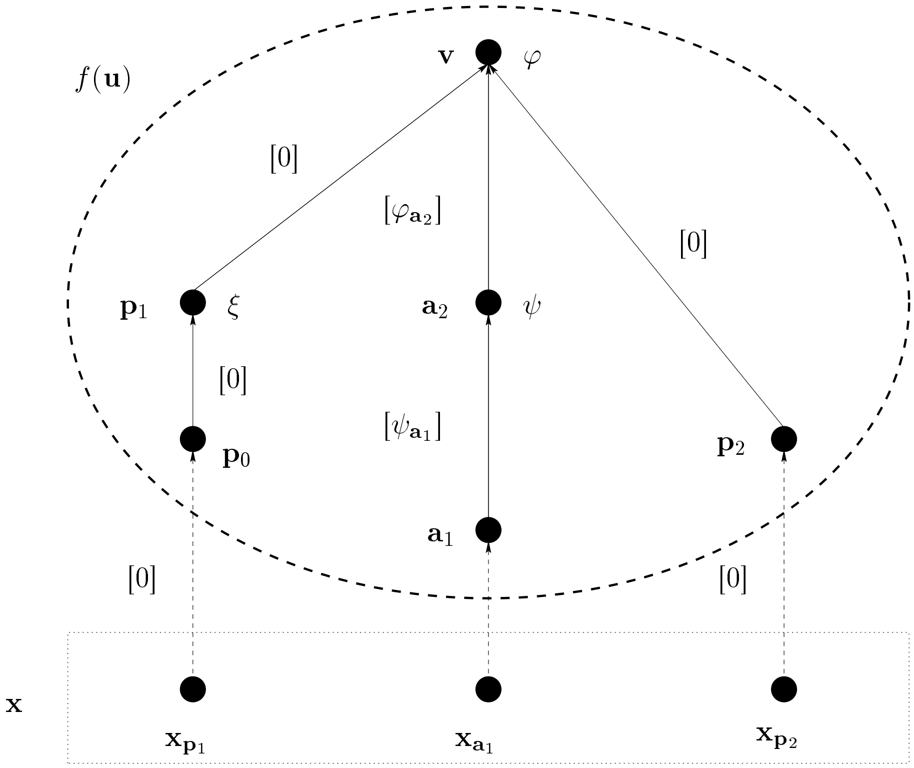
$$\varphi : I\!R^{n_{\tilde{\mathbf{u}}}} \rightarrow I\!R^{n_\mathbf{v}} \quad : \quad \tilde{\mathbf{u}} \mapsto \mathbf{v}$$

such that

$$\frac{\partial \varphi}{\partial \mathbf{u}} \equiv \frac{\partial \varphi}{\partial \mathbf{a}_1} \quad \text{and} \quad \frac{\partial \varphi}{\partial \mathbf{a}_2} = \frac{\partial \varphi}{\partial \mathbf{a}_2}(\mathbf{p}_0).$$

with

$$\xi : I\!R^{n_{\mathbf{p}_0}} \rightarrow I\!R^{n_{\mathbf{p}_1}} \quad : \quad \mathbf{p}_0 \mapsto \mathbf{p}_1 \quad \text{and} \quad \psi : I\!R^{n_{\mathbf{a}_1}} \rightarrow I\!R^{n_{\mathbf{a}_2}} \quad : \quad \mathbf{a}_1 \mapsto \mathbf{a}_2.$$



Trajectory Analysis

- $\mathtt{TA}_i(f)$ $(i = 0, 1, \ldots)$ depends on *type* of f and *depth of recursion* i.
- top-level call

$$\mathtt{TA}_0(f) \quad \wedge \quad (\mathtt{SAVE}(v_i) \text{ if } v_i \in \mathbf{v} \text{ and } \mathbf{TBS}(v_i) = \mathtt{true}).$$

-

$$\mathtt{TA}_i(f) \equiv \begin{cases} \mathbf{TBS}(\lambda_j) = \mathtt{true} & \text{if } f \hat{=} \lambda_j \in \Lambda_\Pi \ \wedge \ i > 0 \\ \mathtt{TA}_i(\varphi(\xi(\mathbf{p}_0), \psi(\mathbf{a}_1), \mathbf{p}_2)) \ \text{(see below)} & \text{if } f \equiv \varphi(\xi(\mathbf{p}_0), \psi(\mathbf{a}_1), \mathbf{p}_2) \end{cases}$$

- chain rule

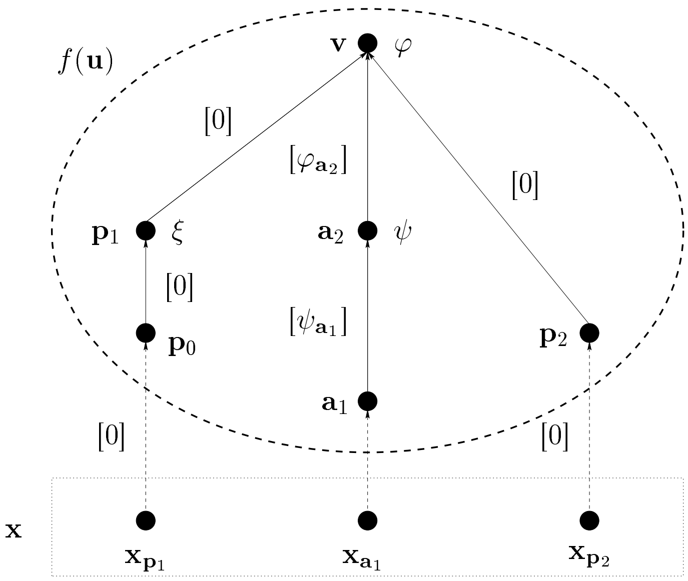
$$\frac{\partial \mathbf{v}}{\partial \mathbf{x}} = \varphi_{\mathbf{a}_2} \psi_{\mathbf{a}_1}(\mathbf{a}_1)_{\mathbf{x}} = \mathbf{v}_{\mathbf{a}_2}(\mathbf{a}_2)_{\mathbf{a}_1}(\mathbf{a}_1)_{\mathbf{x}_{\mathbf{a}_1}}.$$

$$\Rightarrow$$

$$\mathtt{TA}_i(\varphi(\mathbf{p}_1, \mathbf{a}_2, \mathbf{p}_2)) = \mathtt{TA}_i(\varphi(\xi(\mathbf{p}_0), \psi(\mathbf{a}_1), \mathbf{p}_2)) = [\mathtt{TA}_{i+1}(\mathbf{p}_0) \quad \wedge \quad \mathtt{TA}_{i+1}(\mathbf{a}_1)]$$

where

$$\mathtt{TA}_{i+1}(\mathbf{p}_0) = \{\mathtt{TA}_{i+2}(\mathbf{p}_0^j) \ \text{ for } j = 1, .., n_{\mathbf{p}_0}\}, \quad \mathtt{TA}_{i+1}(\mathbf{a}_1) = \{\mathtt{TA}_{i+2}(\mathbf{a}_1^j) \ \text{ for } j = 1, .., n_{\mathbf{a}_1}\}.$$



Uwe Naumann (Uwe.Naumann@sophia.inria.fr)
Action TROPICS (http://www-sop.inria.fr/tropics)

TA - Special Cases

- **Exponential**

$$f(\mathbf{u}) \equiv \varphi(\mathbf{p}_1, \mathbf{a}_1, \mathbf{p}_2) = \mathbf{p}_1 \exp(\mathbf{a}_1) + \mathbf{p}_2, \qquad \frac{\partial \mathbf{v}}{\partial \mathbf{x}} = \mathbf{p}_1 \psi(\mathbf{a}_1)(\mathbf{a}_1)_{\mathbf{x}}$$

$$\mathtt{TA}_i(f) = \mathtt{SAVE}(\mathbf{p}_1 \exp(\mathbf{a}_1)) \;\wedge\; \mathtt{TA}_0(\mathbf{a}_1)$$

$$\Rightarrow$$

$$\frac{[\text{2*READ, 1*MULT, 1*EXP}]}{[\text{1*READ}]}$$

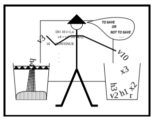
- **Reciprocal**

$$f(\mathbf{u}) \equiv \varphi(\mathbf{p}_1, \mathbf{a}_1, \mathbf{p}_2) = \mathbf{p}_1 \frac{1}{\mathbf{a}_1} + \mathbf{p}_2, \qquad \frac{\partial \mathbf{v}}{\partial \mathbf{x}} = -\mathbf{p}_1 \left(\frac{1}{\mathbf{a}_1}\right)^2 (\mathbf{a}_1)_{\mathbf{x}}$$

$$\mathtt{TA}_i(f) = \mathtt{SAVE}(\mathbf{p}_1 \frac{1}{\mathbf{a}_1}, \frac{1}{\mathbf{a}_1}) \;\wedge\; \mathtt{TA}_0(\mathbf{a}_1)$$

$$\Rightarrow$$

$$\frac{[\text{2*READ, 2*MULT, 1*REC}]}{[\text{2*READ, 1*MULT}]}$$



TA - Power

- $$f(\mathbf{u}) \equiv \varphi(\mathbf{p}_1, \mathbf{a}_1, \mathbf{p}_2, \mathbf{p}_3) = \mathbf{p}_1 \mathbf{a}_1^{\mathbf{p}_2} + \mathbf{p}_3, \qquad \frac{\partial \mathbf{v}}{\partial \mathbf{x}} = \mathbf{p}_1 \mathbf{p}_2 \frac{\mathbf{a}_1^{\mathbf{p}_2}}{\mathbf{a}_1} (\mathbf{a}_1)_{\mathbf{x}}$$

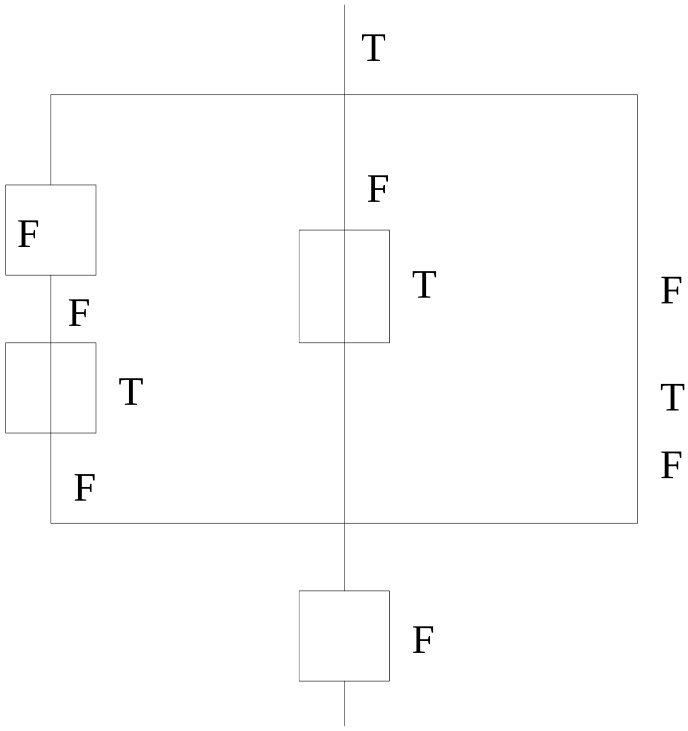
$$\mathtt{TA}_i(f) = \mathtt{SAVE}(\mathbf{p}_1 \mathbf{a}_1^{\mathbf{p}_2}, \mathbf{p}_2, \mathbf{a}_1) \ \wedge \ \mathtt{TA}_0(\mathbf{a}_1)$$
$$\Rightarrow$$
$$\frac{[\text{3*READ, 3*MULT, 1*POW, 1*REC}]}{[\text{3*READ, 2*MULT, 1*REC}]}$$

- $$f(\mathbf{u}) \equiv \varphi(\mathbf{p}_1, \mathbf{p}_2, \mathbf{a}_1, \mathbf{p}_3) = \mathbf{p}_1 \mathbf{p}_2^{\mathbf{a}_1} + \mathbf{p}_3, \qquad \frac{\partial \mathbf{v}}{\partial \mathbf{x}} = \mathbf{p}_1 \mathbf{p}_2^{\mathbf{a}_1} \log(\mathbf{p}_2)(\mathbf{a}_1)_{\mathbf{x}}$$

$$\mathtt{TA}_i(f) = \mathtt{PUSH}(\mathbf{p}_1 \mathbf{p}_2^{\mathbf{a}_1}, \mathbf{p}_2) \ \wedge \ \mathtt{TA}_0(\mathbf{a}_1)$$
$$\Rightarrow$$
$$\frac{[\text{3*READ, 2*MULT, 1*POW, 1*LOG}]}{[\text{2*READ, 1*MULT, 1*LOG}]}$$

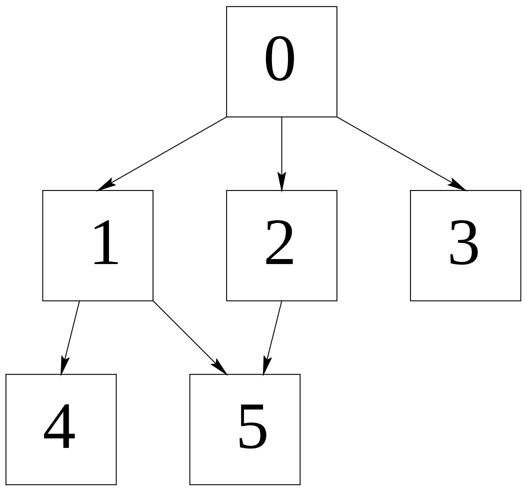
- $$f(\mathbf{u}) = \mathbf{p}_1 \mathbf{a}_1^{\mathbf{a}_2} + \mathbf{p}_2, \qquad \frac{\partial \mathbf{v}}{\partial \mathbf{x}} = \mathbf{p}_1 \mathbf{a}_2 \frac{\mathbf{a}_1^{\mathbf{a}_2}}{\mathbf{a}_1} (\mathbf{a}_1)_{\mathbf{x}} + \mathbf{p}_1 \mathbf{a}_1^{\mathbf{a}_2} \log(\mathbf{a}_1)(\mathbf{a}_2)_x$$

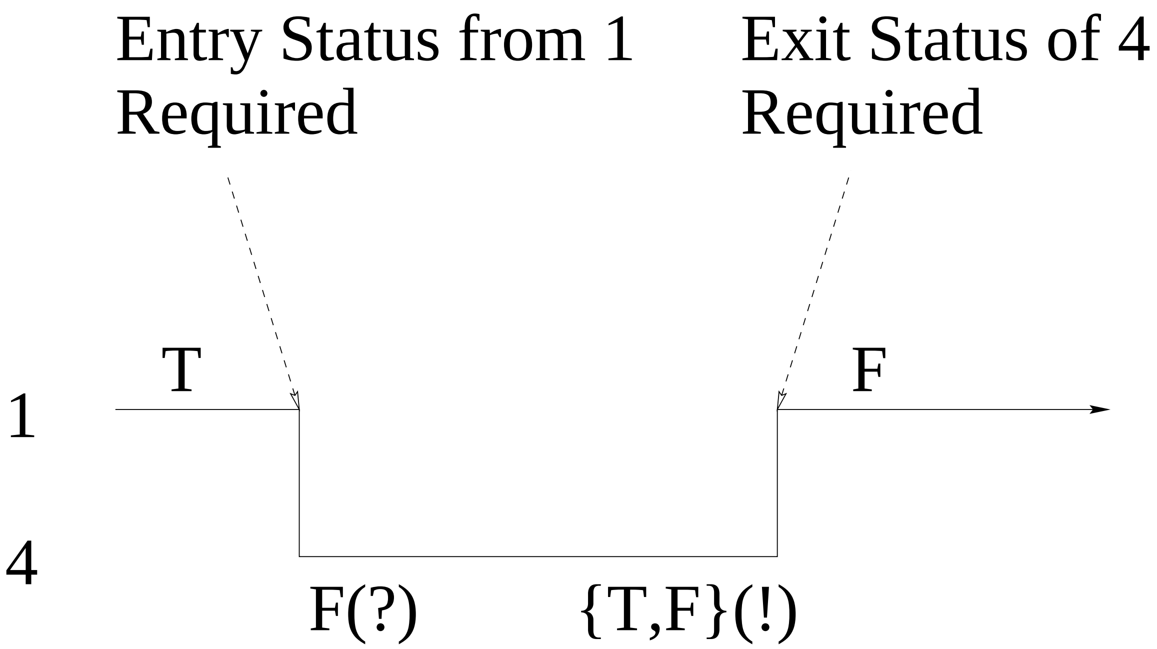
Conservative TA - Branches



Fixed Point TA - Loops

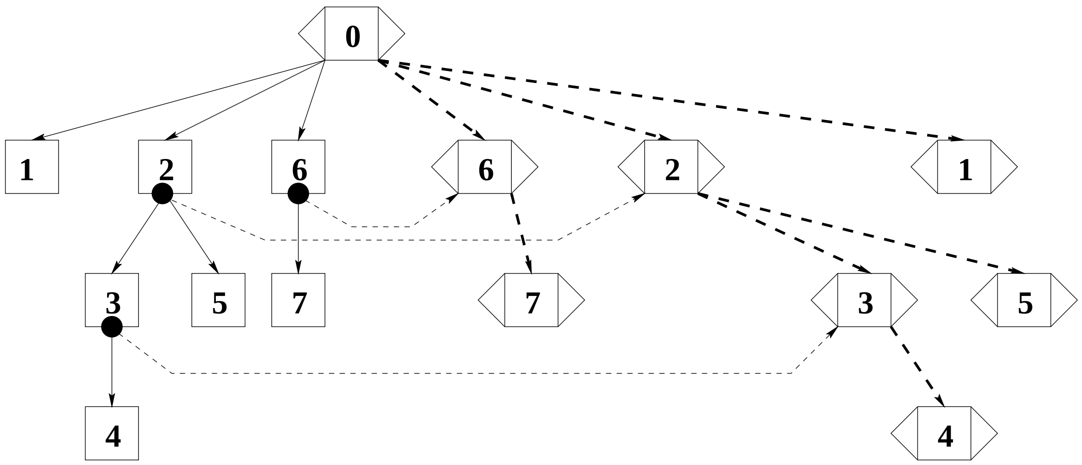
Call Graph Bottom-up-top-down Subroutine Analysis





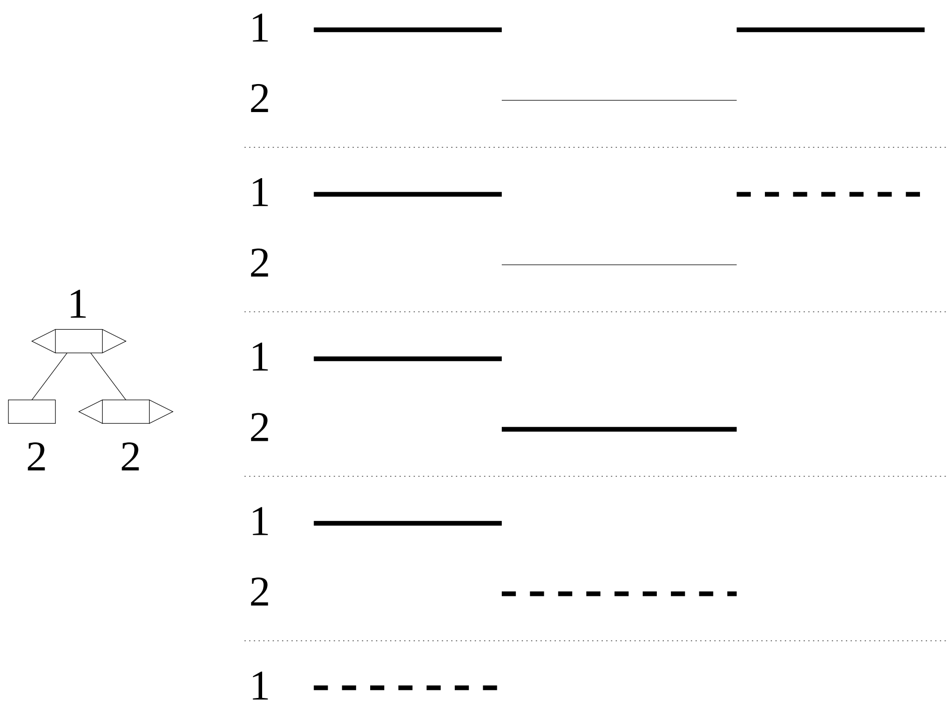
Exit information can be computed "out of context"!

Outlook - Result Checkpointing



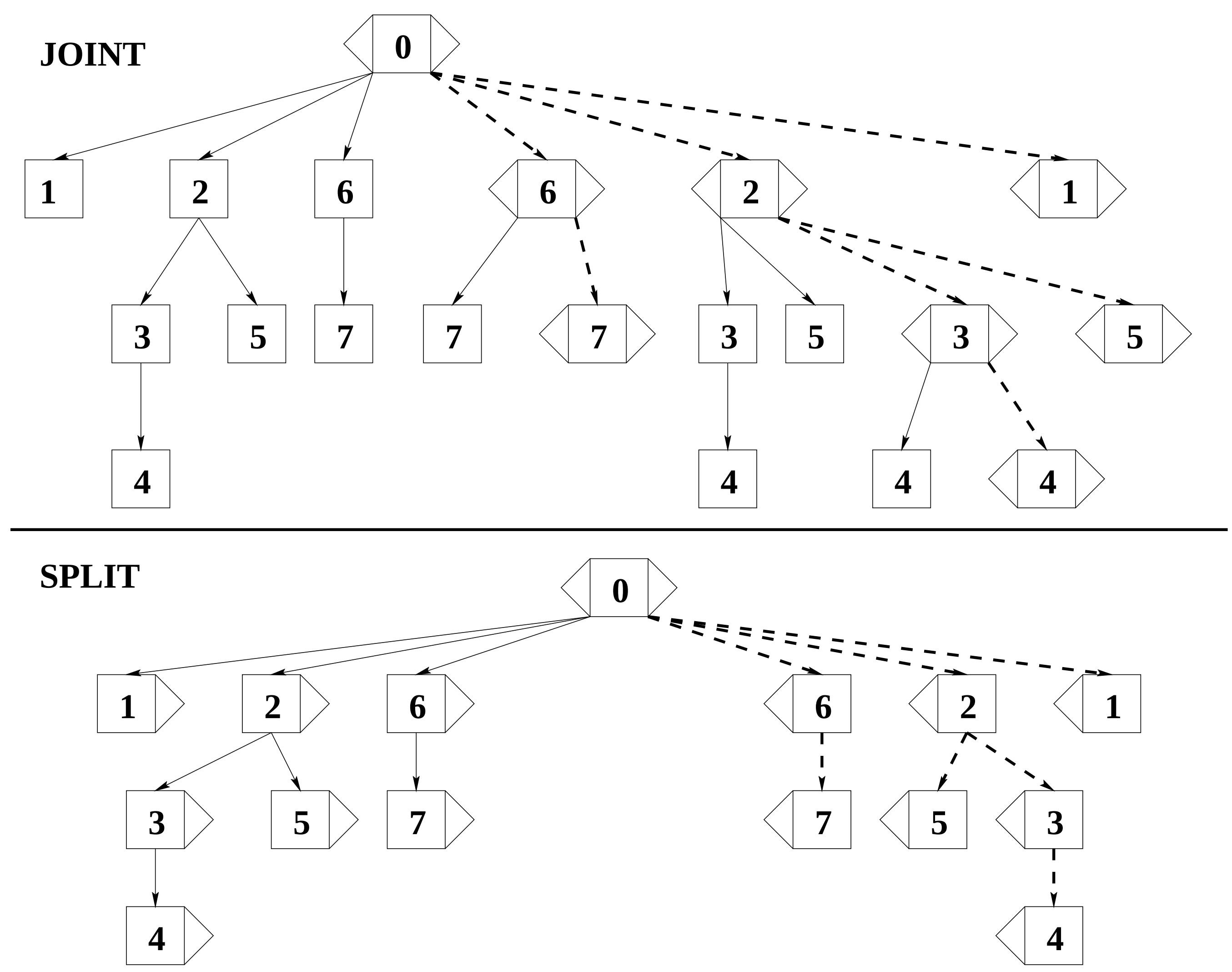
Case study: Tree depth: 10, Expensive Subroutines $\Rightarrow$ Factor 3

Outlook - Call Division



Outlook - Reversal Modes

(Notation by A. Griewank)

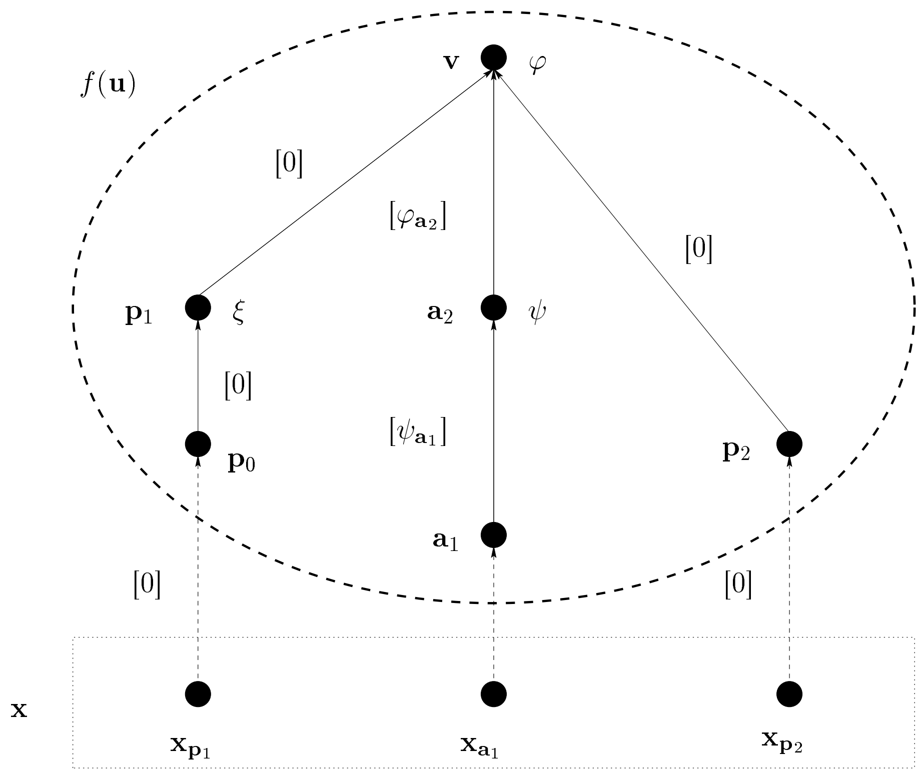


- *joint* saves memory space
- *split* saves operations

Outlook - Advanced TA

$$\mathtt{TA}_i(\varphi(\mathbf{p}_1, \psi(\mathbf{a}_1), \mathbf{p}_2)) = \ldots$$

$$\ldots = \begin{cases} \mathtt{SAVE}(\varphi_{\mathbf{a}_2}\psi_{\mathbf{a}_1}) \ \wedge \ \mathtt{TA}_0(\mathbf{a}_1) & \text{partial} \\ \mathtt{SAVE}(\mathbf{p}_1, \mathbf{a}_1) \ \wedge \ \mathtt{TA}_0(\mathbf{a}_1) & \text{all results} \\ \mathtt{SAVE}(\mathbf{p}_1) \ \wedge \ \mathtt{TA}_{i+1}(\mathbf{a}_1) & \text{passive results} \\ \mathtt{TA}_{i+1}(\mathbf{p}_1) \ \wedge \ \mathtt{TA}_{i+1}(\mathbf{a}_1) & \text{locations} \end{cases}$$



Uwe Naumann (Uwe.Naumann@sophia.inria.fr)
Action TROPICS (http://www-sop.inria.fr/tropics)