

Optimal Derivative Accumulation on Series-Parallel Dags

Andrew Lyons

11/15/07

Outline

- **Intro to Jacobians by AD**
- The Optimal Jacobian Accumulation Problem
- Two-Terminal Series-Parallel Dags and Decomposition Trees
- Our Current View of OJA
- Single-Terminal Series-Parallel Dags and Vertex Elimination
- General Series-Parallel Dags
- Modular Decomposition: a Generalization

Jacobians by Automatic Differentiation (AD)

code for $F(\mathbf{x}_0) : \mathbb{R}^n \rightarrow \mathbb{R}^m$, $\mathbf{x} \mapsto \mathbf{y}$

↓ AD

augmented code for $F(\mathbf{x}_0)$, $F'(\mathbf{x}_0)$

$$\text{Jacobian } J \equiv F'(\mathbf{x}) = \left(\frac{\partial y_j}{\partial x_i} \right)_{\substack{j=1,\dots,m \\ i=1,\dots,n}} \in \mathbb{R}^{m \times n}$$

Jacobians by Automatic Differentiation (AD)

$$\left[y_1 = x_1 * x_2 * \sin(x_1 * x_2), \quad y_2 = \cos(\sin(x_1 * x_2)) \right]^T$$

$$(v_{-1} = x_1, \quad v_0 = x_2)$$

$$v_1 = v_{-1} * v_0$$

$$v_2 = \sin(v_1)$$

$$v_3 = v_1 * v_2$$

$$v_4 = \cos(v_2)$$

$$(y_1 = v_3, \quad y_2 = v_4)$$

Generally

$$v_j = \varphi_j(v_i)_{i \prec j}$$

$$c_{ji} = \frac{\partial \varphi_j}{\partial v_i} \in \mathbb{R}$$

Jacobians by Automatic Differentiation (AD)

$$\left[y_1 = x_1 * x_2 * \sin(x_1 * x_2), \quad y_2 = \cos(\sin(x_1 * x_2)) \right]^T$$

$$(v_{-1} = x_1, \quad v_0 = x_2)$$

$$v_1 = v_{-1} * v_0 \quad c_{1-1} = v_0 \quad c_{10} = v_{-1}$$

$$v_2 = \sin(v_1) \quad c_{21} = \cos(v_1)$$

$$v_3 = v_1 * v_2 \quad c_{31} = v_2 \quad c_{32} = v_1$$

$$v_4 = \cos(v_2) \quad c_{42} = -\sin(v_2)$$

$$(y_1 = v_3, \quad y_2 = v_4)$$

Generally

$$v_j = \varphi_j(v_i)_{i \prec j}$$

$$c_{ji} = \frac{\partial \varphi_j}{\partial v_i} \in \mathbb{R}$$

Linearized Computational Graph

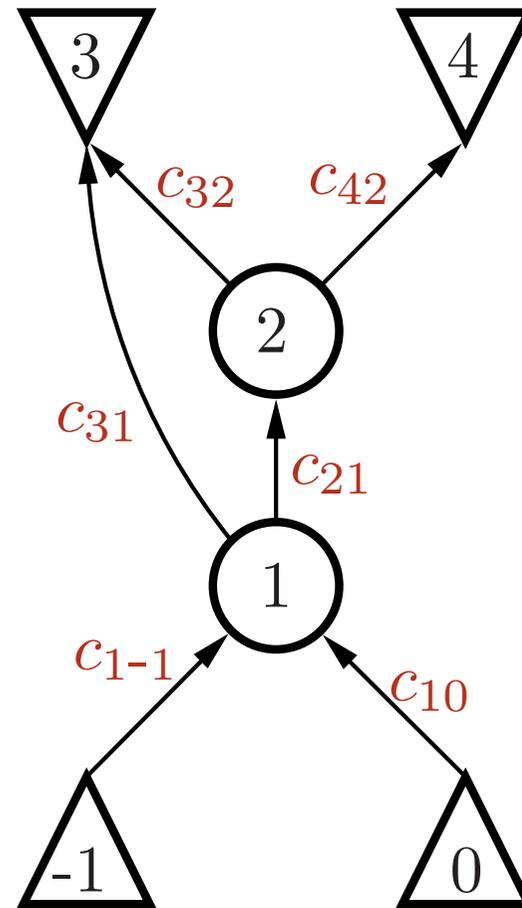
$v_i \prec v_j \Leftrightarrow v_i$ is an argument of φ_j

$$v_1 = \varphi_1(v_{-1}, v_0)$$

$$v_2 = \varphi_2(v_1)$$

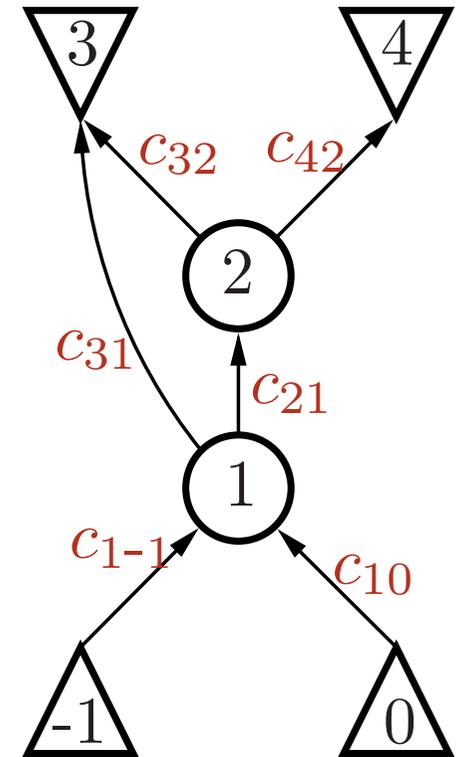
$$v_3 = \varphi_3(v_1, v_2)$$

$$v_4 = \varphi_4(v_2)$$



Baur's Formula (Baur, Strassen 1983)

$$J_{y_l, x_k} = \frac{\partial y_l}{\partial x_k} = \sum_{[x_k \rightarrow y_l] \in \mathbf{G}} \prod_{c_{ji} \in [x_k \rightarrow y_l]} c_{ji}$$



$$J = \begin{pmatrix} c_{1-1}c_{31} + c_{1-1}c_{21}c_{32} & c_{10}c_{31} + c_{10}c_{21}c_{32} \\ c_{1-1}c_{21}c_{42} & c_{10}c_{21}c_{42} \end{pmatrix}$$

The Optimal Jacobian Accumulation Problem

Given an LCG G for the evaluation procedure for some vector function F and a positive integer Ω , is there a sequence of scalar assignments $u_k = s_k \circ t_k$, $\circ \in \{+, *\}$, $k = 1, \dots, \omega$, where each s_k and t_k is either $c_{i,j}$ for some $(j, i) \in E$ or $u_{k'}$ for some $k' < k$ such that $\omega \leq \Omega$ and for every Jacobian entry there is some identical u_k , $k \leq \omega$?

or

Evaluate F' with as few operations ($*$ or $+$) as possible.

The Optimal Jacobian Accumulation Problem

Evaluate F' with as few operations ($*$ or $+$) as possible.

We restrict ourselves to "pure chain rule methods":

- OJA is NP-complete (Naumann 2005). We assume all edge labels algebraically independent (the "structural" problem)
- Don't consider rerouting (Griewank, Vogel 2004)

Accumulation Dags

computational graphs for Jacobians

- Every non-minimal vertex is an operation ($+$ or $*$) and has exactly two predecessors (the operands).
- Minimal vertices are the edge labels in E (the local partials)
- Maximal vertices are Jacobian entries

Each vertex is an operation \Rightarrow We want to find an accumulation dag with a minimum number of vertices

Outline

- Intro to Jacobians by AD
- The Optimal Jacobian Accumulation Problem
- Two-Terminal Series-Parallel Dags and Decomposition Trees
- Our Current View of OJA
- Single-Terminal Series-Parallel Dags and Vertex Elimination
- General Series-Parallel Dags
- Modular Decomposition: a Generalization

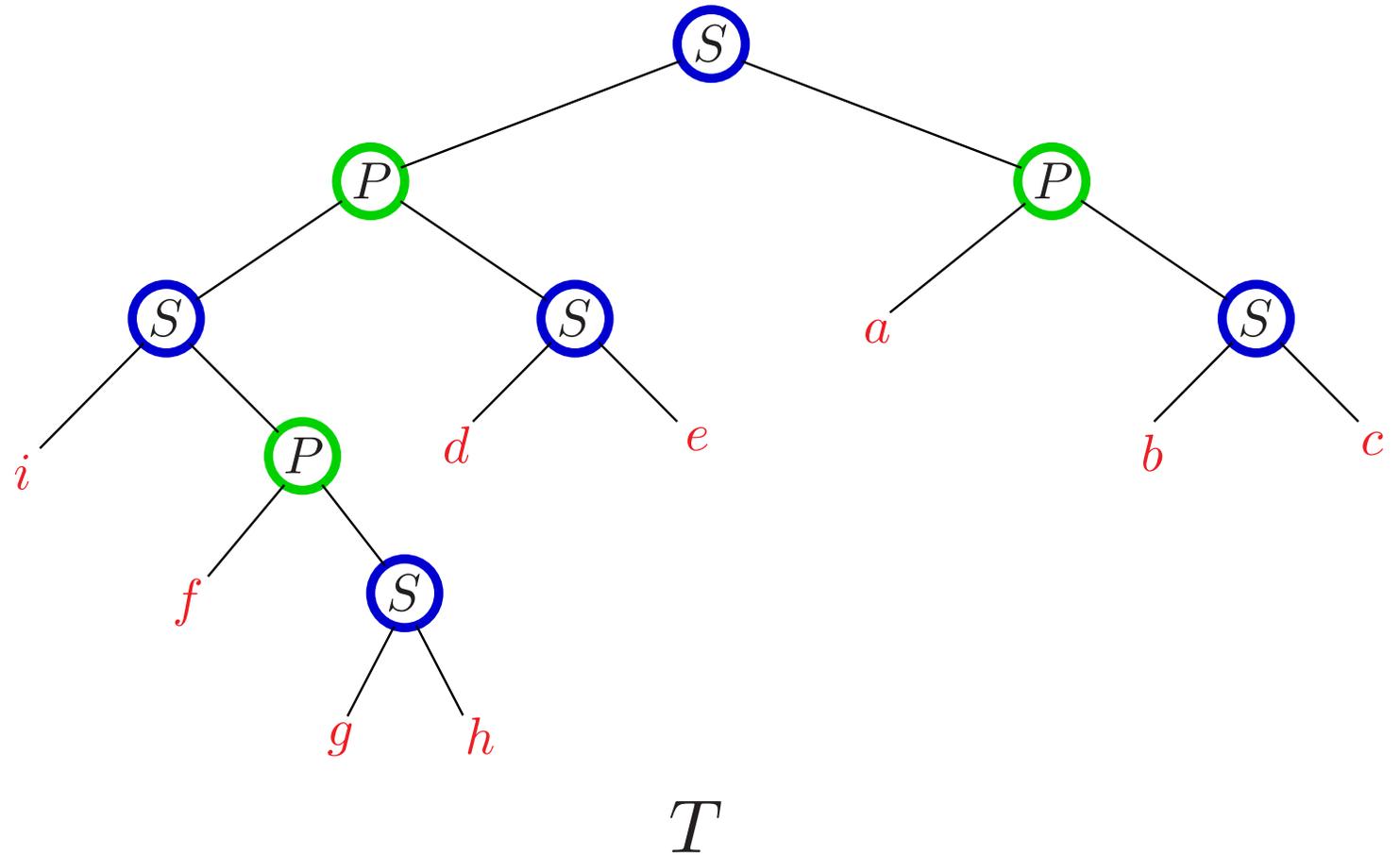
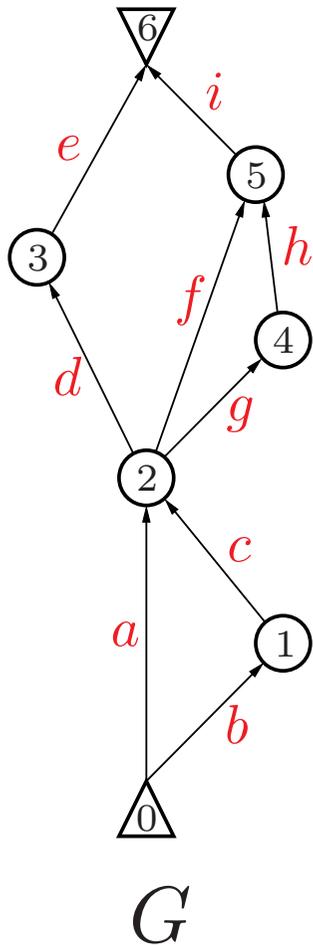
Two-Terminal Series-Parallel dags

Recursive definition:

- An isolated edge is TTSP
- If two graphs G_1, G_2 are TTSP, then so is...
 - *Series composition*: Identifying source of G_1 with sink of G_2
 - *Parallel composition*: Identifying sources, sinks of G_1, G_2

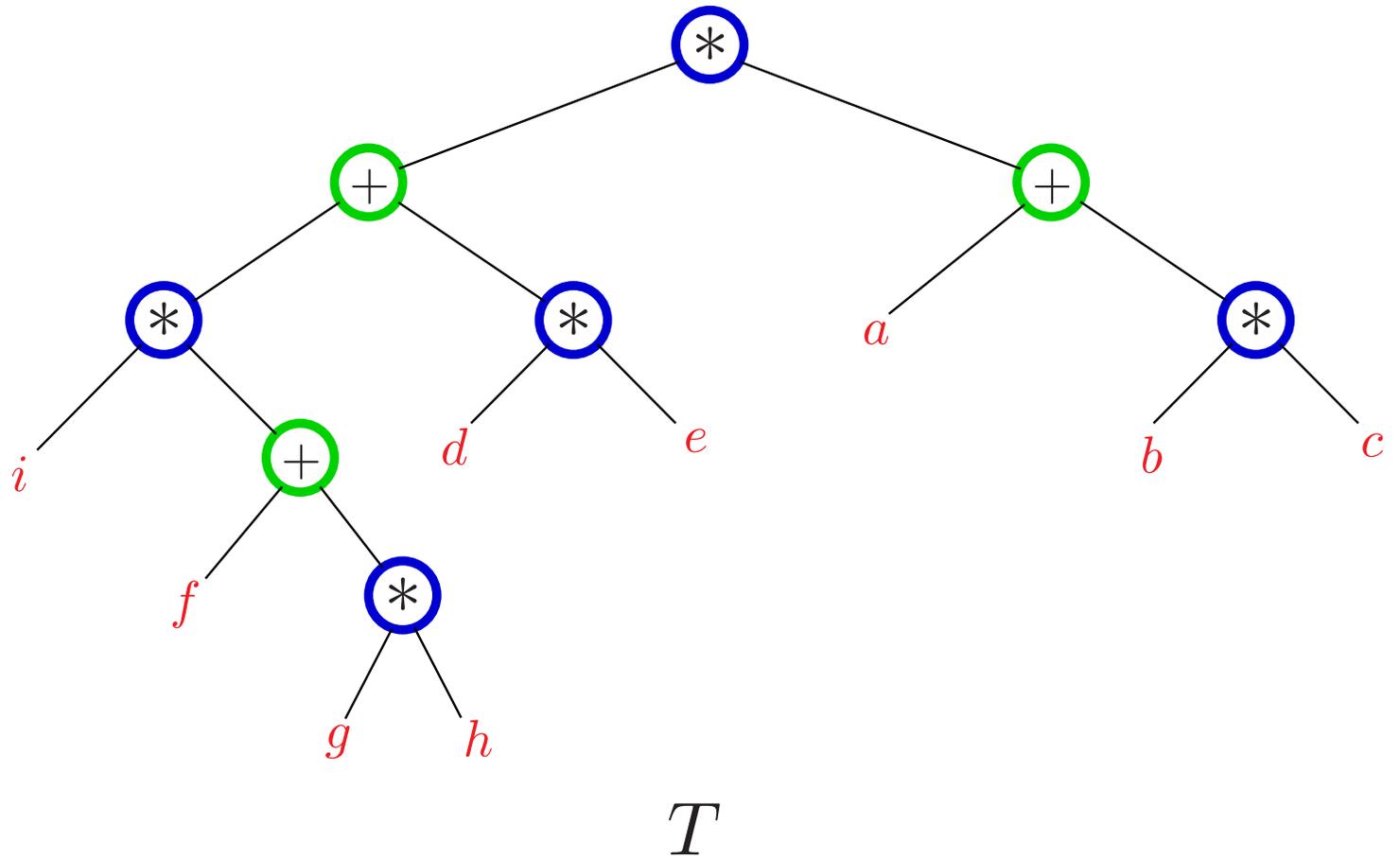
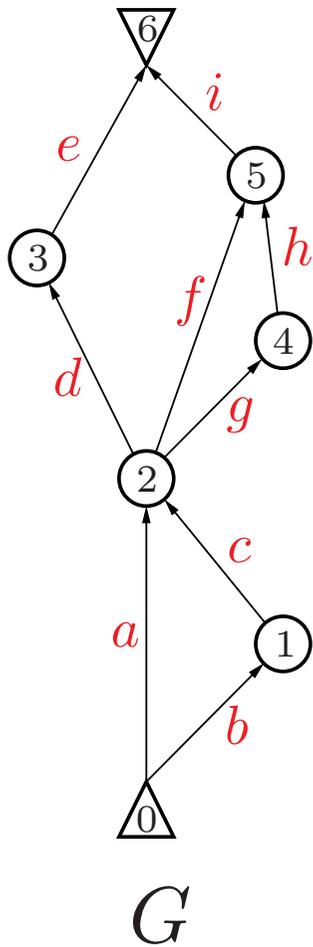
Note: in LCGs, cannot combine two isolated edges in parallel

Decomposition Trees



Binary/canonical versions

Decomposition Trees \rightarrow Accumulation dags



SP Decomposition Solves OJA

Claims:

- The SP decomposition tree computes J
- The SP decomposition tree solves OJA

SP Decomposition Solves OJA

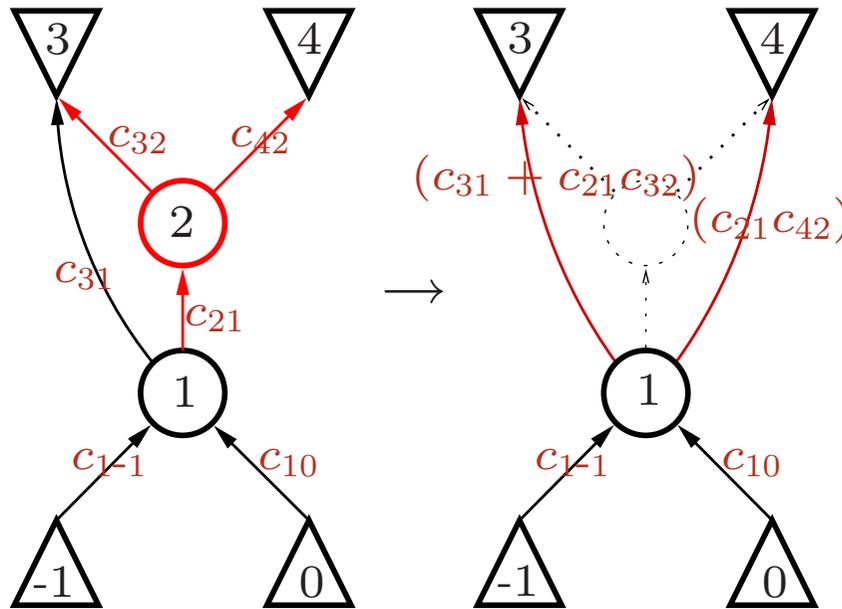
Any (full) binary tree contains $|E| - 1$ non-minimal nodes (operations)

TTSP dags are recognizable in linear time, and the decomposition tree (binary or canonical) can be built in linear time (Valdes, Tarjan, Lawler 1982)

Outline

- Intro to Jacobians by AD
- The Optimal Jacobian Accumulation Problem
- Two-Terminal Series-Parallel Dags and Decomposition Trees
- **Our Current View of OJA**
- Single-Terminal Series-Parallel Dags and Vertex Elimination
- General Series-Parallel Dags
- Modular Decomposition: a Generalization

Vertex Elimination (Griewank, Reese 1991)



$$dv_1 = c_{31} + c_{32} * c_{21}$$

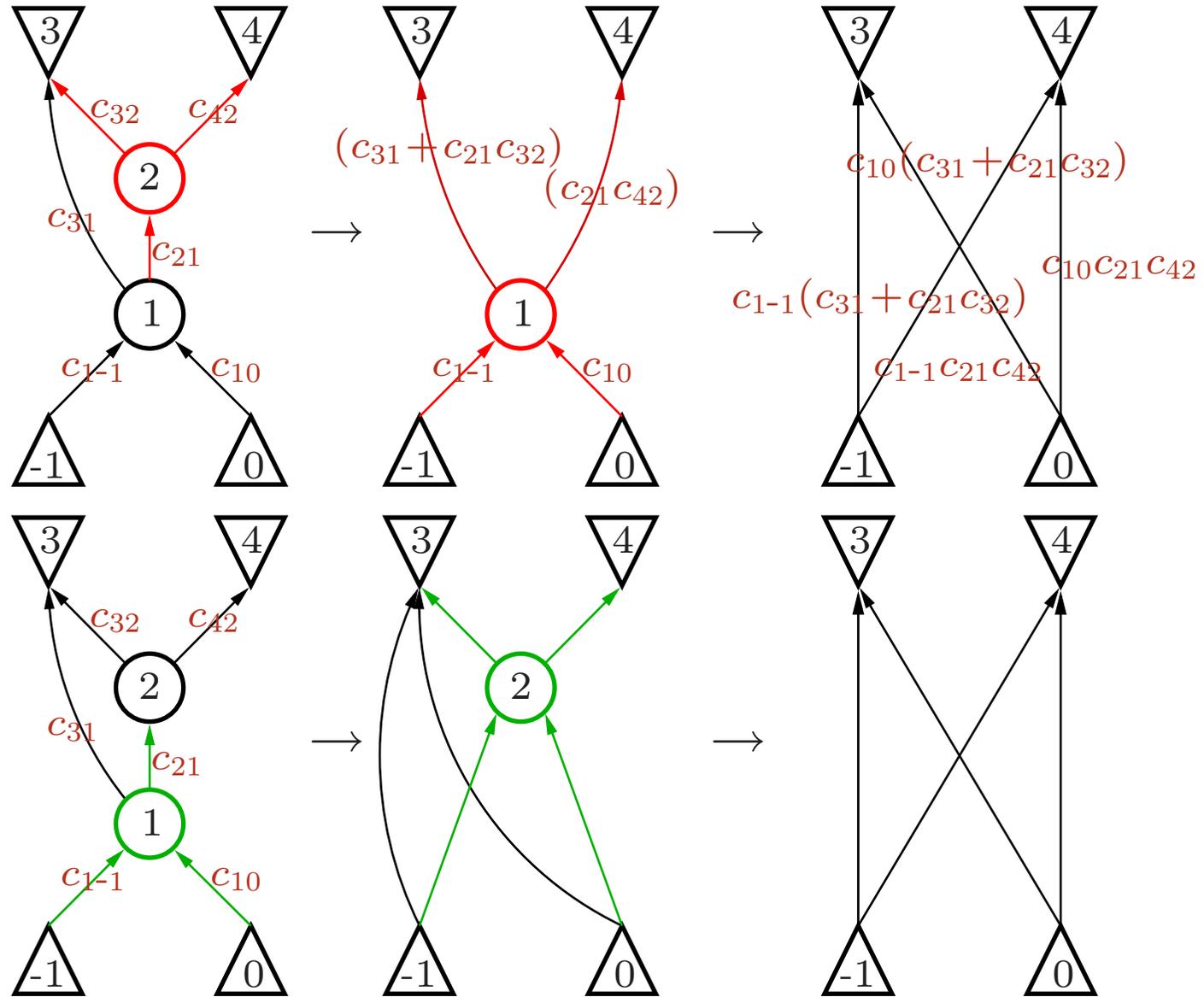
$$dv_2 = c_{21} * c_{42}$$

$$\begin{pmatrix} c_{1-1}c_{31} + c_{1-1}c_{21}c_{32} & c_{10}c_{31} + c_{10}c_{21}c_{32} \\ c_{1-1}c_{21}c_{42} & c_{10}c_{21}c_{42} \end{pmatrix}$$

↓

$$\begin{pmatrix} c_{1-1}(c_{31} + c_{21}c_{32}) & c_{10}(c_{31} + c_{21}c_{32}) \\ c_{1-1}(c_{21}c_{42}) & c_{10}(c_{21}c_{42}) \end{pmatrix}$$

Vertex Elimination (Griewank, Reese 1991)

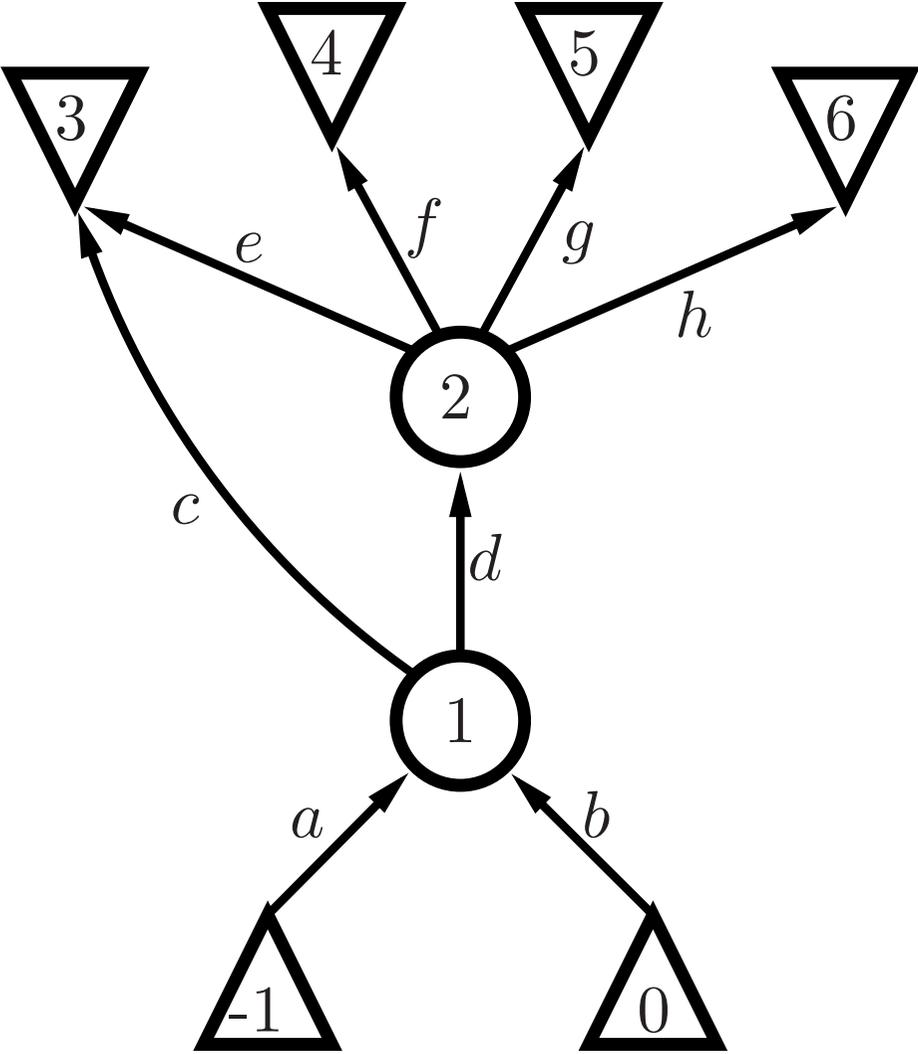


Vertex Elimination (Griewank, Reese 1991)

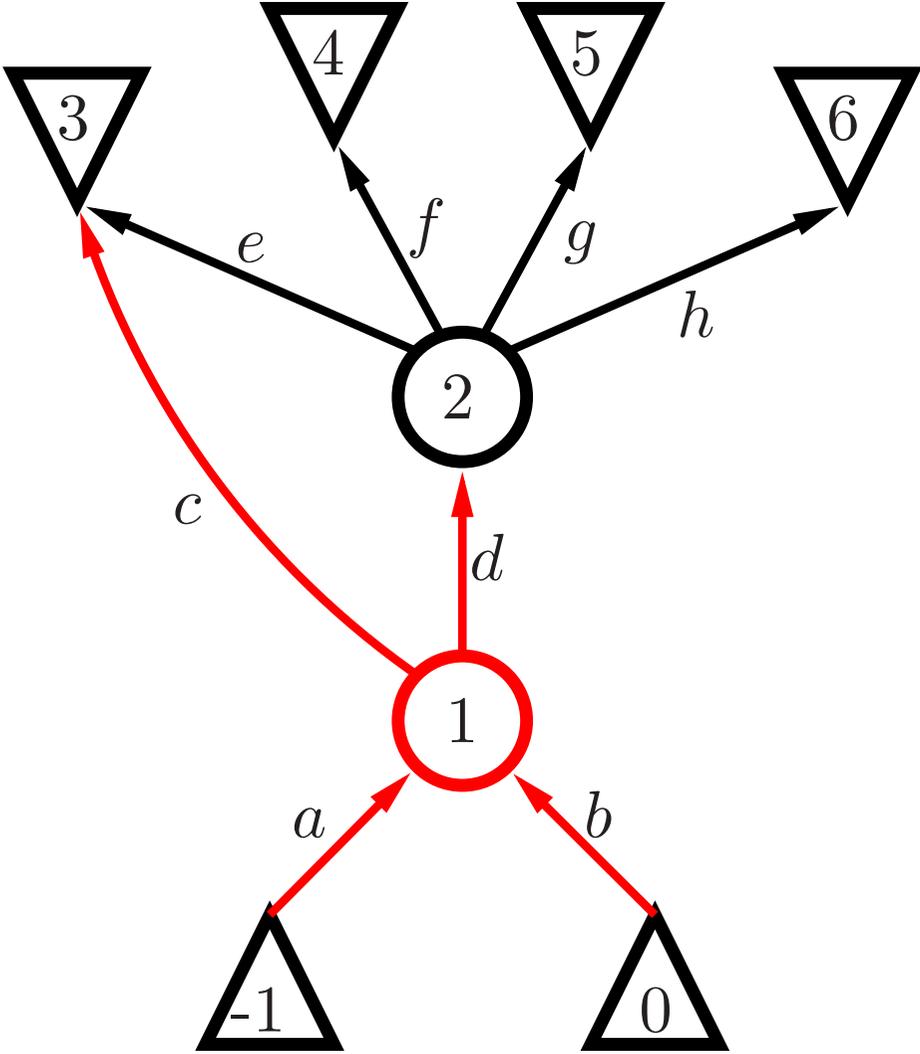
Cost of eliminating a vertex: $|P_v| * |S_v|$ multiplications

Lower bound for vertex elimination: $|\underline{P}_v| * |\underline{S}_v|$ for each v

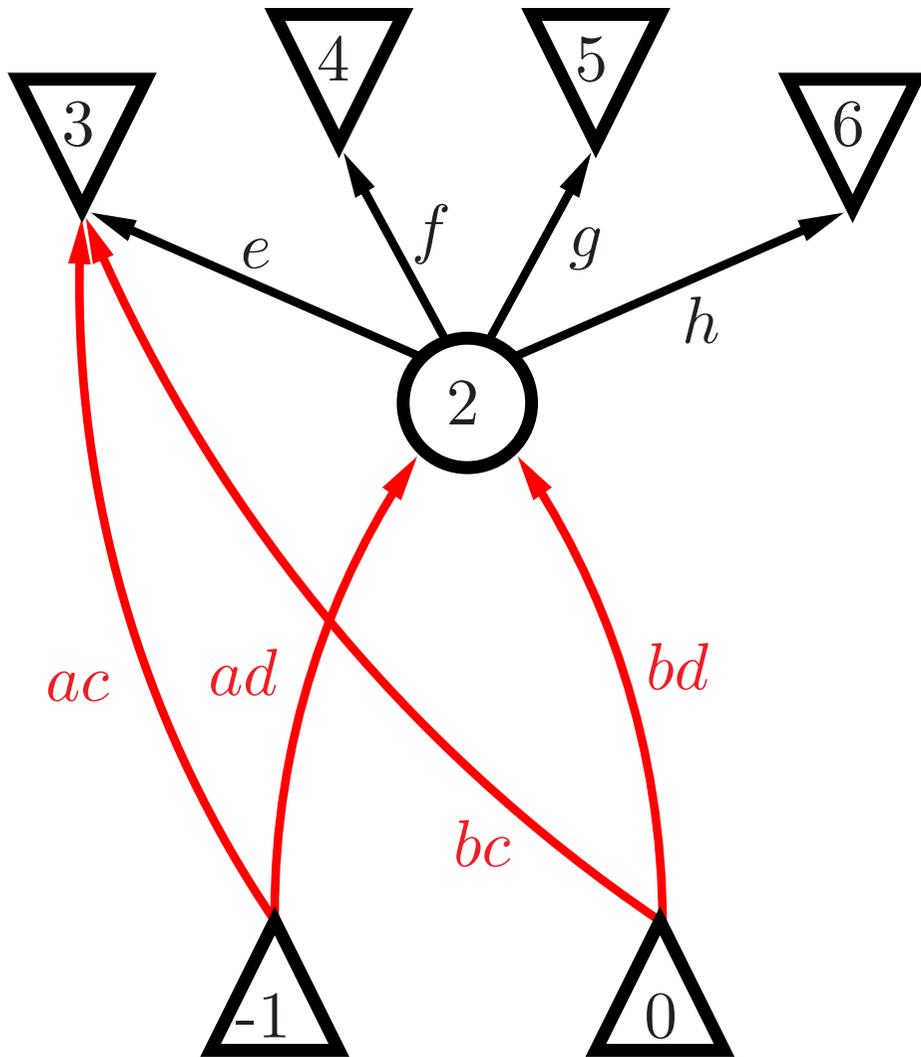
Vertex Elimination



Vertex Elimination

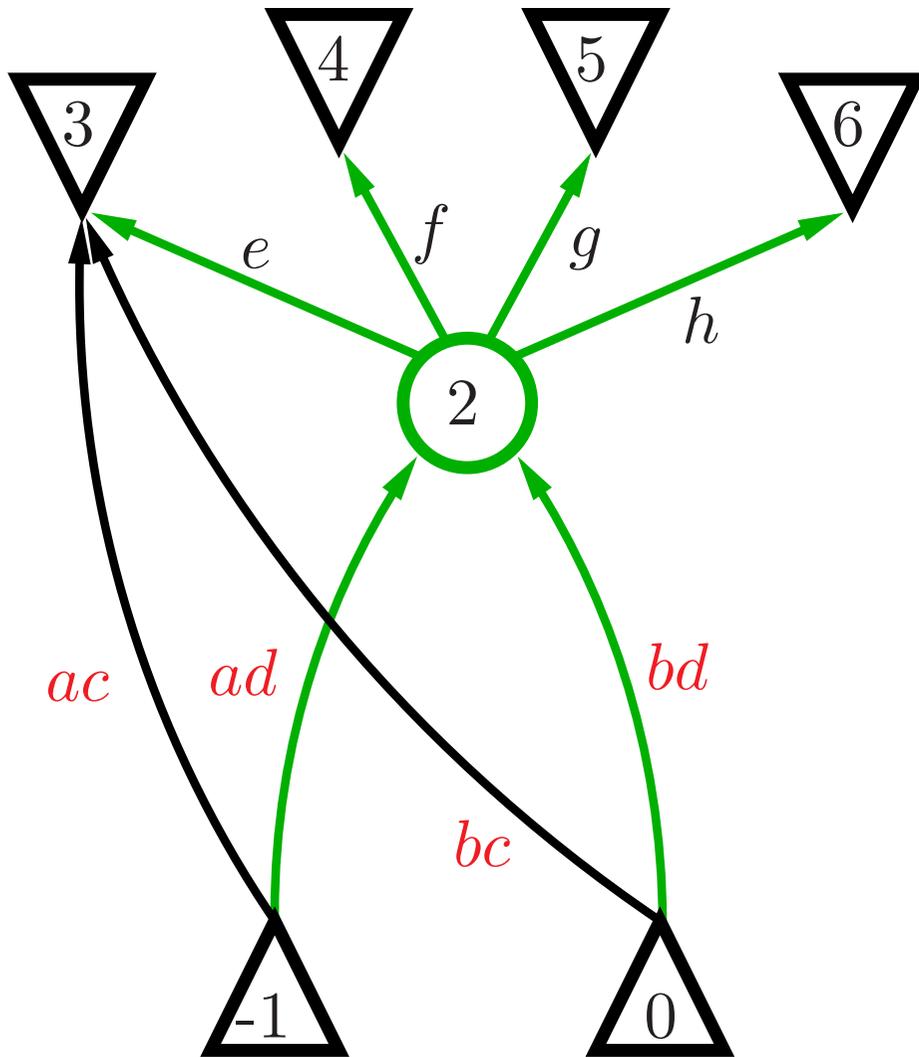


Vertex Elimination



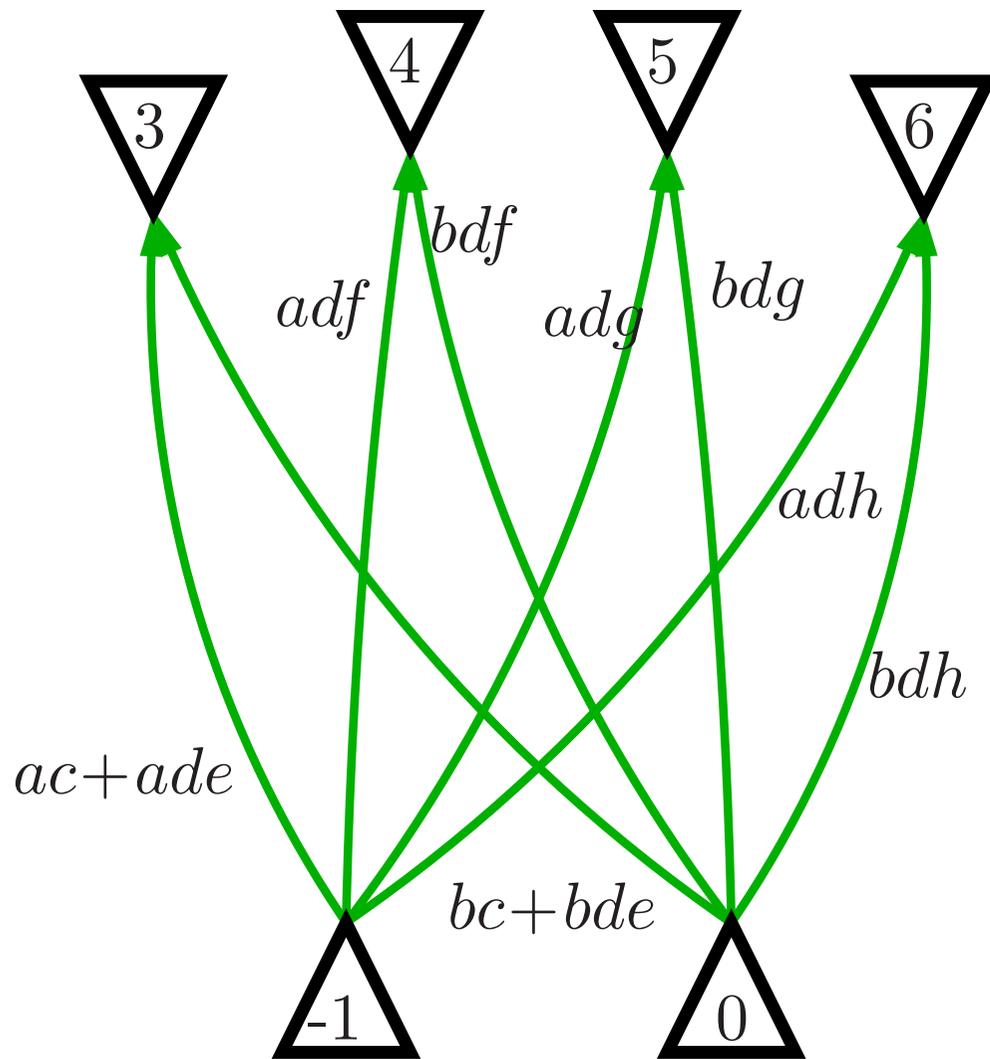
$$ac = a * c$$
$$ad = a * d$$
$$bc = b * c$$
$$bd = b * d$$

Vertex Elimination



$$ac = a * c$$
$$ad = a * d$$
$$bc = b * c$$
$$bd = b * d$$

Vertex Elimination



$$ac = a * c$$

$$ad = a * d$$

$$bc = b * c$$

$$bd = b * d$$

$$ade = ad * e$$

$$(ac + ade) = ac + ade$$

$$adf = ad * f$$

$$adg = ad * g$$

$$adh = ad * h$$

$$bde = bd * e$$

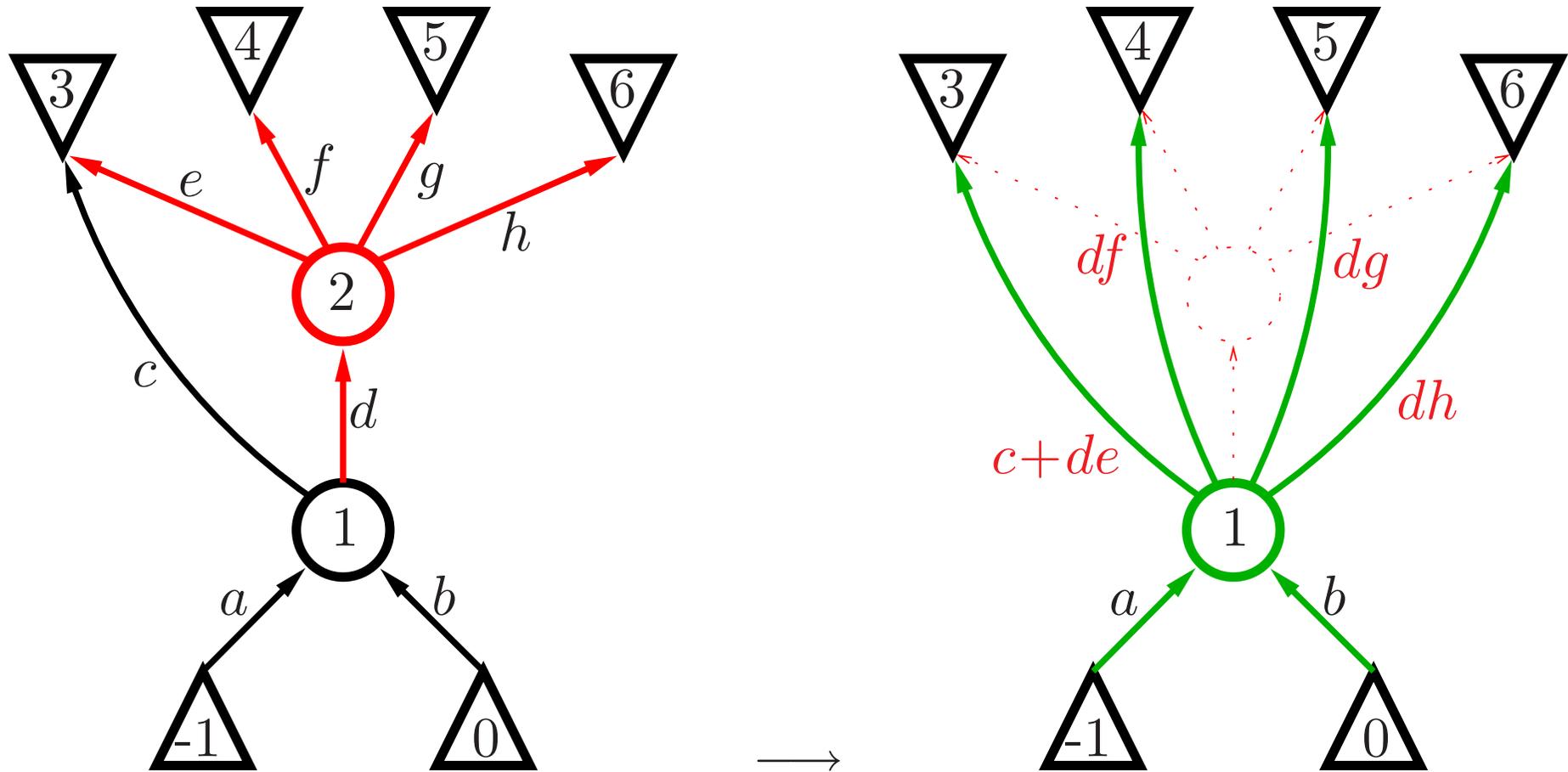
$$(bc + bde) = bc + bde$$

$$bdf = bd * f$$

$$bdg = bd * g$$

$$bdh = bd * h$$

Vertex Elimination



Vertex Elimination

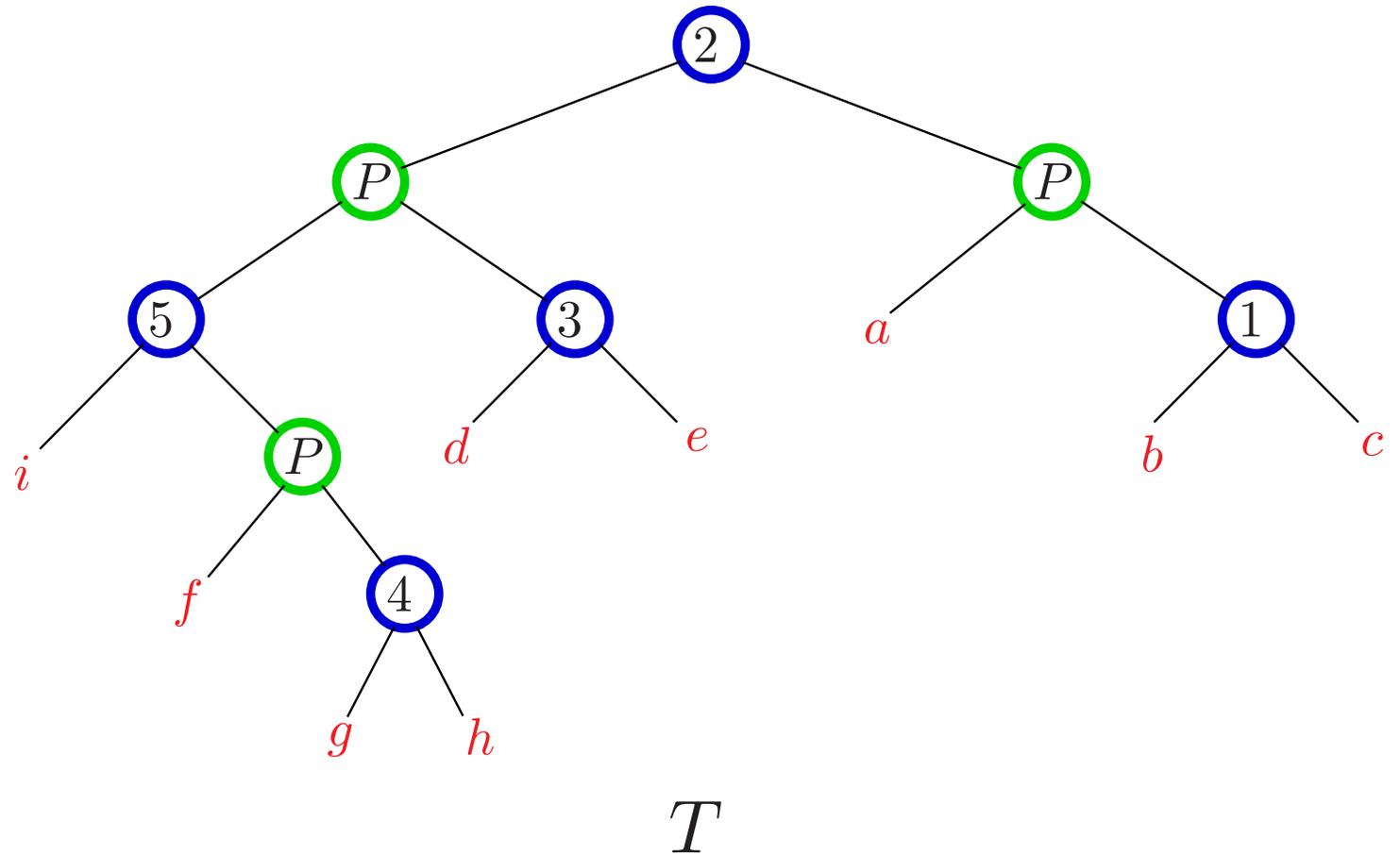
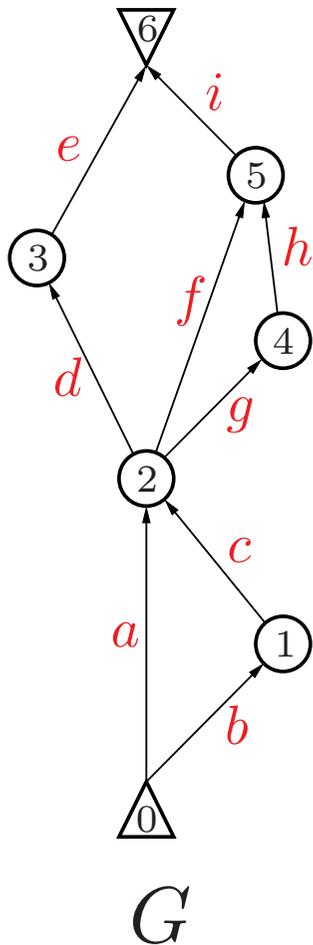
$$\begin{matrix} & v_2, v_1 \\ \left(\begin{array}{cc} a(c + de) & b(c + de) \\ a(df) & b(df) \\ a(dg) & b(dg) \\ a(dh) & b(dh) \end{array} \right) \end{matrix}$$

- Makes *poor* use of common subexpressions
 $(df), (dg), (dh)$
- Exploits distributivity effectively
 $a(c + de), b(c + de)$

$$\begin{matrix} & v_1, v_2 \\ \left(\begin{array}{cc} ac + (ad)e & bc + (bd)e \\ (ad)f & (bd)f \\ (ad)g & (bd)g \\ (ad)h & (bd)h \end{array} \right) \end{matrix}$$

- Makes good use of common subexpressions
 $(ad), (bd)$
- *Doesn't* exploit distributivity
 $ac + ade, bc + bde$

Vertex Elimination in TTSP Dags



Vertex Elimination in TTSP Dags

Every binary decomposition tree corresponds to a collection of vertex elimination sequences

\Rightarrow OJA = OVE for TTSP dags

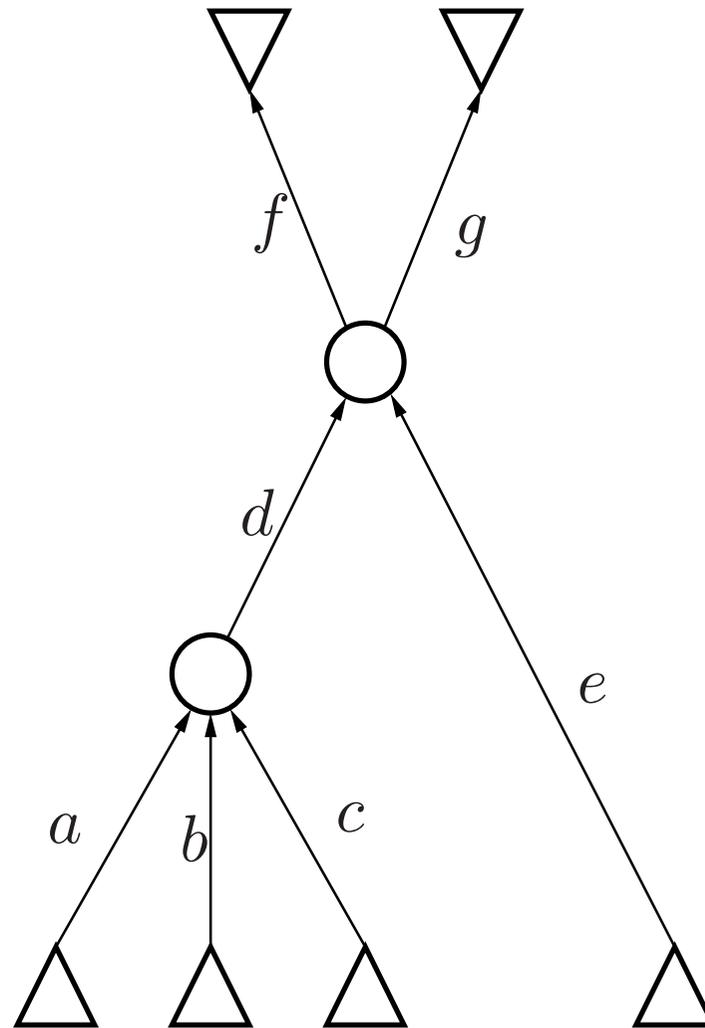
Each vertex eliminated at $|\underline{P}_v| = |\underline{S}_v| = 1$

Generalizing to Scalar Valued Functions

Outline

- Intro to Jacobians by AD
- The Optimal Jacobian Accumulation Problem
- Two-Terminal Series-Parallel Dags and Decomposition Trees
- Our Current View of OJA
- Single-Terminal Series-Parallel Dags and Vertex Elimination
- **General Series-Parallel Dags**
- Modular Decomposition: a Generalization

SP Tree NOT Optimal for General Jacobians



Outline

- Intro to Jacobians by AD
- The Optimal Jacobian Accumulation Problem
- Two-Terminal Series-Parallel Dags and Decomposition Trees
- Our Current View of OJA
- Single-Terminal Series-Parallel Dags and Vertex Elimination
- General Series-Parallel Dags
- **Modular Decomposition: a Generalization**

Generalizing the decomposition: Modules

$$G = (V, E) \rightarrow P = (E, \leq)$$

($a \leq b \Leftrightarrow a$ and b lie on a common path)

Module M :

every $e \in E \setminus M$ is either comparable or incomparable to every $m \in M$

Future Work

- Extracting SP subgraphs from General graphs
- Generalize to modular decomposition
(prove modules can be accumulated in isolation)
- Complexity of OJA by vertex, edge, face elimination (still!)
- Complexity of OJA in general (= face elimination??)