Exploiting Algebraic Dependences Between Local Partial Derivatives in Jacobian Accumulation

Andrew Lyons, lyonsam@gmail.com

Jacobian Accumulation via Automatic Differentiation

Vector function $\mathbf{y} = F(\mathbf{x}) : \mathbb{R}^n \to \mathbb{R}^m$ given as a computer program

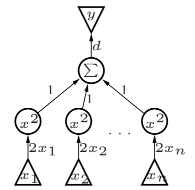
Want to compute Jacobian Matrix $J \equiv F'(\mathbf{x}) = [\frac{\partial y_i}{\partial x_j}]$ using **chain rule**

Example Sum of squares

$$y = \sin(\sum_{i=1}^{n} x_i^2)$$

in context of larger routine

Want $J = (2dx_i)_{i=1,\ldots,n} = 2d(x_i)_{i=1,\ldots,n}$



The Optimal Jacobian Accumulation Problem (OJA)

Assume cost of evaluating J dominated by floating-point multiplications

$\Rightarrow$ Objective is to minimize multiplications in generated derivative code

Directed, acyclic *linearized computational graph* (LCG) $\mathbf{G} = (V, E)$:

- $v_i \in V$ are variables in evaluation of F
- $(i, j) \in E$ are labeled with local partials $c_{ji} = \frac{\partial v_j}{\partial v_i}$

Jacobian entries accumulated in accordance with *Baur's formula*

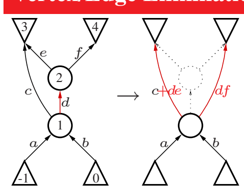
$$J_{y_\ell, x_k} = \sum_{[x_k \to y_\ell] \in \mathbf{G}} \prod_{c_{ji} \in [x_k \to y_\ell]} c_{ji} ,$$

where $[x_k \to y_\ell]$ denotes a path in $\mathbf{G}$ from x_k to y_ℓ

Graph transformations $\leftrightarrow$ exploiting properties of the field $(\mathbb{R}, \cdot, +)$

OJA is NP-complete! (Naumann, 2006) (see talk tomorrow @ 11:00)

Vertex/Edge Elimination: (Griewank, Reese 1991)/(Naumann 1999)



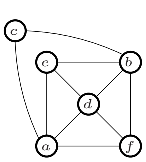
- Edges can be forward- or back-eliminated
- Isolated vertices removed from $\mathbf{G}$
- Vertex elim $\leftrightarrow$ elim. all in-/out-edges of a vertex
- bipartite $\mathbf{G}$ $\leftrightarrow$ Jacobian representation

Algebraic Dependences Between Local Partials

In the methods listed above, we assume that all local partials c_{ji} are algebraically independent. Recent research in AD acknowledges that this is not always the case, and that we can use this fact to our advantage.

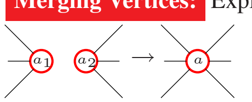
- Naumann's NP-completeness proof relies on this observation

The Path Comparability Graph



- Define (strict) partial order $P = \langle E, <_p \rangle$ on edges in $\mathbf{G}$:
  $a <_p b \Leftrightarrow a$ comes before b in a path in $\mathbf{G}$
- *Path comparability graph* $\mathbf{G}^p = (V^p, E^p)$ defined by
  $\{a, b\} \in E^p \Leftrightarrow a <_p b$ or $b <_p a$
- Maximal paths in $\mathbf{G}$ $\leftrightarrow$ maximal cliques in $\mathbf{G}^p$ $\leftrightarrow$ edges in *clique hypergraph* of $\mathbf{G}^p$
- Sequence of *edge contractions* and *clique merges* reduces $\mathbf{G}^p$ to an edgeless graph, with one vertex for each nonzero Jacobian entry
- Most general Jacobian accumulation model to date (too general??)

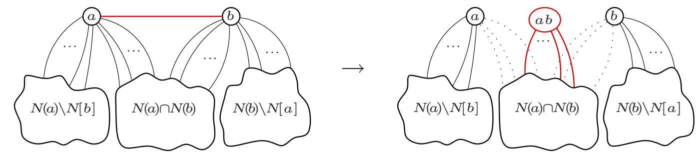
Merging Vertices: Exploiting algebraic dependences



Merge equivalent and non-adjacent vertices a_1, a_2 before beginning accumulation procedure

- New vertex a adjacent to all $v \in V^p : \{a_1, v\} \in E^p$ or $\{a_2, v\} \in E^p$
- Must maintain max. cliques/hyperedges/Jacobian associations

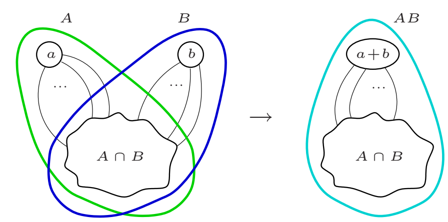
Edge Contractions: multiplications in generated code



Notation: $N(a) = \{v : \{a, v\} \in E^p\}$, $N[a] = N(a) \cup \{a\}$

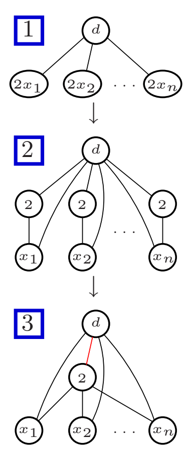
- Create new vertex labeled ab, adjacent to every vertex in $N(a) \cap N(b)$
- All max. cliques/hyperedges that contained both vertices a and b now contain ab instead
- Remove all edges $\{u, v\}$ where $u \in \{a, b\}$ and $v \in N(a) \cap N(b)$
- $N(a) \setminus N[b] = \emptyset \Rightarrow$ isolated vertex
  $\to$ remove a from $\mathbf{G}$ (and vice versa)
- Optimal eliminations when a, b is a module?
  $(N(a) \setminus N[b] = N(b) \setminus N[a] = \emptyset)$

Clique Merges: additions in generated code

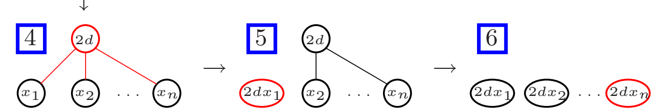


- Performed on cliques A, B (or hypergraph edges) that contribute to the same Jacobian entry
- Require $A = B \setminus \{a\}, B = A \setminus \{b\}$ for some $a \in A, b \in B$
- Can be performed at any time. Best performed immediately?
- Many paths $\Rightarrow$ many clique merges at once (performed pairwise)

Example



1. The path comparability graph $\mathbf{G}^p$ for sum of squares example shown above
2. For $i = 1 \ldots n$, split all vertices $2x_i$ into $2, x_i$
   The new vertices labeled 2 will be equivalent
3. Merge all vertices labeled 2 into a single vertex
4. Contract edge $\{2, d\}$ to create new vertex $2d$
   Vertices 2 and d become isolated
   $\to$ they are removed from $\mathbf{G}^p$
5. Contract edge $\{x_1, 2d\}$ to create vertex $2dx_1$
   (is Jacobian entry $\frac{\partial y}{\partial x_1} = 2dx_1$)
6. Contract remaining edges $\{x_2, 2d\}$ through $\{x_n, 2d\}$ and arrive at the Jacobian representation after $n + 1$ multiplications



Acknowledgements

Uwe Naumann and Jean Utke, for teaching me everything I know about AD

Alex Pothen, Paul Hovland, & the program committee for providing support

Prepared February 2007 for CSC 2007: *The SIAM Workshop on Combinatorial Scientific Computing*, Costa Mesa CA, USA