

A new graph construction for the optimal Jacobian accumulation problem

Andrew Lyons

Computation Institute, The University of Chicago

Mathematics and Computer Science Division, Argonne National Laboratory

lyonsam@gmail.com

Abstract

...

1 Introduction

...

2 The Path Graph

Definition 1 (Path Graph). For a given Linearized Computational Graph (LCG) $\mathbf{G} = ((\mathbf{X}, \mathbf{Y}, \mathbf{Z}), \mathbf{E})$ we define the path graph $\mathbf{G}^p = (V^p, E^p, C^p)$ as follows:

1. The vertex set V^p is equal to the set E of edges in \mathbf{G} (the local partial derivatives $c_{j,i}$ are retained as labels on the elements of V^p).
2. The Edge set E^p is the set of all edges $\{a \in V^p, b \in V^p\}$ such that the corresponding edges $a, b \in E$ lie on a common maximal path in \mathbf{G} (a path from some independent vertex $x \in X$ to some dependent vertex $y \in Y$).
3. The clique set $C^p = \{\mathcal{C}_{y,x}\}_{y \in Y, x \in X}$ comprises entry sets $\mathcal{C}_{y,x}$ defined by

$$\mathcal{C}_{y,x} = \{\{v \in \mathbf{G}^p : v \in [x \rightarrow y]\}\}_{[x \rightarrow y] \in G} \quad .$$

In other words, for every entry $F'_{y \in Y, x \in X} = c_{y,x} = \frac{\partial y}{\partial x}$ of the Jacobian matrix F' there is a set $\mathcal{C}_{y,x} \in C^p$ whose members correspond to the paths from x to y in \mathbf{G} .

Proposition 1. There is a one-to-one correspondence between maximal paths in \mathbf{G} and maximal cliques in \mathbf{G}^p .

Proof. (\Rightarrow) Let P be the set of all edges along any path $[x \rightarrow y] \in \mathbf{G}$. The corresponding vertices $C \subseteq V^p$ in \mathbf{G}^p are pairwise adjacent and thus form a clique. Assume there exists some vertex $v \in V^p$ such that $v \notin C$ and $C \subseteq N(v)$. Then for every $c_i \in C$ there exists a path P_i in \mathbf{G} such that the edges in \mathbf{G} corresponding to both v and c_i lie on P_i .

We have reached a contradiction. So the assumption is false, thus Because no such v could exist, we may conclude that every such clique C is maximal in \mathbf{G}^p .

(\Leftarrow) Let $C \subseteq V^p$ be a maximal clique in \mathbf{G}^p ... this implies a maximal path in \mathbf{G} ...

Let $P_i = (p_1 \in X, p_2, \dots, p_{|P_i|} \in Y)$ □

Observation 1. The graph \mathbf{G}^p is a composition of cliques. Could we gain anything by working with the complement instead?

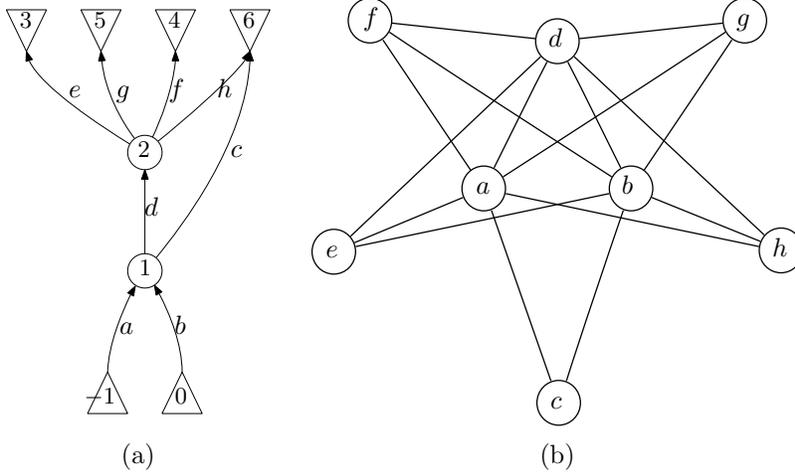


Figure 1: Linearized computational graph \mathbf{G} (a), and the corresponding path graph \mathbf{G}^p (b) for the lion example

2.1 Example

Figure 1 depicts the (LCG) lion graph [5] (a) and the corresponding path graph (b).

The clique set $C^p = \{\mathcal{C}_{y,x}\}_{y \in \{3,4,5,6\}, x \in \{-1,0\}}$ for this example is

$$\begin{aligned} \mathcal{C}_{3,-1} &= \{\{a, d, e\}\} & \mathcal{C}_{4,-1} &= \{\{a, d, f\}\} & \mathcal{C}_{5,-1} &= \{\{a, d, g\}\} & \mathcal{C}_{6,-1} &= \{\{a, d, h\}, \{a, c\}\} \\ \mathcal{C}_{3,0} &= \{\{b, d, e\}\} & \mathcal{C}_{4,0} &= \{\{b, d, f\}\} & \mathcal{C}_{5,0} &= \{\{b, d, g\}\} & \mathcal{C}_{6,0} &= \{\{b, d, h\}, \{b, c\}\} \end{aligned}$$

Definition 2 (set membership). *The set membership $C(v)$ of a vertex $v \in V^p$ is defined by*

$$C(v) = \{C \in \mathcal{C} : v \in C\}_{C \in C^p} \quad .$$

Definition 3 (neighbourhood). *The open neighbourhood $N(v) \subseteq V^p$ of a vertex $v \in V^p$ is defined by*

$$N(v) = \{u \in V^p : \{u, v\} \in E^p\} \quad .$$

The closed neighbourhood $N[v] \subseteq V^p$ of v , is defined by

$$N[v] = N(v) \cup \{v\} \quad .$$

3 The Edge Contraction Operation

The Jacobian matrix F' is peaccumulated by a sequence of *edge contraction* operations on \mathbf{G}^p , each of which is equivalent to a single fused multiply-add (**fma**) operation. The result of a complete sequence of edge contractions is a dual graph $\mathbf{G}^p = (V^p, \emptyset, C^p)$ where $|V^p| = |||$ and every $\mathcal{C}_{y,x} \in C^p$ contains a single set C whose sole member bears the Jacobian entry $F'_{y,x}$.

Rule 1 (edge contraction). *The contraction of any edge $\{a, b\} \in E^p$ is defined by the following:*

1. Remove $\{a, b\}$ from E^p .
2. Create a new vertex $ab \in V^p$ and new edges incident on ab such that $N(ab) = N(a) \cap N(b)$.
3. For all $C \in (C(a) \cap C(b))$, set $C = (C \setminus \{a, b\}) \cup \{ab\}$.
4. Remove all edges that connect a vertex in $\{a, b\}$ to a vertex in $(N(a) \cap N(b))$. If this process leaves either a or b isolated, then remove it from V^p .

5. (clique merge rule) If there exists some entry set $\mathcal{C} \in C^p$ such that there exist sets $C, C' \in \mathcal{C}$ such that $C \setminus C' = \{ab\}$ and $C' \setminus C = \{ab'\}$ for some $ab' \in V^p$, then create a new vertex $(ab' + ab) \in V^p$ and merge cliques for all $\mathcal{C} \in C^p$ as follows. For all $C \in (C(ab) \cap \mathcal{C})$:

- (a) Identify a set $C' \in \mathcal{C}$ such that $C \setminus C' = \{ab\}$ and $C' \setminus C = \{ab'\}$ for some $ab' \in V^p$.
- (b) Create new edges connecting $ab' + ab$ to every member of the set $C \cap C'$.
- (c) Create a new set $C^+ = (C \cap C') \cup \{ab' + ab\}$ and make it a member of \mathcal{C} .
- (d) Remove sets C and C' from \mathcal{C} .
- (e) Remove vertices ab' and ab from V^p .

Lemma 1. Let $\{a, b\}$ be any edge whose contraction calls for one or more clique merge operations. Then the following are true:

- (i) There is no C in any $\mathcal{C} \in C^p$ such that $ab, ab' \in C$ ($C(ab) \cap C(ab') = \emptyset$).
- (ii) There is a unique vertex $ab' \in V^p$ such that ab' is eligible to be merged with ab as above.
- (iii) For any set C which contains ab , there is at most one other set C' that it is eligible to be merged with.
- (iv) Let $\mathcal{C} \in C^p$ be any entry set. Then either every set $C \in (\mathcal{C} \cap C(ab))$ is merged with a corresponding set $C' \in \mathcal{C} \setminus C(ab)$ or none of them is.
(There is a one-to-one correspondence between such sets C and C' in \mathcal{C} .)
- (v) Vertices ab and ab' are isolated by the contraction of $\{a, b\}$.
- (vi) The merging of two cliques cannot result in other cliques becoming mergable.

Proposition 2. Under edge contractions with or without merges, the following properties are preserved:

- The set $\{C \in \mathcal{C}\}_{\mathcal{C} \in C^p}$ is exactly the set of maximal cliques in \mathbf{G}^p .
- \mathbf{G}^p is a composition of cliques.

Proposition 3. Clique merge operations should be performed as soon as possible

Proposition 4 (termination). Any sequence of edge contraction operations that can be applied to a path graph \mathbf{G}^p is finite.

Proof. To begin with, we have that $|V^p|$, $|E^p|$, and $|\{C \in \mathcal{C}\}_{\mathcal{C} \in C^p}|$ (The number of maximal cliques in \mathbf{G}^p) are all finite. Contraction of any edge $\{a, b\}$ with no subsequent clique merging entails the following:

- No change in the number of maximal cliques.
- $|N(a) \cap N(b)|$ new edges are added (all incident on ab).
- $2|N(a) \cap N(b)|$ edges are removed (two from each node adjacent to both a and b).

Thus we have $|(E^p)'| < |E^p|$.

Suppose merge operation(s) are involved:

- Decrease in number of maximal cliques.
- $|N(a) \cap N(b)|$ new edges are added (to ab).
- ...

Because the edge contraction process is complete whenever there are no edges in the graph, and each step reduces the number of edges in the graph ... □

Lemma 2 (structural correctness). Edge contraction operations preserve cliques as maximal, distinct

Lemma 3 (numerical correctness). ...

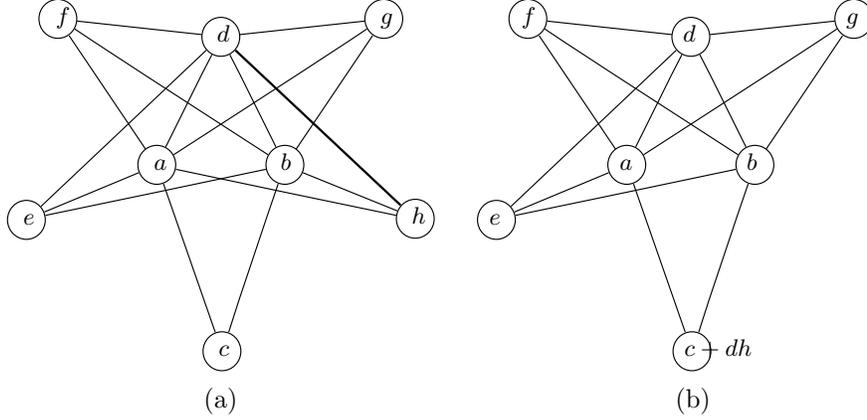


Figure 2: \mathbf{G}^p before (a), and after (b) the contraction of edge $\{d, h\}$

3.1 Example

Consider the contraction of edge $\{d, h\}$ in the lion example. We have $N(d) \cap N(h) = \{a, b\}$, so the new vertex dh is made to be adjacent to both a and b . All edges incident on h are removed, which leaves h isolated, thus h is removed. We now have that $\mathcal{C}_{6,-1} = \{\{a, dh\}, \{a, c\}\}$ and $\mathcal{C}_{6,0} = \{\{b, dh\}, \{b, c\}\}$, and so vertices c and dh must be merged (because $\{a, dh\} \setminus \{dh\} = \{a, c\} \setminus \{c\}$ and $\{b, dh\} \setminus \{dh\} = \{b, c\} \setminus \{c\}$). This leaves us with $\mathcal{C}_{6,-1} = \{\{b, c + dh\}\}$ and $\mathcal{C}_{6,0} = \{\{b, c + dh\}\}$.

3.2 Some Observations on Edge Contraction Operations

Observation 2 (fill). *Consider the contraction of some edge $\{a, b\} \in E^p$. The isolation of a or b constitutes one fill-out each, and the generation of ab is a single fill-in. No fill-in occurs if the newly-generated ab is merged with some ab' . Thus, the possibilities include:*

- 0 fill-out and 0 fill-in (neither a nor b isolated, and merge).
- 0 fill-out and 1 fill-in (neither a nor b isolated, no merge).
- 1 fill-out and 0 fill-in (either a or b isolated, and merge).
- 1 fill-out and 1 fill-in (either a or b isolated, no merge).
- 2 fill-out and 0 fill-in (both a and b isolated, and merge).
- 2 fill-out and 1 fill-in (both a and b isolated, no merge).

At worst, a single new memory slot is required. At best, two slots are made free.

Observation 3. *Consider the contraction of some edge $\{a, b\} \in E^p$.*

- If $\{a, b\}$ is a module in \mathbf{G}^p (meaning $N[a] = N[b]$), then we have $C(a) = C(b)$, and both a and b are isolated by the contraction.
- If $N[a] \subsetneq N[b]$, then $C(a) \subsetneq C(b)$, and only vertex a becomes isolated by the contraction (and vice versa).
- Neither a nor b is isolated if and only if $N[a] \not\subseteq N[b]$ and $N[b] \not\subseteq N[a]$.

Observation 4. *In order to provide for commutivity, vertices could be labelled with multiplication and addition sets rather than sequences*

Observation 5. *Could the path graph be represented as a hypergraph where the cliques are the edges?*

4 Exploiting Algebraic Dependencies

4.1 Simple vertex merges

Because the path graph doesn't have any order to it, and all the information is stored in the cliques, we can perform any graph transformation that preserves the clique information.

Observation 6. *When we merge two vertices, we can no longer guarantee that the clique sets C are exactly the maximal cliques in \mathbf{G}^p .*

For “simple vertex merges“ we require that no clique C in any entry set \mathcal{C} carry more than one vertex with the same label (the cliques are sets, anyway)

Define an equivalence relation \sim on the vertices in V^p .

Rule 2 (simple vertex merges). *For every equivalence class $A \subseteq V^p$ of vertices identified as equal to each other, merge the elements a of A to form a single vertex according to the following:*

1. $N(a+) = \bigcap_{a \in A} N(a)$
- 2.

4.2 Example

Consider the vector function $\mathbf{y} = \mathbf{F}(\mathbf{x})$ defined by $y_1 = x_2 * \sin(\sqrt{x_1})$, $y_2 = x_2 * \sin(\sqrt{x_1})$. We take F to be implemented as code that implies the following linearized code list:

$$\begin{array}{ll}
 c_{1,-1} =; & v_1 = \sqrt{v_{-1}} \\
 c_{2,1} = \cos(v_1); & v_2 = \sin(v_1) \\
 c_{3,1} = -\sin(v_1); & v_3 = \cos(v_1) \\
 c_{4,0} = v_2; & c_{4,2} = v_0; \quad v_4 = v_2 * v_0 \\
 c_{5,0} = v_3; & c_{4,3} = v_0; \quad v_5 = v_3 * v_0
 \end{array}$$

The assignments $v_{-1} = x_1$, $v_0 = x_2$, $y_1 = v_4$, and $y_2 = v_5$ are implied.

\mathbf{G} , the LCG corresponding to F , is shown in Figure 3(a). Figure 3(b) shows the path graph \mathbf{G}^p that corresponds to \mathbf{G} . The clique set $C^p = \{\mathcal{C}_{y,x}\}_{y \in \{4,5\}, x \in \{-1,0\}}$ for \mathbf{G}^p is $\mathcal{C}_{4,-1} = \{\{a, b, d_1\}\}$, $\mathcal{C}_{4,0} = \{\{a, c, d_2\}\}$, $\mathcal{C}_{5,-1} = \{\{e\}\}$, $\mathcal{C}_{5,0} = \{\{f\}\}$.

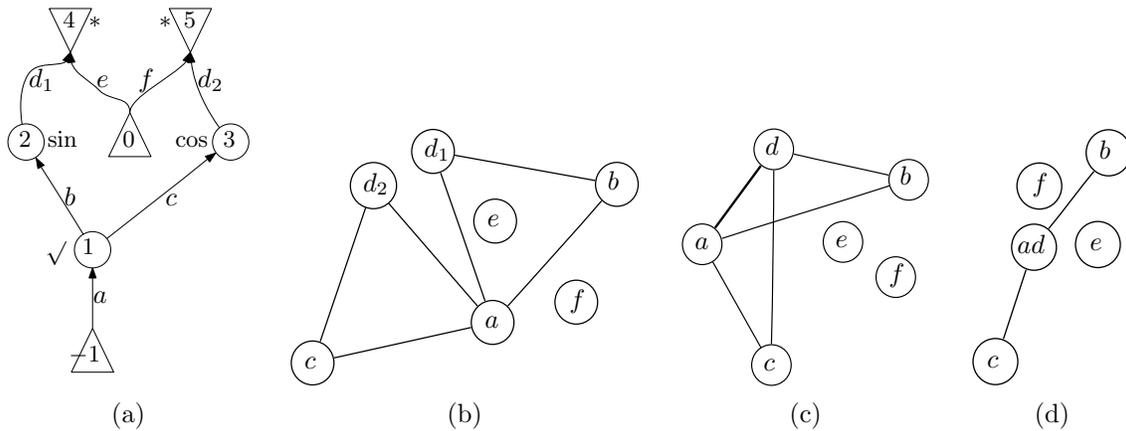


Figure 3: Linearized computational graph \mathbf{G} (a), and the corresponding path graph \mathbf{G}^p (b) for example 2

Now assume the compiler is capable of recognizing the following. The successors of vertex $-2 \in \mathbf{G}$, namely vertices 1 and 2, both correspond to multiplications in the assignment sequence. Vertex 1 has only one other predecessor, -1 , so we have $a_1 = c_{1,-1} = v_{-2}$. By applying the same considerations to vertex 2,

we may observe that $a_2 = c_{2,0} = v_{-2}$. Thus we may conclude that vertices a_1 and a_2 in the path graph \mathbf{G}^p are both members of the same equivalence class $A \subseteq V^p$. Such equivalence relationships between vertices in \mathbf{G}^p can be determined in polynomial time.

If the compiler is capable of performing the aforementioned analysis, then we may merge vertices a_1 and a_2 into a new vertex a according to Rule 2 before any edge contraction operations are performed. The resulting path graph \mathbf{G}^p and is shown in Figure 4(a) and has clique set $C^p \in \mathbf{G}^p$ equal to $\{\mathcal{C}_{5,-2} = \{c_1, e, b_2\}\}, \mathcal{C}_{5,-1} = \{f\}, \mathcal{C}_{5,0} = \{a, e, b_2\}, \mathcal{C}_{6,-2} = \{b_1, d, c_2\}, \mathcal{C}_{6,-1} = \{a, d, c_2\}, \mathcal{C}_{6,0} = \{g\}\}$.

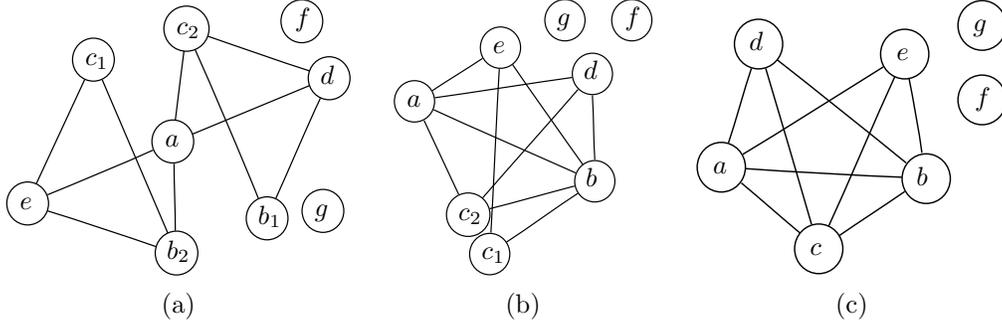


Figure 4: The path graph \mathbf{G}^p corresponding to F after vertices a_1 and a_2 have been merged (a), after vertices b_1 and b_2 have been merged (b), and after vertices c_1 and c_2 have been merged (c)

The assumed compiler analysis will also tell us that we may merge both vertices b_1, b_2 together and c_1, c_2 together to create new vertices b and c , respectively. The results of these merges are shown in Figure 4. The graph in Figure 4(c) has clique set $\{\mathcal{C}_{5,-2} = \{c, e, b\}, \mathcal{C}_{5,-1} = \{f\}, \mathcal{C}_{5,0} = \{a, e, b\}, \mathcal{C}_{6,-2} = \{b, d, c\}, \mathcal{C}_{6,-1} = \{a, d, c\}, \mathcal{C}_{6,0} = \{g\}\}$.

Note that the graphs in Figures 4(a) and 4(b) both have ten edges. The same is true of the original path graph in Figure ??(b). However, the graph in Figure 4(c) shows that the merging of c_1 and c_2 reduces the number of edges in \mathbf{G}^p by one. In general, merging two vertices a_1 and a_2 reduces the number of edges in \mathbf{G}^p by the number of neighbors they have in common, or $|N(a_1) \cap N(a_2)|$.

The edge contraction sequence ($\{$

4.3 more on algebraic dependences

Theorem 1 (NP-completeness). *Ensemble Computation is equivalent to Jacobian preaccumulation by edge contractions (after simple vertex merges have been performed for all equivalence classes) on distinct paths (maximal cliques where $|\mathcal{C}| = 1$ for all $\mathcal{C} \in C^p$) where each path comprises distinct vertices.*

Proof. The set objects in ensemble computation mean that no edge label can occur twice on the same path. The cliques are the sets, and edge contraction (note that there are no merges) corresponds to set union. \square

Observation 7. *general edge dependencies (hessians, related partials) could be exploited by a few simple modifications to the contraction procedure? (it gets messy?) (see handwritten example?)*

Observation 8. *Any LCG vertex that is multiplied means an edge carries that label. If it is multiplied with n other variables, then there is an equivalence class of cardinality n that is a subset of \mathbf{G}^p .*

5 Further thoughts

1. It may be useful to label edges in E^p with the number of cliques they occur in.
2. What are vertex/edge/face elimination in terms of edge contraction? Vertex is contracting an entire bi-clique, and edge is contracting a bi-clique where one partition has 1 element. . . but not quite. Order information is LOST in the path graph

3. Is there a sub-class of paths graphs that are mappable to LCGs, or dual graphs? how are they characterized? (composition of bi-cliques rather than cliques?)
4. We also lose information about when edges come into the picture during the evaluation of the code. A topological sort wont help. Number vertices in \mathbf{G}^p by the time their source vertex in \mathbf{G} comes active in the code? but then there's lots of orderings.
5. What is it about the restrictions on vertex/edge that make them so amenable to linear-algebra operations (Gaussian elimination, Schur complement, matrix chaining, etc. . .)?

6 Conclusions

Bauer's formula:

$$F'_{y,x} = \sum_{[x \rightarrow y] \in \mathbf{G}} \prod_{c_{j,i} \in [x \rightarrow y]} c_{j,i} \quad , \quad (1)$$

See ([1],[2],[4],[5],[3],[6]).

References

- [1] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Freeman, San Francisco, 1979.
- [2] Andreas Griewank. On automatic differentiation. In Masao Iri and Kunio Tanabe, editors, *Mathematical Programming: Recent Developments and Applications*, pages 83–108. Kluwer Academic Publishers, Dordrecht, 1989.
- [3] Andreas Griewank. *Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation*. Number 19 in Frontiers in Appl. Math. SIAM, Philadelphia, PA, 2000.
- [4] Andreas Griewank and Shawn Reese. On the calculation of Jacobian matrices by the Markowitz rule. In Andreas Griewank and George F. Corliss, editors, *Automatic Differentiation of Algorithms: Theory, Implementation, and Application*, pages 126–135. SIAM, Philadelphia, PA, 1991.
- [5] U. Naumann. *Efficient Calculation of Jacobian Matrices by Optimized Application of the Chain Rule to Computational Graphs*. PhD thesis, Technical University of Dresden, December 1999.
- [6] Uwe Naumann. Optimal accumulation of Jacobian matrices by elimination methods on the dual computational graph. *Math. Prog.*, 99(3):399–421, 2004.