Preprint
UCRL-JC-150789

Level-based Incomplete LU Factorization: Graph Model and Algorithms

David Hysom and Alex Pothen

This article was submitted to:
SIAM Journal On Matrix Analysis and Applications

November 2002

U.S. Department of Energy

Lawrence Livermore National Laboratory

University of California
Lawrence Livermore National Laboratory
Technical Information Department
Livermore, CA 94551

LEVEL-BASED INCOMPLETE LU FACTORIZATION: GRAPH MODEL AND ALGORITHMS*

DAVID HYSOM† AND ALEX POTHEN‡

Abstract. A graph theoretic process that models level-based, incomplete LU factorization ($ILU(\ell)$) of sparse unsymmetric matrices is developed. The model leads to two *incomplete fill path theorems* that are generalizations of the original fill path theorem of Rose, Tarjan, and Lueker. Our *S-level incomplete fill path theorem* leads to the development of new, embarrassingly parallel algorithms for computing the structure and storage requirements of $ILU(\ell)$ factors.

Key words. ILU-factorization, graph theory, parallel algorithms

AMS subject classifications. 05C50, 05C70, 05C75, 68W10

1. Introduction. Incomplete *LU* factorization (ILU) is widely recognized as an effective method for preconditioning iterative sparse linear system solvers. In general, ILU algorithms fall into one of two categories:

(i) threshold based ILUT methods;

(ii) structure based ILU(ℓ) methods.

In ILUT methods, the locations of permissible fill entries are determined in conjunction with numeric factorization. Fill entries are permitted only if they are larger than a specified value, and an upper limit may be placed on the number of entries permitted in a row. In contrast, ILU(ℓ) methods are separated into two phases. In the first phase, which is often referred to as *symbolic factorization*, the locations of permissible fill entries are determined. Each potential fill entry is assigned a level, and an entry is permitted in the factor if its level is not greater than ℓ. In the second phase, numeric factorization is performed.

In this work we are concerned with the symbolic phase of ILU(ℓ) factorization. We present a graph theoretic process that models the action of existing ILU(ℓ) algorithms. The model is used to develop two *incomplete fill path theorems* that are generalizations of the original fill path theorem that characterizes fill in complete *LU* factors [13, 14].

This paper is organized as follows. § 2 contains essential background material. This includes basic notation; a description of existing ILU(ℓ) factorization algorithms; a description of the two rules (*sum* and *max*) that are commonly used to compute fill levels; and a review of the original fill path theorem for complete factorization.

In § 3 we present a graph theoretic construct that models the operation of existing ILU(ℓ) factorization algorithms.

In § 4 we use our model to prove theorems concerning structural properties of ILU(ℓ) factors. Results include the *S-level* and *M-level* incomplete fill path theorems, which describe relationships between a fill entry's level and paths in graphs. The

*This work was performed under the auspices of the U.S. Department of Energy by University of California Lawrence Livermore National Laboratory under contract number W-7405-Eng-48. UCRL-JC-150789. This research was supported in part by NSF grant DMS-9807172, by DOE SCIDAC grant DE-FC02-01ER25476, by a GAANN fellowship to the first author from the Department of Education, and by NASA under contract NAS1-19480 while the second author was in residence at ICASE.

†Center for Applied Scientific Computing, Lawrence Livermore National Laboratory, Livermore, CA, 94551 (email: hysom@llnl.gov). (`hysom@llnl.gov`).

‡Department of Computer Science, Old Dominion University, Norfolk VA 23529-0162 (`pothen@cs.odu.edu`); and ICASE, NASA Langely Research Center, Hampton VA 23681-0001.

S-level theorem characterizes fill that is computed using the sum rule; the M-level theorem characterizes fill computed using the max rule.

In § 5 we show how the S-level incomplete fill path theorem can be used to develop novel, embarrassingly parallel algorithms for performing ILU(ℓ) symbolic factorization. We show how the algorithm can be modified to compute ILU(ℓ) storage requirements in $O(n)$ space, where n is the number of rows in a matrix. We include summary examples that demonstrate how the S-level theorem can be used to analytically bound fill amounts for certain classes of graphs.

In § 6 we offer some concluding remarks.

2. Background.

2.1. Preliminaries.

A directed graph of a matrix $G(A) = (V, E)$ has vertex set V, which contains a vertex for every row in the matrix, and edgeset E, which contains a directed edge $\langle i, j\rangle$ for every nonzero entry a_{ij}. An edge $\langle i, j\rangle$ is directed from i to j. $G(A)$ is an ordered graph: if matrix row i is numbered less than matrix row j, then vertex i is ordered before vertex j. The term *filled matrix* denotes the matrix $F = L+U-I$, where L and U are factors (either complete or incomplete) of A. The symbol F includes symmetric as well as unsymmetric cases. If A is symmetric, $F = L + L^T - I$, in which case $U = L^T$.

During ILU(ℓ) symbolic factorization, matrix entries are associated with integer valued *levels*. The level associated with a matrix entry f_{ij} or a_{ij}, and by extension with a graph edge $\langle i, j\rangle$, is denoted $level(i, j)$.

2.2. Classical ILU(ℓ) factorization.

Figure 2.1 contains a statement of the symbolic row-oriented Classic-ILU algorithm. By *classical ILU(ℓ)* we refer to a family of widely known and implemented algorithms that compute the structure of ILU(ℓ) factors by mimicking direct factorization (see, for example, [15]). As in direct factorization, in their outermost loops classical ILU(ℓ) algorithms may iterate over matrix rows, columns, or diagonal entries. Multifrontal approaches are also possible. Since many scientific codes are equation oriented, we restrict our attention to row-oriented algorithms.

Row-oriented ILU is said to be *upward looking*. That is, for every nonzero entry f_{ih} with $h < i$, row i is updated by merging in the upper-triangular portion of the previously factored row h (Figure 2.1 Step 9). During this process a matrix entry f_{ij}, whose value was previously zero, may become nonzero (i.e, may “fill in”) if there exists a nonzero entry f_{hj}. We say the fill entry f_{ij} is *caused by* the existence of the two entries f_{ih} and f_{hj}.

ILU(ℓ) algorithms determine permitted fill based on the concept of a matrix entry’s *level*. Matrix entries in F that correspond to nonzero entries in A are assigned the level zero (Figure 2.1, Step 6). A potential fill entry f_{ij} is assigned a level based on the levels of its two causative entries (Figure 2.1, Step 11). If the assigned level is not greater than ℓ, the entry is admitted to the factor (Figure 2.1 Step 12). A fill entry may have several pairs of causative entries, and hence be assigned several levels. The convention (Figure 2.1, Step 17) is to assign a fill entry the smallest possible level.

Two rules appear in the literature for assigning levels to fill entries. We refer to these as the *sum* rule and the *max* rule. Assuming f_{ij} is caused by previously admitted entries f_{ih} and f_{hj}. the sum rule states

$$level(i,j) = \min_{1 \le h < \min\{i,j\}} \{level(i,h) + level(h,j) + 1\}. \tag{1}$$

In words, this rule assigns a level which is the sum of the level of two causative entries, incremented by 1. All ILU(ℓ) implementations of which the authors are aware make use of the sum rule when assigning levels during factorization.

The sum rule is attractive since, by this rule, an entry's level is a direct indication of the minimum number of times any of its updates will be divided by a pivot value during the numeric factorization phase.

The max rule for level assignment states

$$level(i,j) = \min_{1 \le h < \min\{i,j\}} \max\{level(i,h), level(h,j)\} + 1. \tag{2}$$

In words, this rule assigns a level which is the largest of of the two causative entries, incremented by 1.

The max rule is intuitively appealing (particularly to computer scientists) due to its recursive flavor. To compute an ILU(ℓ) factor using the max rule, one can perform an ILU(1) factorization ℓ times. The input for the first iteration is the structure on the initial matrix A. The input for subsequent iterations is the structure computed during the previous iteration, with fill levels for all entries set to level zero.

The `computeWt` function in Figure 2.1, Step 11, can be implemented to enforce either the sum or max rules. The sum rule form of the function is

```
computeWeight(level(i,h), level(h,j))
{
    return level(i,h) + level(h,j) + 1
}
```

The max rule form of the function is

```
computeWeight(level(i,h), level(h,j))
{
    return max{level(i,h), level(h,j)} + 1
}
```

To distinguish between levels computed using the max or sum functions, we sometimes write "S-level" or "M-level" in place of the more general term, "level." Similarly, we may write S-level(i,j) or M-level(i,j) instead of level(i,j).

The origins of the sum and max rules are difficult to pin down precisely because the foundational ideas were developed gradually, over many years. Historically, incomplete factorization structures were first specified by considering gridpoint operators associated with discretizations of PDEs on structured grids; a concise review of these developments can be found in [2].

The term "incomplete factorization" appears to have been coined by Meijerink and Van der Vorst [11]. A discussion of recursive factorization, which can be shown equivalent to the max rule, is presented by Axelsson [1], who attributes its origin to Gustafsson [7]. The first clear statement of the sum rule that we have been able to locate was enunciated by D'Azevedo, Forsyth, and Tang [4].

```
Classic-ILU(A, ℓ)
# Loop over rows
for j = 1 to n
    # Initialization phase: admit entries in A, and assign them the level zero.
    adj'(j) ← ∅
    for t ∈ adj(j)
        level(j, t) ← 0
        insert t in adj'(j)
    # Row-merge update phase
    for each unprocessed i ∈ adj'(j) with i < j in ascending order
        for t ∈ adj'(i) with t > i
        wt = computeWeight(level(j, i), level(i, t))
            if wt ≤ ℓ
                if t ∋ adj'(i)
                    insert t in adj'(j)
                    level(j, t) ← wt
                else
                    level(j, t) ← min{level(j, t), wt)}
```

Fig. 2.1. Classic-ILU *algorithm. The input matrix A contains n rows. The structure of a row* a_{j*} *is represented by the list* $adj(j)$*. The structure of a factor row* f_{j*} *is represented by the list* $adj'(j)$*.*

2.3. The fill path theorem for complete factorization. Parter [12], and later Rose, Tarjan, and Leuker [13, 14], developed graph theoretic vertex elimination processes that model complete Gaussian Elimination. One of the highlights of this body of work was the development of a *fill path theorem* that provides a static characterization of fill in F. By *static* we mean that the theorem permits one to determine the location of all fill entries in F by examining paths in the graph $G(A)$. The definition of a *fill path*, and a statement of the original fill path theorem follow.

Definition 1. *A* fill path *is a path joining two vertices i and j, all of whose interior vertices are numbered lower than the end vertices i and j.*

Theorem 2. *[13, 14] Let $F = L + U - I$ be the filled matrix corresponding to the complete factorization of A. Then $f_{ij} \neq 0$ if and only if there exists a fill path joining i and j in the graph $G(A)$.*

Application of Theorem 2 has resulted in the gradual development of the notion of elimination trees ([10] provides a review with many references), elimination dags [6], and many algorithms of practical importance for direct methods. For symmetric problems $A = LL^T$, algorithms exist for computing the structure of L in time proportional to the number of nonzeros in the factor [10]. For unsymmetric problems, $A = LU$, computing the structure of the L and U factors is less optimal. However, here also application of the fill path theorem has resulted in several interesting algorithms that, in practice, work quite effectively [5, 6, 13].

3. Graph Theoretic ILU(ℓ) Model.

The *partial elimination process* is a sequence of graphs that models Gaussian elimination. The initial graph in the sequence, G_0, is identical to the graph of the matrix, $G(A) = (V, E)$, where all edges are associated with the level zero. We assume the vertex set V contains n vertices. The graph G_{i+1}, for $0 < i \leq n$, is formed by examining all pairs of directed edges in G_i

that are directed paths of length two, and are of the form i, h, j, with $h < \min\{i, j\}$. For each such path $P(i, j)$, a directed edge $\langle i, j\rangle$ is inserted in E_{i+1} if and only if `computeWeight`(level(i, h), level(h, j)) is not greater than ℓ. If the hypothetical edge $\langle i, j\rangle$ has already been inserted, its weight is adjusted to the minimum of its present weight and the newly calculated weight. Hence, we denote the partial elimination process as

$$G(A) = G_0, G_1, G_2 \ldots, G_n = G_*.$$

For specificity, we use a superscript "S" to indicate when edge weights were calculated using the sum rule, e.g., G_*^S. Similarly, a superscript "M" indicates that edge weights were calculated using the max rule, e.g., G_*^M.

This partial elimination process models ILU(ℓ) factorization since matrix fill entries created or updated when row i is factored correspond exactly to edges inserted or updated during the formation of graph G_i. Hence we have $G_* = G(F)$.

The sequence of graphs defined above differs from previous models for complete factorization in one important aspect. The models for complete factorization are based on bordering methods, in which outer-product updates are performed while marching down a matrix's diagonal. Accordingly, one vertex is eliminated (removed) from V_i during the formation of graph $G_{i+1} = (V_{i+1}, E_{i+1})$ from graph $G_i = (V_i, E_i)$.

Row oriented factorization requires that we leave the vertex set intact, i.e, V_i and V_{i+1} are identical for $0 \le i < n$. While one can as easily formulate a graph theoretic construct based on bordering for incomplete factorization, such a construct would not model the operation of the Classic-ILU algorithm, which is one of our prime objectives.

4. Structural characterizations. In this section we present a collection of definitions, observations, lemmas, and theorems that provide characterizations of incomplete S-level and M-level fill. We also introduce the concept of 1-alternating fill paths, which are particular configurations of fill paths for which the S-level and M-level characterizations coincide.

Figure 4.1 provides a pictorial summary of the interconnections of this section's results.

4.1. Static characterization of S-level fill. This section's first result tells us that nontrivial fill paths can always be decomposed into shorter fill paths.

Lemma 3. *Any fill path $P(i, j)$ that contains two or more edges can be uniquely decomposed into two fill paths, $P(i, h)$ and $P(h, j)$, each of which contains at least a single edge.*

Proof. Given a fill path $P(i, j)$ containing two or more edges, let h denote the highest numbered interior vertex on the path. The $P(i, h)$ section of this path is a fill path by the choice of h, since all intermediate vertices on this section are numbered lower that h. Similarly, the $P(h, j)$ section of this path is also a fill path. Thus, the fill path can clearly be decomposed in two subpaths, both of which are fill paths (existence).

To show uniqueness, suppose there exists some other decomposition. Let g be an interior vertex on the $P(i, j)$ path, distinct from h, such that both $P(i, g)$ and $P(g, j)$ sections are fill paths. Then h, which is also on the $P(i, j)$ path, must either be situated between vertices i and g, or between vertices g and j. Without loss of generality, assume vertex h is situated between vertices i and g. Then by Definition 1,

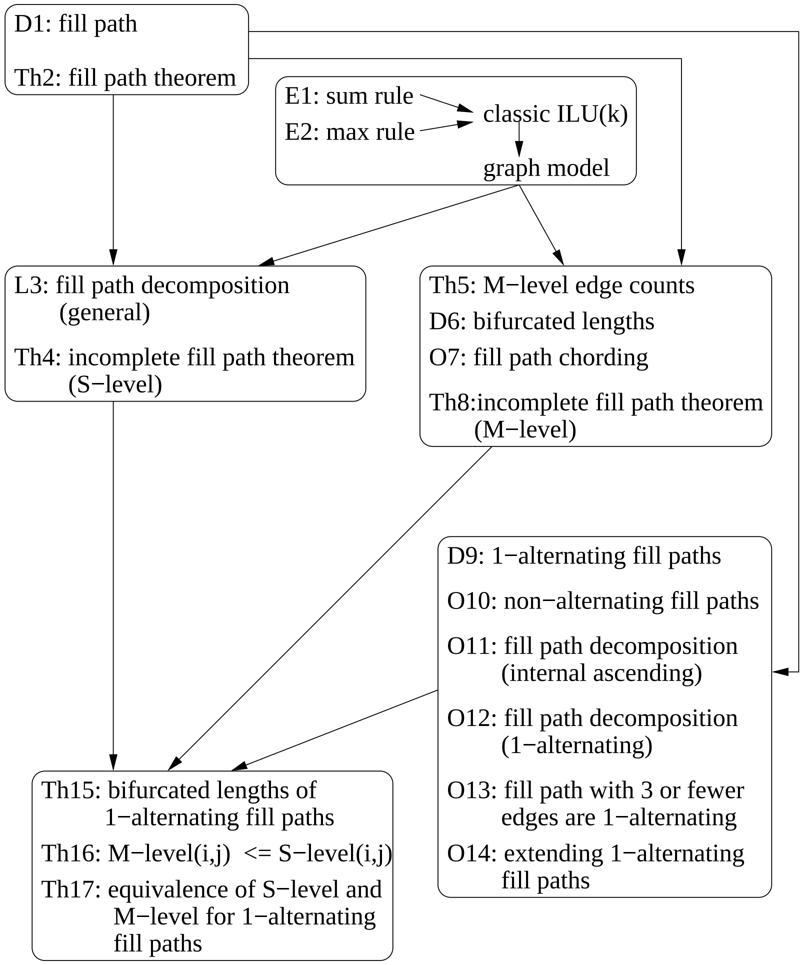


Fig. 4.1. *Relationships of definitions, theorems, and observations.*

$P(i, g)$ is not a fill path, since the path contains an interior vertex that is numbered higher than one of the end vertices. □

We next present the *S-level incomplete fill path theorem.* This theorem provides a static characterization of fill for classical ILU(ℓ) factors that are computed using the sum rule for level assignment [1]. The theorem is analogous to the original fill path theorem in that it provides a means of determining fill locations by examining paths in the graph of a matrix.

Theorem 4. *Let $G(A) = (V, E)$ be the graph of a square matrix A, and let $\langle i, j\rangle$ be a permitted edge in G^S_*. Then S-level$(i, j) = k$ if and only if there exists a shortest fill path of length $k + 1$ that joins i and j in $G(A)$.*

Proof. If there is a shortest fill path of length $k + 1$ joining i and j in $G(A)$, we prove the result, that an edge $\langle i, j\rangle$ with S-level$(i, j) = k$ exists in G^S_*, by induction on u, which is the length of the fill path.

The base case $u = 1$ is immediate, since, by the construction in § 3, a fill path of

[1] A version of this proof was originally presented in [9].

length one in the graph $G(A)$ is an edge $\langle i,j \rangle$ in $G(A)$, and edges in $G(A)$ are assigned level zero, and are also edges in G_*^S.

Now assume that the result is true for all lengths u less than $k+1$; we show it is also true for shortest paths of length $u = k+1$. Let $P(i,j)$ be a shortest fill path joining vertices i and j in $G(A)$, and let this path have length $u = k+1$.

Let h denote the highest numbered interior vertex on the fill path $P(i,j)$. We claim that the $P(i,h)$ section of this path is a shortest fill path in $G(A)$ joining i and h. This section is a fill path by the choice of h and Lemma 3. If there were a fill path joining i and h that was shorter than the $P(i,h)$ section, we would be able to concatenate it with the $P(h,j)$ section to form a shorter $P(i,j)$ fill path. Hence the $P(i,h)$ section is a shortest fill path joining i and h. Similarly, the $P(h,j)$ section of this path is the shortest fill path joining h and j.

Since each of these sections has fewer than $k+1$ edges, and is a shortest fill path, the inductive hypothesis applies. Denote the number of edges in the $P(i,h)$ ($P(h,j)$) section of this path by v (w), where $v + w = u = k+1$. By the inductive hypothesis the edge $\langle i,h \rangle$ is a fill edge of level $v - 1 = k_1$, and the edge $\langle h,j \rangle$ is a fill edge of level $w - 1 = k_2$. Now by the sum rule for updating fill levels, when the vertex h is eliminated, we have a fill edge $\langle i,j \rangle$ of level

$$k_1 + k_2 + 1 = (v-1) + (w-1) + 1 = v + w - 1 = u - 1 = (k+1) - 1 = k.$$

Now we prove the converse. Suppose that $\langle i,j \rangle$ is a fill edge of level k in G_*^S; we show the result that there exists a shortest fill path in $G(A)$ containing $u = k+1$ edges by induction on the level k.

The base case $k = 0$ is immediate since, by the construction in § 3, the edge $\langle i,j \rangle$ constitutes a trivial fill path of length one.

Assume that the result is true for all fill levels less than k. Let the fill edge $\langle i,j \rangle$ with S-level$(i,j) = k$ be created in G_*^S, when vertex h is eliminated, by the previously existing edges $\langle i,h \rangle$ and $\langle h,j \rangle$. Let the edge $\langle i,h \rangle$ have level k_1 and the edge $\langle h,j \rangle$ have level k_2. By the sum rule for computing levels, we have that $k_1 + k_2 + 1 = k$. By the inductive hypothesis, there is a shortest fill path of length $v = k_1 + 1$ joining i and h, and such a path of length $w = k_2 + 1$ joining h and j. Concatenating these paths, we find a fill path joining i and j of length

$$v + w = (k_1 + 1) + (k_2 + 1) = k_1 + k_2 + 2 = k + 1.$$

We need to prove that the $P(i,j)$ fill path in the previous paragraph is a shortest fill path between i and j. Consider any other pair of edges $\langle i,g \rangle$ and $\langle g,j \rangle$ in G_i^S that causes the fill edge $\langle i,j \rangle$ when vertex i is eliminated. By the choice of the vertex h, if the level of the edge $\langle i,g \rangle$ is k_1', and that of $\langle g,j \rangle$ is k_2', then $k_1' + k_2' \geq k$.

The inductive hypothesis applies to the $P(i,g)$ and $P(g,j)$ sections, and hence the sum of their lengths is at least $k+1$. □

D'Azevedo, Forsyth, and Tang [4] were aware of the connection between matrix entry levels and fill path lengths in graphs. However, they framed the connection as a definition for *all* fill levels. We quote from their work.

> We define the fill level for the node pair (v_i, v_j) in G_k to be the length of the shortest path from v_i to v_j minus one, i.e. $level_{ij}^{(k)} = m$ We define initially
>
> $$level_{ij}^{(0)} = \begin{cases} 0 & \text{if } a_{ij} \neq 0 \\ \infty & \text{otherwise.} \end{cases}$$

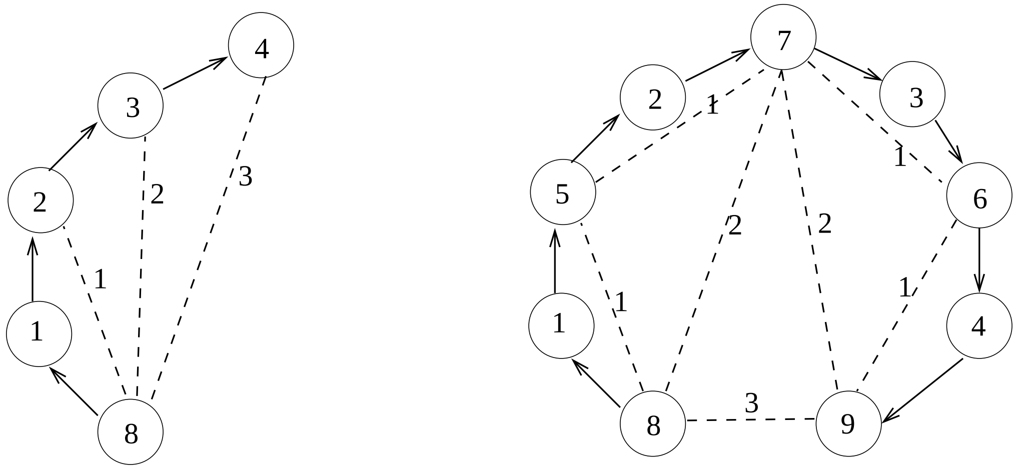


FIG. 4.2. *M-level fill level and path length relationships. Edges in $G(A)$ are drawn with solid lines. Edges in G_*^M are drawn with dashed lines, and labeled with their levels. The vertex numbering indicates elimination ordering. Both graphs contain an M-level $k = 3$ fill edge. The corresponding fill path in the graph on the left, 8,1,2,3,4, contains $3 + 1 = k + 1 = 4$ edges. The corresponding fill path in the graph on the right, 8,1,5,2,7,3,6,4,9, contains $2^3 = 2^k = 8$ edges.*

In contrast, we start with a weaker premise—the definition of level zero fill—then prove an *if and only if* connection.

D'Azevedo, Forsyth, and Tang's work centers around a novel algorithm that combines an ordering technique with ILU factorization. They consider $G(A)$ to be initially unordered, and one vertex is ordered during each elimination step. They define fill levels in terms of path lengths through vertices in reachable sets, with a reachable set consisting of vertices already eliminated and ordered. They did not postulate or prove this connection as a theorem, as we have done.

4.2. Static characterization of M-level fill. We now turn our attention towards M-level fill entries and their associated fill paths. While a fill edge with S-level$(i, j) = k$ corresponds to a fill path with exactly $k + 1$ edges, this section's first result says that a fill edge with M-level$(i, j) = k$ corresponds to a fill path that may contain anywhere between $k+1$ and 2^k edges. Figure 4.2 illustrates the intuition underlying this claim. Two very simple graphs are shown, both of which contain fill paths that correspond to level $k = 3$ fill edges. The fill path in the Figure 4.2(a) contains $3+1 = k+1 = 4$ edges, while the fill path in Figure 4.2(b) contains $2^3 = 2^k = 8$ edges.

THEOREM 5. *Let $G(A) = (V, E)$ be the graph of a square matrix A, let $\langle i, j\rangle$ be a permitted edge in G_*^M with M-level$(i, j) = k$, and let $P(i, j)$ be a shortest fill path in $G(A)$. Let u represent the number of edges in the path $P(i, j)$. Then $k+1 \le u \le 2^k$.*

Proof. We argue by induction on the fill edge's level, k. The base case $k = 0$ is immediate since, by the construction specified in § 3, a fill edge of level zero corresponds to a fill path that contains $u = 1$ edges. In this case $k + 1 = 1 \le u \le 2^k = 1$, so the result is true.

Now assume the result is true for all edges whose M-level is less than k; we show it is also true for edges with level k. Let h be the vertex whose elimination creates the fill edge $\langle i, j\rangle$ of M-level k. Let the edge $\langle i, h\rangle$ have M-level k_1 and the edge $\langle h, j\rangle$ have M-level k_2. By the max rule for computing levels, we have that $\max\{k_1, k_2\} + 1 = k$, hence both k_1 and k_2 are less than k, so the inductive hypothesis applies. Also, either $k_1 = k-1$ or $k_2 = k-1$ or both. Without loss of generality, assume $k_1 = k-1$.

Let v represent the number of edges in the fill path joining i and h in $G(A)$, and w the number of edges in the fill path joining h and j in $G(A)$. By the inductive hypothesis, $k_1 + 1 \le v \le 2^{k_1}$, and $k_2 + 1 \le w \le 2^{k_2}$. When h is eliminated these paths are concatenated, resulting in the fill path $P(i,j)$ whose length u is bounded:

$$(k_1 + 1) + (k_2 + 1) \le u \le 2^{k_1} + 2^{k_2}.$$

To make the left-hand side as small as possible, assume $k_1 = k-1$ and $k_2 = 0$, which is possible if $P(h,j)$ contains a single edge. In this case

$$u = (k_1 + 1) + (k_2 + 1) = ((k-1) + 1) + (0 + 1) = k + 1.$$

To make the right-hand side as large as possible, let $k_1 = k-1$ and $k_2 = k-1$. In this case

$$u = 2^{k_1} + 2^{k_2} = 2^{(k-1)} + 2^{(k-1)} = 2^k.$$

Therefore, $k + 1 \le u \le 2^k$. □

Not only is there wide latitude in fill path lengths associated with M-level fill edges, but it is also the case that a fill path $P(i,j)$ in $G(A)$ that is associated with a fill edge $\langle i,j \rangle$ in G_*^M may not be the shortest fill path (that is, the fill path containing the fewest number of edges) that connects vertices i and j in $G(A)$. Figure 4.3 illustrates this point. The figure shows a fill edge with M-level(i,j)=3 that arises due to the existence of a fill path that contains eight edges. Vertices i and j are also connected by a fill path that only contains five edges; however, this fill path would cause $\langle i,j \rangle$ to have M-level$(i,j) = 4$.

Hence, when fill is computed using the max rule, it appears that there is no necessary connection between fill levels and path lengths (where "length" indicates, as we use the term, the number of edges in a path). These observations suggest the need for a definition of *path length* that does not strictly depend on the number of edges in the path. Accordingly, we introduce the concept of *bifurcated length*, which is recursive in nature. In the following definition the phrase "unique fill subpaths" refers to the unique decomposition stated in Lemma 3.

DEFINITION 6. *A fill path containing a single edge has bifurcated length zero. A fill path containing two or more edges, whose unique fill subpaths have bifurcated lengths v and w, has bifurcated length $u = \max\{v, w\} + 1$.*

Heretofore, we have used the phrase "shortest fill path" to indicate, of all possible fill paths connecting two vertices in a graph, a (possibly nonunique) path containing the fewest number of edges. When discussing bifurcated lengths we use an analogous phrase, "fill path with shortest bifurcated length." This term indicates, of all possible fill paths connecting two vertices in a graph, a (possibly nonunique) path *whose bifurcated length is the smallest possible.*

A *chord* of a path is an edge that joins two non-consecutive vertices on the path. If an edge is added to a graph such that the shortest fill path $P(i,j)$ is chorded, the result will be that vertices i and j are joined by a shorter fill path than previously, and hence the corresponding S-level(i,j) will be reduced. This concept does not transfer to the study of bifurcated path lengths.

OBSERVATION 7. *A fill path may be chorded, and its bifurcated length unchanged.*

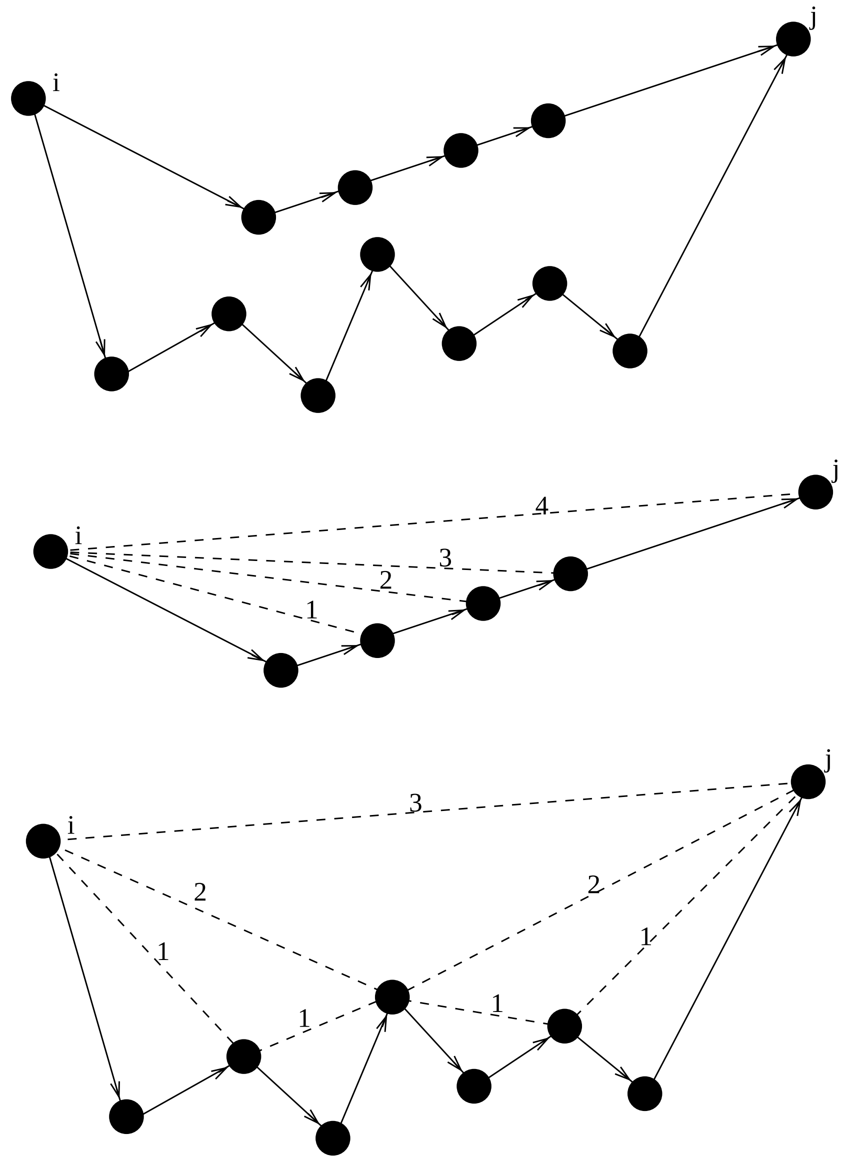
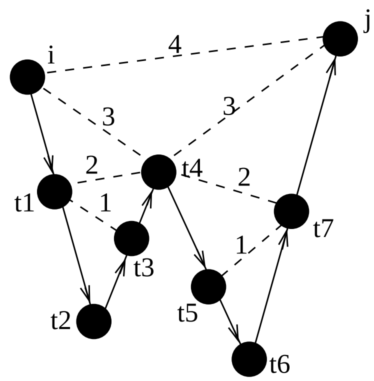


Fig. 4.3. *M-level path lengths and edge counts. Top: a graph in which vertices i and j are connected by two fill paths. In the middle and bottom, the paths are shown separately, with fill edges indicated by dashed lines. The path in the middle contains fewer edges but results in a fill edge that has a higher level than does the path at the bottom. Vertex ordering is indicated by vertical placement: vertices that are lower on the page are assumed to be ordered before vertices placed higher on the page.*

Figure 4.4 shows a fill path that contains 8 edges and has bifurcated length 4. After chording, the resulting shorter fill path contains only 7 edges, however, its bifurcated length is unchanged.

The next theorem provides a static characterization of M-level fill.

Theorem 8. *Let $G(A) = (V, E)$ be the graph of a square matrix A, and let $\langle i, j\rangle$ be a permitted edge in G_*^M. Then M-level$(i, j) = k$ if and only if there exists a fill*



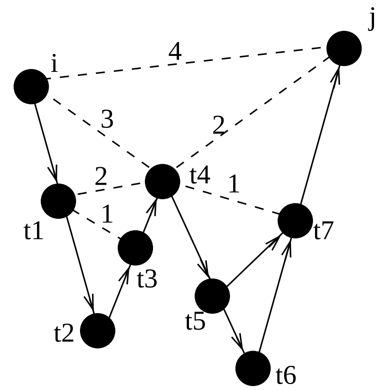


FIG. 4.4. *A fill path may be chorded, and its bifurcated length unchanged. Left: fill path $P(i,j) = i, t_1, t_2, t_3, t_4, t_5, t_6, t_7, j$ in $G(A)$ contains 8 edges and has bifurcated length 4. Right: the sub path t_5, t_6, t_7 has been chorded in $G(A)$; i and j are now connected by the shorter fill path $P(i,j) = i, t_1, t_2, t_3, t_4, t_5, t_7, j$. This path contains 7 edges, but the bifurcated length of $P(i,j)$ remains 4. Edges in $G(A)$ are drawn with solid lines. Edges in G_*^M are drawn with dashed lines, and labeled with their levels. Vertex ordering is indicated by vertical placement: vertices that are lower on the page are assumed to be ordered before vertices placed higher on the page.*

path joining vertices i and j in $G(A)$ with bifurcated length k, and this path has the shortest bifurcated length amongst all fill paths between i and j.

Remark. In contrast to Theorem 4, here there is no "+1" difference between bifurcated path lengths and M-levels. This is because the "+1" is incorporated into the definition of bifurcated path lengths.

Proof. If there is a fill path with shortest bifurcated length k joining i and j in $G(A)$, we prove the result, that an edge $\langle i, j\rangle$ with M-level$(i,j) = k$ exists in G_*^M, by induction on u, which is the bifurcated length of the fill path.

The base case $u = 0$ is immediate, since, by the construction in § 3 and Definition 6, a path with bifurcated length zero corresponds to an original edge in $G(A)$.

Now assume the result is true for all fill paths with bifurcated length u less than k. We will prove that the result is true when the bifurcated length of a fill path is $u = k$.

Let $P(i,j)$ be a fill path with shortest bifurcated length that joins i to j in $G(A)$, and let the bifurcated length of this path be $u = k$. Let h be the highest-numbered interior vertex in this fill path. Then $P(i,h)$ and $P(h,j)$ are also fill paths by Lemma 3.

Let the bifurcated length of the fill path $P(i,h)$ be v and let the bifurcated length of the fill path $P(h,j)$ be w. By Definition 6, the bifurcated path length of $P(i,j)$ is $\max\{v,w\}+1$, so the bifurcated lengths of v and w are both less than k. Note that either v or w (or both) is equal to $k-1$. Without loss of generality, assume that w is less than $k-1$. Then it must be that $v = k-1$, and therefore the fill path $P(i,h)$ has the shortest bifurcated length possible.

Now suppose there is a path $P'(h,j)$ whose bifurcated length is less than w. Then we can freely replace the path $P(h,j)$ with the path $P'(h,j)$, and the bifurcated length of the path $P(i,j)$ will be unchanged.

Thus $P(i,j)$ is decomposable into two subpaths, $P(i,h)$ and $P(h,j)$, both of which are fill paths and have shortest bifurcated lengths less than k. Hence, the inductive hypothesis applies, so there exists a fill edge $\langle i,h\rangle$ with M-level v, and a fill edge $\langle h,j\rangle$ with M-level w. By the max level rule, when vertex h is eliminated, the fill edge $\langle i,j\rangle$

is created with M-level$(i,j) = \max\{v, w\} + 1 = k$.

Now we prove the converse. Suppose that $\langle i,j\rangle$ is a fill edge with M-level$(i,j) = k$ in G_*^M; we show the result that there exists a fill path $P(i,j)$ in $G(A)$ with shortest bifurcated length $u = k$ by induction on k, the edge's level.

The base case $k = 0$ is immediate since, by the construction in § 3, a fill edge with level zero corresponds to a fill path that contains a single edge, and by Definition 6 this path has bifurcated length zero.

Now assume the result is true for all fill edges with M-level less than k; we show it is also true for fill edges with M-level equal to k.

Assume the fill edge $\langle i,j\rangle$ with M-level$(i,j) = k$ is created, when vertex h is eliminated from G_h^M, by the previously existing edges $\langle i,h\rangle$ and $\langle h,j\rangle$.

Let the edge $\langle i,h\rangle$ have M-level$(i,h) = k_1$ and the edge $\langle h,j\rangle$ have M-level$(h,j) = k_2$. By the max rule for computing levels, we have that $\max\{k_1, k_2\} + 1 = k$. Then both fill edges $\langle i,h\rangle$ and $\langle h,j\rangle$ have levels less than k, so the inductive hypothesis applies. Thus there exists a fill path that connects vertices i and h and has shortest bifurcated length $v = k_1$, and a fill path that connects vertices h and j and has shortest bifurcated length $w = k_2$. Additionally, either $k_1 = k - 1$ or $k_2 = k - 1$ or both. Without loss of generality, assume $k_1 = k - 1$.

Now from Definition 6, the bifurcated length of the fill path $P(i,j)$ is

$$u = \max\{v, w\} + 1 = \max\{k_1, w\} + 1 = \max\{k - 1, w\} + 1 = k.$$

We also need to prove that the $P(i,j)$ fill path has the shortest bifurcated length amongst all fill paths connecting vertices i and j in $G(A)$. Suppose there were a path $P'(i,j)$ in $G(A)$ that had a shorter bifurcated length, that is, a bifurcated length u' less than k. From the first part of this proof, the edge $\langle i,j\rangle$ in G_*^M would then have an M-level less than k, which contradicts the premise that the fill edge $\langle i,j\rangle$ has M-level$(i,j) = k$. □

4.3. Similarity of S-level and M-level fill for 1-alternating fill paths. A graph sometimes has the property that an ILU(ℓ) factorization computed using the sum rule is identical to that which results when the max level is used. This property is an attribute, e.g., of graphs whose associated matrices arise from the discretization of partial differential equations on naturally ordered, structured grids, when the factorization level is $\ell \le 3$. For these graphs, the shortest fill path connecting any two vertices i and j, and the fill path with shortest bifurcated length connecting the same two vertices i and j, are always identical when level$(i,j) \le 3$. A consequence (which is the main result of this section) is that M-level(i,j) = S-level(i,j) for such cases. To capture and generalize the particular feature responsible for this consonance of level assignment, we define *1-alternating fill paths.*

As a preliminary, an *ascending path* is a path $(t_1, \ldots, t_k)$ that contains at least two vertices, with $t_h < t_{h+1}$ for $1 \le h < k$. Similarly, a *descending path* is a path $(t_1, \ldots, t_k)$ that contains at least two vertices, with $t_h > t_{h+1}$ for $1 \le h < k$.

Definition 9. *A fill path $P(i,j)$ is* 1-alternating *if it has one of the following forms.*
(i) A single edge, $\langle i,j\rangle$.
(ii) An edge $\langle i,h\rangle$ with $i > h$, concatenated with an ascending path $P(h,j)$.
(iii) A descending path $P(i,h)$ concatenated with an edge $\langle h,j\rangle$ with $h < j$.
(iv) A descending path $P(i,h)$ concatenated with an ascending path $P(h,j)$.

Note that forms (ii) and (iii) are restricted forms of form (iv). We call a 1-alternating path *internal-ascending* if it is either of form (*ii*), or consists of a single

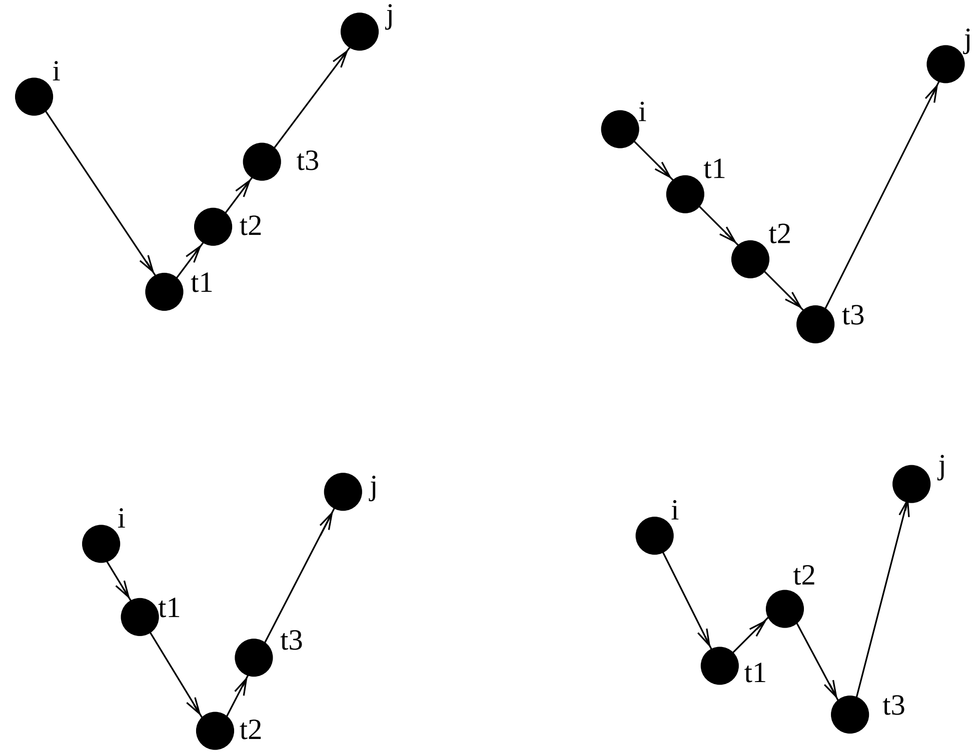


FIG. 4.5. *1-alternating and non-alternating fill paths. Top left: internal-ascending fill path. Top right: internal-descending fill path. Bottom left: 1-alternating fill path. Bottom right: non-alternating fill path. Here as elsewhere, vertical positioning of vertices is indicative of their relative orderings.*

edge $\langle i, j \rangle$ with $i < j$. We call a 1-alternating path *internal-descending* if it is either of form (iii), or consists of a single edge $\langle i, j \rangle$ with $i > j$. Figure 4.5 illustrates the different species of 1-alternating fill paths, and the difference between 1-alternating and non-alternating fill paths.

By way of building up to this section's main result, and as an aid to intuition, several observations concerning properties of 1-alternating fill paths follow.

OBSERVATION 10. *A fill path is non-alternating if the path contains a sequence of interior vertices, $t_f, \ldots, t_g, \ldots, t_h$, such that $t_f < t_g$ and $t_g > t_h$.*

In the non-alternating path at the bottom right of Figure 4.5, $t_1 < t_2$ and $t_2 > t_3$.

OBSERVATION 11. *If $P(i, j) = i, t_1, t_2, \ldots, j$ is an internal-ascending fill path that contains at least three edges, and h is any interior vertex on the path with $h > t_1$, then $P(i, h)$ is also an internal-ascending fill path.*

The truth of this observation follows immediately from Definitions 1 and 9. Referring to the $P(i, j)$ fill path illustrated in the top left portion of Figure 4.5, this observation says that the paths $P(i, t_2)$ and $P(i, t_3)$ are internal-ascending fill paths. Note, however, that neither $P(t_1, j)$ nor $P(t_2, j)$ is a fill path. A similar observation holds for internal-descending paths.

The next observation is based on the fact that the highest numbered interior vertex of a 1-alternating path is necessarily adjacent to one of the end points of the path.

Observation 12. *If $P(i,j)$ is a 1-alternating fill path containing $k+1$ edges, with $k \geq 1$, then the path can be uniquely decomposed into two 1-alternating fill paths $P(i,h)$ and $P(h,j)$. One of these fill paths will contain k edges, and the other a single edge.*

The existence and uniqueness of the decomposition was shown in Lemma 3. In that lemma's proof, we saw that the vertex h is necessarily the largest interior vertex on the $P(i,j)$ path. From Definition 9, this vertex is adjacent to either vertex i or vertex j, hence either $P(h,j)$ is a path containing a single edge, in which case the path $P(i,h)$ must contain k edges, or $P(i,h)$ is a path containing a single edge, in which case the path $P(h,j)$ must contain k edges. Referring again to Figure 4.5, the $P(i,j)$ fill path in the top left contains four edges, and can be decomposed into the fill paths (i,t_3) and (t_3,j), containing three edges and a single edge, respectively.

Observation 13. *Any fill path with three or fewer edges is 1-alternating.*

The truth for the one and two edge cases follows directly from Definition 9. Now consider a fill path with three edges, i,t_1,t_2,j. Either $t_1 < t_2$, or $t_2 < t_1$; in either case the fill path is 1-alternating by Definition 9. As illustrated in the bottom right of Figure 4.5, paths with four or more edges are not necessarily 1-alternating.

Observation 14. *If $P(i,j)$ is any species of 1-alternating fill path, then*
(i) if $\langle j,t\rangle$ is an edge with $j < \min\{i,t\}$, then $P(i,t)$ is also a 1-alternating fill path;
(ii) if $\langle t,i\rangle$ is an edge with $i < \min\{j,t\}$, then $P(t,j)$ is also a 1-alternating fill path.

In the top left of Figure 4.5, $P(i,t_3)$ is a 1-alternating fill path, and $\langle t_3,j\rangle$ is an edge. By this observation, $P(i,j)$ is therefore a 1-alternating fill path. This observation states a condition that permits a fill path to be extended while preserving its 1-alternating character. As such it is the complement of Observation 12, which says that any 1-alternating fill path can be decomposed.

Note that extending a 1-alternating path does not necessarily preserve any internal-descending or internal-ascending property it may possess. For example, if an internal-descending fill path $P(i,j)$ is extended by concatenation with an edge $\langle j,t\rangle$ with $t > j$, then the resulting fill path $P(i,t)$ is no longer internal-descending.

The following theorem establishes a relationship between path lengths and bifurcated lengths of 1-alternating fill paths.

Theorem 15. *Let $P(i,j)$ be a fill path that contains $k+1$ edges. The bifurcated length of $P(i,j)$ is k if and only if the fill path is 1-alternating.*

Proof. Suppose there exists a 1-alternating fill path that connects vertices i and j and contains $k+1$ edges. We prove the path has bifurcated length k by induction on k, the number of edges in the path.

The base case $k = 0$ is immediate since a fill path containing a single edge has bifurcated length zero by Definition 6. Now assume the result is true for all 1-alternating fill paths containing k or fewer edges; we show it is also true for 1-alternating fill paths containing $k+1$ edges.

Let h denote the highest numbered interior vertex on the path joining i and j. From Observation 12, h must be adjacent to either vertex i or vertex j. Without loss of generality, assume it is adjacent to vertex j.

Thus, $P(i,h)$ is a 1-alternating fill path containing k edges, and $P(h,j)$ is a 1-alternating fill path containing a single edge, so the inductive hypothesis applies to both subpaths.

By the inductive hypothesis, $P(i,h)$ has bifurcated length $k-1$, and $P(h,j)$ has bifurcated length zero. When these two paths are concatenated, the resulting path $P(i,j)$ has bifurcated length, by Definition 6, of

$$\max\{k-1, 0\} + 1 = k.$$

Now we prove the converse. Suppose vertices i and j are connected by a fill path that contains $k+1$ edges and has bifurcated length k; we show that the path is 1-alternating by induction on k, the number of edges in the path.

The base case $k = 1$ is immediate since a fill path containing a single edge is 1-alternating (Definition 6). Now assume the result is true for any fill path that contains m edges and has bifurcated length $m-1$, where $m \leq k$. We show the result is also true for paths that contain $k+1$ edges.

Let h denote the highest numbered interior vertex on the fill path joining i and j. Let m_1 be the number of edges in the $P(i,h)$ subpath, and m_2 the number of edges in the $P(h,j)$ subpath. Then $m_1 + m_2 = k+1$.

Let k_1 be the bifurcated length of the $P(i,h)$ subpath, and k_2 be the bifurcated length of the $P(h,j)$ subpath. Then $\max\{k_1, k_2\} + 1 = k$. Hence, either $k_1 = k-1$ or $k_2 = k-1$ or both. Without loss of generality, suppose $k_1 = k-1$. Then from Theorem 5, the $P(i,h)$ subpath must contain at least k edges, that is, $m_1 \geq k$. And since $m_1 + m_2 = k+1$, it must contain exactly k edges, and m_2, the number of edges in the $P(h,j)$ subpath, must be 1.

Since $P(i,h)$ has bifurcated length $k-1$ and contains k edges, the inductive hypothesis applies, i.e, $P(i,h)$ is a 1-alternating fill path. Similarly, since $P(h,j)$ contains a single edge, and by definition 6 has bifurcated length zero, the inductive hypothesis applies.

Finally, by Observation 14, when the $P(i,h)$ path is concatenated with the $P(h,j)$ path, the resulting $P(i,j)$ fill path is 1-alternating. ▯

The next observation tells us that the *M-level* associated with a fill path is never greater than the *S-level* associated with that same path.

OBSERVATION 16. *Let $G(A) = (V,E)$ be the graph of a square matrix A, and let M-level(i,j) $= k$. Then S-level(i,j) $\geq$M-level(i,j).*

From the left hand side inequality of Theorem 5, any fill path $P(i,j)$ must contain at least $k+1$ edges. The truth of the preceding observation then follows directly from Theorem 4.

This section's final theorem formalizes the relationship between M-level and S-level fill that was alluded to in this section's introduction.

THEOREM 17. *Let $G(A) = (V,E)$ be the graph of a square matrix A, and let $\langle i,j \rangle$ be a permitted edge in G_*^M with M-level(i,j) $= k$. Then M-level(i,j) $=$ S-level(i,j) if and only if there exists a fill path with shortest bifurcated length connecting i and j that is also 1-alternating.*

Proof. Suppose M-level(i,j) $=$ S-level(i,j); we show there must exit a fill path with shortest bifurcated length connecting i and j that is also 1-alternating.

By the supposition that S-level(i,j) $= k$ and Theorem 4, there exists a fill path $P(i,j)$ connecting i and j that has $k+1$ edges. By the same theorem, no fill path connecting i and j can have fewer than $k+1$ edges.

By the supposition that M-level(i,j) $= k$ and Theorem 8, no fill path connecting i and j can have bifurcated length shorter than k. In particular, the bifurcated length of $P(i,j)$ can not be shorter than k.

By Observation 16, the M-level value derived from any fill path can not be greater than the S-level derived from that path. Hence, the bifurcated length of $P(i,j)$ can not be longer than k.

Since the bifurcated length of $P(i,j)$ can be neither longer nor shorter than k, it must be equal to k, and since no fill path connecting these two nodes can have shorter bifurcated length, $P(i,j)$ must be a fill path with shortest bifurcated length. Since the path contains $k+1$ edges, by Theorem 15 it is 1-alternating.

Now we prove the converse. Suppose there exists a fill $P(i,j)$ with shortest bifurcated length k that is also 1-alternating; we show that M-level$(i,j) =$ S-level(i,j).

By Theorem 8, M-level$(i,j) = k$, so we must show that S-level$(i,j) = k$; by Theorem 4, it suffices to show that there exists a shortest fill path connecting nodes i and j that contains $k+1$ edges.

Since M-level$(i,j) = k$, by Theorem 5 every fill path connecting nodes i and j contains at least $k+1$ edges. It therefore suffices to show that $P(i,j)$ contains $k+1$ edges. We can now restate our goal as follows.

Suppose there exists a fill path $P(i,j)$ with bifurcated length k that is also 1-alternating; we show that this path contains $k+1$ edges by induction on u, the bifurcated length of the fill path.

The base case, $u = 0$, is immediate, from Definitions 6 and 9. Now assume the result is true for all values of $u < k$; we show the result is also true for $u = k$.

By Observation 12, $P(i,j)$ can be decomposed into two fill paths, $P(i,h)$ and $P(h,j)$, both of which are 1-alternating. One of these paths is exactly one edge long, so the inductive hypothesis applies. Without loss of generality, assume this edge is $P(i,h)$.

By Definition 6, the bifurcated length of $P(h,j)$ must be $u = k-1$. If it were not, then when fill paths $P(i,h)$ and $P(h,j)$ were joined, the resulting fill path would not have a bifurcated length of k, as was supposed. Since $P(h,j)$ is 1-alternating and has bifurcated length $u < k$ the hypothesis applies, so the path contains $u+1 = k$ edges. Then when $P(i,h)$ and $P(h,j)$ are concatenated, the result is $P(i,j)$, which contains $k+1$ edges.
□

5. Applications.

This section provides a brief overview of some of the practical applications of the S-level incomplete fill path theorem. Additional applications and detailed explanations can be found in [8].

5.1. Computing upper triangular structures.

Figure 5.1 shows a procedure, GS-Urow, that can be used to compute ILU(ℓ) structures that are identical to those computed by Classic-ILU. The procedure is invoked separately for each row in the matrix. Unlike Classic-ILU, which requires the results of previously computed rows 1 through $i-1$ to factor a row i, the new GS-Urow procedure requires only the graph $G(A)$, and hence has the novel feature that the structure of each row in the factor can be computed independently, in parallel.

The procedure operates via a simple breadth first search [3] that finds a shortest path between a seed vertex i and vertices reachable from i via traversal of $\ell+1$ or fewer edges. Hence, the correctness of the algorithm follows directly from Theorem 4. Other algorithms can be devised that compute the structure of rows in the lower triangular factor; see [8] for details.

For structured 3D graphs, it can be shown that GS-Urow has a lower runtime complexity ($O(n\ell^3/p)$) than does Classic-ILU ($O(n\ell^4)$). (Here, p is the number of processors, n is the number of rows in the matrix, and ℓ is the factorization level.) For structured 2D graphs, runtimes are ($O(n\ell^2/p)$) for GS-Urow and ($O(n\ell^2)$) for Classic-ILU [8].

```
GS-Urow(G(A), ℓ, i, adj′(i))
1   # Initialization for BFS from vertex i
2   Q ← {i}
3   length[i] ← 0
4   visited[i] ← i
5   # BFS phase
6   while Q ≠ ∅
7       h ← Dequeue(Q)
8       for t ∈ adj(h) with visited[t] ≠ i
9           visited[t] ← i
10          if t < i and length[h] < ℓ
11              Enqueue(Q, t)
12              length[t] = length[h] + 1
13          if t > i
14              insert t in adj′(i)
```

Fig. 5.1. GS-Urow. *This procedure computes the structure of row i in the the factor's upper triangle. Inputs are $G(A)$; ℓ, the factorization level; and i, the row whose structure is to be computed. The row's structure is returned in $adj'(i)$).*

5.2. ILU(ℓ) memory allocation. As noted in §2.3, it is possible to compute storage requirements for factors of symmetric matrices in time proportional to the number of nonzeros in F, and in space proportional to the row count. This enables one to efficiently allocate storage and set up data structures prior to the commencement of numeric factorization.

In general, there is no equivalent procedure for predicting ILU storage requirements. One practice is to guess at the number of nonzeros in the factor and initially allocate that much storage; if this proves insufficient, the factorization fails. In some ILU schemes, such as ILUT, an arbitrary limit is set on the number of nonzeros in each row. This ensures that adequate storage will be allocated and, unless a zero-pivot is encountered, the factorization will succeed. Another approach, common in implementations coded in C or C++, is to dynamically reallocate storage when the initial guess is insufficient. However, this reallocation strategy can incur non-trivial overhead and can also fragment memory.

The GS-Urow procedure presented in the previous section (Figure 5.1) can be modified to compute storage requirements for ILU(ℓ) upper triangular factors using $O(n)$ space. The modification is accomplished as follows. Initialize a counter to zero. Change Step 14, which previously inserted an element in an adjacency list, to increment the counter. Return the counter's value.

While these modifications permit computation of a factor's storage requirements in $O(n)$ space, the time complexity is identical to that required for actually performing symbolic factorization. It is an open question whether faster methods for computing exact ILU(ℓ) storage requirements can be devised.

5.3. Bounding ILU(ℓ) fill amounts. The S-level incomplete fill path theorem can be used as an analytic tool to bound the amount of fill in ILU(ℓ) factors. The core idea is to devise an expression that bounds the number of vertices that are reachable from a seed vertex via paths that contain $\ell + 1$ or fewer edges.

Suppose we are given a graph of bounded degree that corresponds to some coef-

Table 5.1

Bounds for fill in ILU(ℓ) factors. "2D" and "3D" graphs result from the discretization of partial differential equations on structured grids using 5-point and 7-point stencils; n is the number of rows in the matrix.

graph classification	**fill bound**
bounded degree c	$O(nc^{\ell+1})$
2D, any ordering	$O(n\ell^2)$
2D, natural ordering	$O(n\ell)$
3D, any ordering	$O(n\ell^3)$
3D, natural ordering	$O(n\ell^2)$

ficient matrix. We argue as follows. From any vertex in the graph, at most c vertices are reachable via paths of length one; these paths correspond to fill of level $\ell = 0$. From each of those vertices, we can potentially discover not more than an additional c vertices. Thus, there are at most c^2 fill paths of length two, corresponding to fill of level $\ell = 1$. Continuing in this vein, it is easily seen that there are at most c^3 fill paths of length three, corresponding to fill of level $\ell = 2$; c^4 fill paths of length four, corresponding to fill of level $\ell = 3$; and so on. The total fill for an arbitrary level ℓ is thus bounded by

$$\sum_{i=1}^{\ell+1} c^i = O(c^{\ell+1})$$

Using the S-level incomplete fill path theorem as our springboard, we have devised similar expressions for several types of graphs. These are summarized in Table 5.1. Again, see [8] for detail.

6. Concluding remarks. This paper has generalized the fill path theorem, which is of long standing importance in direct factorization methods, by adding the concept of path length. Specifically, we have shown how path lengths in graphs correlate to a matrix entry's fill level for both of the rules (sum and max) that are used for computing an entry's level. Our primary result was the framing of two incomplete fill path theorems that describe the structure of ILU(ℓ) factors.

We consider our S-level incomplete fill path theorem of principal practical importance, since the sum rule is almost always employed in ILU(ℓ) library codes. We showed that this theorem leads directly to a novel, embarrassingly parallel algorithm that computes ILU(ℓ) symbolic structures that are identical to those computed by previously existing row-based algorithms.

We showed how the new algorithm can be modified to compute ILU(ℓ) storage requirements for arbitrary fill levels ℓ, using $O(n)$ space, where n is the number of rows in the coefficient matrix. Finally, we gave examples of how the S-level incomplete fill path theorem can be used as an analytical tool to bound the amount of fill in various types of graphs.

REFERENCES

[1] O. Axelsson, *Iterative Solution Methods*, Cambridge University Press, Cambridge, UK, 1994.

[2] E. Chow and Y. Saad, *Experimental study of ILU preconditioners of indefinite matrices*, J. Comput. Appl. Math, 86 (1997), pp. 387–414.

[3] T. H. Cormen, C. E. Leiserson, and R. L. Rivest, *Introduction to algorithms*, McGraw-Hill, San Francisco, CA, 1990.

[4] E. F. D'Azevedo, P. A. Forsyth, and W.-P. Tang, *Towards a cost-effective ILU preconditioner with high level fill*, BIT, 32 (1992), pp. 442–463.

[5] S. C. Eisenstat and J. W. H. Liu, *Exploiting structural symmetry in unsymmetric sparse symbolic factorization*, SIAM J. Mat. Anal. and App., 13 (1992), pp. 202–211.

[6] J. R. Gilbert and J. W. H. Liu, *Elimination structures for unsymmetric sparse LU factors*, SIAM J. Mat. Anal. and App., 14 (1993), pp. 334–352.

[7] I. Gustafsson, *A class of first-order factorization methods*, BIT, 18 (1978), pp. 142–156.

[8] D. Hysom, *New Sequential and Scalable Parallel Algorithms for Incomplete Factor Preconditioning*, PhD thesis, Old Dominion University, December 2001.

[9] D. Hysom and A. Pothen, *A scalable parallel algorithm for incomplete factor preconditioning*, SIAM J. Sci. Comput., 22 (2001), pp. 2194–2215.

[10] J. W. H. Liu, *The role of elimination trees in sparse factorization*, SIAM J. Mat. Anal. and App., 11 (1990), pp. 134–172.

[11] J. Meijerink and H. van der Vorst, *An iterative solution method for linear systems of which the coefficient matrix is a symmetric M-matrix*, Math. Comp., 31 (1977), pp. 148–162.

[12] S. Parter, *The use of linear graphs in Gauss elimination*, SIAM Review, 3 (1961), pp. 119–130.

[13] D. J. Rose and R. E. Tarjan, *Algorithmic aspects of vertex elimination on directed graphs*, SIAM J. Appl. Math., 23 (1978), pp. 176–197.

[14] D. J. Rose, R. E. Tarjan, and G. S. Lueker, *Algorithmic aspects of vertex elimination on graphs*, SIAM J. Comput., 5 (1976), pp. 266–283.

[15] Y. Saad, *Iterative Methods for Sparse Linear Systems*, PWS Publishing Company, 20 Park Plaza, Boston, MA 02116, 1996.