Row Ordering Schemes for Sparse Givens Transformations
I. Bipartite Graph Model

Alan George
Dept. of Computer Science, Univ. of Waterloo

Joseph Liu
Dept. of Computer Science, York Univ.

Esmond Ng
Dept. of Computer Science, Univ. of Waterloo

CS-83-13
May 1983

Row Ordering Schemes for Sparse Givens Transformations

I. Bipartite Graph Model

Alan George[1]

Department of Computer Science
University of Waterloo

Joseph Liu[2]

Department of Computer Science
York University

Esmond Ng[3]

Department of Computer Science
University of Waterloo

ABSTRACT

Let A be an m by n matrix, $m \geq n$, and let P_r and P_c be permutation matrices of order m and n respectively. Suppose $P_r A P_c$ is reduced to upper trapezoidal form $\begin{pmatrix} R \\ O \end{pmatrix}$ using Givens rotations, where R is n by n and upper triangular. The sparsity structure of R depends only on P_c. For a fixed P_c, the number of arithmetic operations required to compute R depends on P_r. In this paper, we consider row ordering strategies which are appropriate when P_c is obtained from nested dissection orderings of $A^T A$. Recently, it was shown that so-called "width-2" nested dissection orderings of $A^T A$ could be used to simultaneously obtain good row and column orderings for A. In this paper, we show that the conventional (width-1) nested dissection orderings can also be used to induce good row orderings. In part I of this paper, we analyze the application of Givens rotations to a sparse matrix A using a bipartite graph model.

[1] Research of the first author was supported in part by the Canadian Natural Sciences and Engineering Research Council under Grant A8111.

[2] Research of the second author was supported in part by the Canadian Natural Sciences and Engineering Research Council under Grant A5509.

[3] Research of the third author was supported in part by the Canadian Natural Sciences and Engineering Research Council through a postgraduate scholarship.

1. Introduction

In computational linear algebra, one of the standard methods for computing the orthogonal decomposition of a given matrix is by successive applications of *Givens rotations*. The decomposition is useful in the solution of linear least squares problems [11].

It has been generally recognized that the use of Givens transformations adapts quite well when applied to large sparse systems. The process can be organized in the form of "row eliminations" [4,6], so that the data and storage management is simple and efficient in both in-core and out-of-core implementations. Moreover, the use of Givens rotations can often reduce the amount of computation required to effect the decomposition.

In this paper, we study row ordering schemes for sparse Givens transformations. Let A be a large sparse m by n matrix $(m \geq n)$, which is of full column rank. The orthogonal decomposition of A obtained via Givens rotations can be expressed as

$$A = Q \begin{pmatrix} R \\ O \end{pmatrix} ,$$

where Q is an m by m orthogonal matrix defined by the sequence of rotations, and R is an n by n upper triangular matrix. Since

$$A^T A = \begin{pmatrix} R^T & O \end{pmatrix} Q^T Q \begin{pmatrix} R \\ O \end{pmatrix} = \begin{pmatrix} R^T & O \end{pmatrix} \begin{pmatrix} R \\ O \end{pmatrix} = R^T R \quad ,$$

and $A^T A$ is symmetric and positive definite, the upper triangular matrix R is mathematically and structurally the same as the Cholesky factor of $A^T A$, apart from possible sign differences in some rows.

It is well known that ordering can drastically reduce storage and computation in the Cholesky factorization of symmetric and positive definite matrices [7]. In the context of orthogonal decomposition, it is therefore important to find a good fill-reducing ordering for the matrix $A^T A$, so that the permuted system has a sparse Cholesky factor R. Since symmetric permutations of $A^T A$ correspond to *column orderings* of A, we may then view the process as follows:

a) find a "good" column permutation P_c for A and form AP_c,

b) apply a sequence of Givens rotations to decompose

$$AP_c = Q \begin{pmatrix} R \\ O \end{pmatrix} .$$

In this way, the resulting upper triangular matrix R will correspond to the factor of $(AP_c)^T (AP_c)$. It should be pointed out that once the structure of $A^T A$ is determined, there are well-developed sparse matrix software packages [2,7] available to produce a good fill-reducing permutation P_c.

Column orderings of A can indeed have a drastic effect on the number and locations of nonzeros in the resulting upper triangular factor R. However, the number and locations of nonzeros in R does not depend on the order in which the rows are processed [5,13]. Consider any row ordering P_r of A. Since

$$(P_r A)^T (P_r A) = A^T A \quad ,$$

the factor from the orthogonal decomposition of $P_r A$ must be mathematically and structurally the same as the corresponding factor from A.

However, row orderings can have significant impact on the amount of computation required to compute the decomposition [4,5]. In this paper, we analyze how row orderings can affect the arithmetic cost. The approach taken is combinatorial in nature, and we use the bipartite graph of A to model the orthogonal decomposition process by Givens transformations. This provides insight into the process and suggests ways in which good row orderings can be determined.

An outline of the paper is as follows. In Section 2, we introduce the necessary terminology for bipartite graphs and some preliminary results. A graph-theoretic analysis of Givens transformations using bipartite graphs is presented in Section 3. In Section 4, we consider the different row ordering strategies that are motivated by the results of Section 3. In particular, we introduce the concept of clique separators. The good performance of "width-2" nested dissection orderings of $A^T A$, introduced by George and Ng [9], in producing good row and column orderings for A can be explained in terms of clique separators. Moreover, we show that the conventional (width-l) nested dissection column ordering can also be used to induce good row orderings using clique separators. Indeed, the latter approach can often produce a better column and row ordering than that produced by width-2 nested dissection orderings of $A^T A$.

In part III of this paper [8], we shall analyze the two strategies on a k by k model grid problem. Although the orthogonal decompositions in both approaches require $O(k^3)$ arithmetic operations, and the resulting factors have $O(k^2 \log_2 k)$ nonzeros, the constant of proportionality is much less when the conventional width-1 dissection column ordering (and its induced row ordering) is used.

2. Graph Terminology

We assume that the reader is familiar with the basic graph theory terminology associated with sparse matrix computations. In particular, the concepts of symmetric graphs, nodes, edges, adjacent sets, degrees, paths, connectivity, cliques, and reachable sets are assumed. Readers are referred to [7] for formal definitions.

In this section, we introduce a bipartite graph model, which is a convenient tool in the study of row orderings for Givens transformations. Gilbert [10] has used bipartite graphs to model Gaussian elimination of unsymmetric sparse matrices.

Let A be an m by n sparse matrix of full column rank, where $m >> n$. The *bipartite graph* $H(A)$ of A is defined to be the graph $(Q(A), X(A), B(A))$, where

$$Q(A) = \left\{ q_1, q_2, \cdots, q_m \right\}$$

and

$$X(A) = \left\{ x_1, x_2, \cdots, x_n \right\} \quad .$$

Moreover, $\{q_r, x_c\}$ is in $B(A)$ if and only if a_{rc} is nonzero. Here, $B(A)$ is the edge set of the bipartite graph, while $Q(A)$ and $X(A)$ are the node sets corresponding respectively to the rows and columns of A.

Consider the 7 by 4 matrix example in Figure 2.1. The corresponding bipartite graph $H(A)$ is as shown in the same figure, where $Q(A)$ has 7 nodes, $X(A)$ has 4 nodes, and there are 14 edges in the graph.

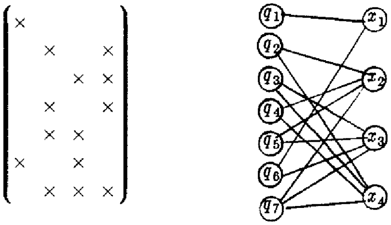


Figure 2.1: A 7 by 4 matrix example and its associated bipartite graph.

Let M be the n by n symmetric and positive definite matrix defined by A^TA. Consider the symmetric graph $G(M)$ associated with M; i.e., $G(M)$=($X(M)$,$E(M)$) where

$$X(M) = X(A) = \left\{x_1,x_2, \cdots ,x_n\right\}$$

and $\{x_i,x_j\}$ is in $E(M)$ if and only if m_{ij} is nonzero.

When the matrices are clear from context, we shall use H, Q, X and B to stand for $H(A)$, $Q(A)$, $X(A)$, and $B(A)$ respectively. Furthermore, for the symmetric matrix M, we use G and E to represent $G(M)$ and $E(M)$ respectively.

The following lemmas relate the structure of the bipartite graph $H(A)$ with that of the symmetric graph $G(A^TA)$. We need to define a slight modification of the notion of reachable set as introduced in [7]. This extension is useful in dealing with bipartite graphs.

Let G be a given graph. The *reachable set* of x in T through S, denoted by $Reach_G(x,S,T)$, is defined to be the set of nodes in T–($S\bigcup\{x\}$) that are reachable from x through S.

Lemma 2.1:

For each q_r in Q, $Adj_H(q_r)$ forms a clique in $G(M)$.

Proof:

For any x_j, x_k in $Adj_H(q_r)$, this implies a_{rj} and a_{rk} are both nonzeros. Therefore,

$$m_{jk} = a_{rj}\,a_{rk} + \cdots$$

must be nonzero (assuming the usual no-cancellation rule). Hence, $\{x_j,x_k\}$ is in $E(M)$.

□

Lemma 2.2:

$$Adj_G(x) = Reach_H(x, Q, X).$$

□

Consider the matrix example A in Figure 2.1. The structure of the matrix $A^T A$ and its symmetric graph are illustrated in Figure 2.2. In this example,

$$Adj_H(q_4) = \{x_2, x_4\} \ ,$$

$$Adj_H(q_7) = \{x_2, x_3, x_4\} \ ,$$

and they are both cliques in the symmetric graph $G(A^T A)$.

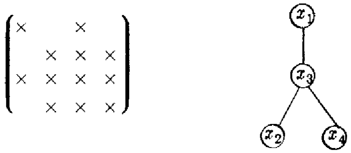


Figure 2.2: Matrix structure of $A^T A$ and its symmetric graph.

3. Graph-Theoretic Analysis of Givens Transformations

3.1. Givens Transformation

Consider row p and row r in the sparse matrix A, where the column subscripts of the first nonzero elements of these two rows have the same value c.

$$\begin{array}{lccccccccccccccl}
\text{row } p & 0 & \cdots & 0 & \times & 0 & 0 & 0 & \times & \times & 0 & 0 & 0 & \times & 0 & \times \quad \text{pivot row} \\
\text{row } r & 0 & \cdots & 0 & \times & 0 & \times & 0 & 0 & \times & 0 & 0 & \times & \times & \times & \times \\
 & & & & c & & & & & & & & & & &
\end{array}$$

A Givens rotation can be constructed and applied to these two rows to annihilate the first nonzero in one of the two rows, say row r. Row p is called the *pivot row*. In this way, these two rows are transformed, and the structure of the remaining parts of the transformed rows is the *union* of those of the original [4,5,13].

In the example, the transformed rows become

$$\begin{array}{lccccccccccccccc}
\text{row } p & 0 & \cdots & 0 & \underline{\times} & 0 & \underline{\times} & 0 & \underline{\times} & \underline{\times} & 0 & 0 & \underline{\times} & \underline{\times} & \underline{\times} & \underline{\times} \\
\text{row } r & 0 & \cdots & 0 & \underline{0} & 0 & \underline{\times} & 0 & \underline{\times} & \underline{\times} & 0 & 0 & \underline{\times} & \underline{\times} & \underline{\times} & \underline{\times} \\
 & & & & c & & & & & & & & & & &
\end{array}$$

In the transformation, only entries underlined in the two rows are modified. The remaining rows of the matrix do not change.

We will use the notation $[p,r,c]$ to represent the Givens rotation on row p and row r of the matrix, with row p as the pivot row and c as the column index of the common first nonzero entry in the two rows. When the column subscript c is the same as p, we shall use $[p,r]$ to represent the rotation.

Row Elimination Sequence

One way of applying a sequence of Givens rotations to transform the given matrix A into upper trapezoidal form is to process the entries row by row from row 2 to row m. This means the sequence can be depicted by the following.

$$\begin{array}{c}
[1,2,1] \\
\\
[1,3,1] \\
[2,3,2] \\
\\
[1,4,1] \\
[2,4,2] \\
[3,4,3] \\
\\
\cdots \\
\\
[1,m,1] \\
[2,m,2] \\
[3,m,3] \\
\cdots \\
[n,m,n]
\end{array}
\qquad\qquad
\begin{array}{cccc}
\times & & & \\
1 & \times & & \\
2 & 3 & \times & \\
4 & 5 & 6 & \times \\
7 & 8 & 9 & 10 \\
11 & 12 & 13 & 14 \\
15 & 16 & 17 & 18 \\
19 & 20 & 21 & 22 \\
23 & 24 & 25 & 26 \\
27 & 28 & 29 & 30
\end{array}$$

Case when $m=10,n=4$

Column Elimination Sequence

Another sequence of Givens rotations can be used to process the matrix column by column. This sequence can be depicted by our notation as follows.

[1,2,1]
[1,3,1]
[1,4,1]
. . .
$[1,m,1]$

[2,3,2]
[2,4,2]
. . .
$[2,m,2]$

. . .

$[n,n+1,n]$
$[n,n+2,n]$
. . .
$[n,m,n]$

$$\begin{array}{cccc}
\times & & & \\
1 & \times & & \\
2 & 10 & \times & \\
3 & 11 & 18 & \times \\
4 & 12 & 19 & 25 \\
5 & 13 & 20 & 26 \\
6 & 14 & 21 & 27 \\
7 & 15 & 22 & 28 \\
8 & 16 & 23 & 29 \\
9 & 17 & 24 & 30
\end{array}$$

Case when $m=10,n=4$

The two elimination sequences are equivalent in terms of arithmetic operations and final fill-in. The row scheme is preferred in practice because it has advantages with respect to implementation. Specifically, when processing the $r-th$ row, we only need a temporary vector of length n to annihilate the necessary entries in the current row. This temporary vector can be re-used for each successive row until the entire reduction is completed.

However, in the following analysis of the Givens rotations, we shall use the column scheme since it can be modelled in terms of bipartite graph transformations.

It should be noted that in the above two sequences of annihilation, the diagonal entries are assumed to be nonzeros. This makes the rotations $[i,j,i]$ for $i=1,\cdots,n$ and $j=i+1,\cdots,m$ meaningful. However, in the case of *sparse* transformations, this condition may not be satisfied. To simplify our analysis, we assume that the sparse matrix has nonzero diagonals. The results can be extended to the general sparse case.

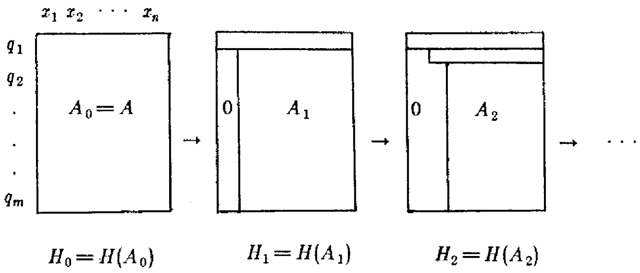
3.2. The Bipartite Graph Model

Let A be an m by n given sparse matrix of full rank, where $m>>n$. We assume that the matrix A has *nonzero diagonal* entries. In other words, the pivot rows in the orthogonal transformation are always selected from the first n rows in the matrix. We leave the extension to the general case to the reader.

Consider the use of a sequence of Givens rotations

$$[i,i+1]\ ,\ [i,i+2]\ ,\ \cdots\ ,\ [i,m-1]\ ,\ [i,m]\ \ ,$$

for $i=1,\cdots,m$, to annihilate nonzero elements in the lower trapezoidal portion of the matrix *column by column*. Obviously, the rotation is only performed if the corresponding entry is nonzero at the time of annihilation.



A sequence of bipartite graphs

$$H_i = H(A_i) = (Q_i, X_i, B_i) \ , \qquad i = 1,2, \cdots ,n \ ,$$

can be associated with the sequence of submatrices A_i remaining to be processed. Obviously, the node set for the rows of H_i is

$$Q_i = \left\{ q_{i+1}, \cdots , q_m \right\}$$

and the node set for the columns of H_i is

$$X_i = \left\{ x_{i+1}, \cdots , x_n \right\} \ .$$

This sequence of graphs can be used to model the decomposition process, and in the next subsection we shall analyze Givens rotations using this model.

We now give an example. Consider the 7 by 4 matrix structure in Figure 2.1. The annihilation process by Givens rotations is shown in Figure 3.1 and the corresponding bipartite graph sequence in Figure 3.2.

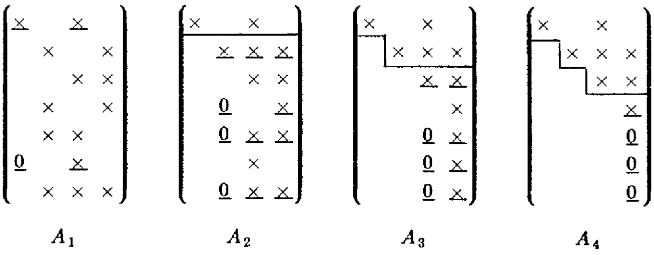


Figure 3.1: Sequence of matrix structures in the column-wise decomposition.

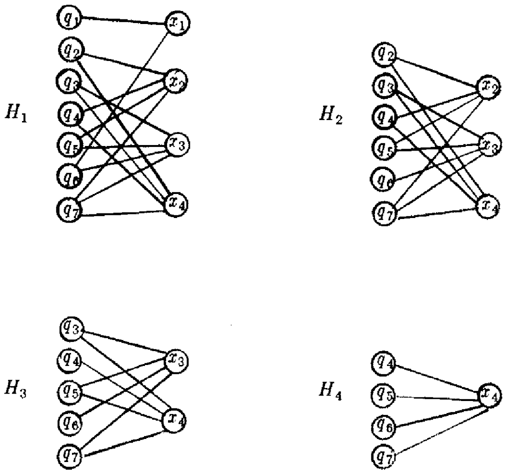


Figure 3.2: Sequence of bipartite graphs in the column-wise decomposition.

3.3. Analysis Using Bipartite Graphs

The bipartite graph model is useful in representing the transformation process when sparse Givens rotations are applied. In the sequence of matrix transformations

$$A = A_0 \rightarrow A_1 \rightarrow A_2 \rightarrow \cdots \rightarrow A_n = R \quad ,$$

zeros may become nonzeros. They can be classified into two types: those that correspond to final fill-in in the upper triangular factor R, and those that will eventually be annihilated by subsequent Givens transformations. The latter has been referred to as "intermediate fill-in" [9].

Fill-in (permanent or intermediate) can be determined by the zero-nonzero structures of the submatrices A_i. In this section, the main result is a partial characterization of the structures of the A_i's in terms of the original matrix structure. In order not to obscure the main result with a sequence of formal and lengthy lemmas and corollaries, we have left the proofs of the theorems to the appendix. We simply state the theorems and some related observations.

Here, we assume that the row and column orderings of the matrix A are fixed. Denote the node corresponding to row i by q_i, $1 \leq i \leq m$. Similarly, denote the node corresponding to column j by x_j, $1 \leq j \leq m$.

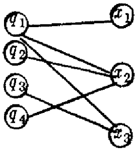
Theorem 3.1:

For $r > i$, $Adj_{H_i}(q_r) \subseteq Reach_{H_0}(q_r, \{x_1, \cdots, x_i, q_1, \cdots, q_{r-1}\}, X)$.

$\square$

In Theorem 3.1, the adjacent set on the left-hand side represents the structure of the r-*th* row in the submatrix A_i. The reachable set on the right-hand side is specified in the *initial* bipartite graph H_0; that is, in the structure of the original matrix A. Although this result does not characterize the complete structure of the r-th row of A_i, it provides a *necessary* condition for a zero element in the original matrix A to become nonzero during the course of Givens transformation. That this condition is not sufficient can be illustrated by the following example.

$$\begin{pmatrix} \times & \times & \times \\ & \times & \\ & & \times \\ & \times & \end{pmatrix}$$



Consider the path in $H(A)$:

$$(q_4, x_2, q_1, x_3)$$

which means

$$x_3 \in Reach_{H(A)}(q_4, \{x_1, x_2, q_1, q_2, q_3\}, X) \ .$$

However, it is straightforward to verify that *no* zero element in A becomes nonzero in reducing A to upper trapezoidal form by Givens transformations.

The next theorem gives a sufficient condition. Let

$$x_c \in Reach_{H_0}(q_r, \{x_1, \cdots, x_i, q_1, \cdots, q_{r-1}\}, X) \ .$$

Theorem 3.2:

If the path

$$(q_r, x_{g(1)}, q_{f(1)}, x_{g(2)}, q_{f(2)}, \cdots, x_{g(t)}, q_{f(t)}, x_c)$$

in H_0 through $\{x_1, \cdots, x_i, q_1, \cdots, q_{r-1}\}$ satisfies

$$g(k) \leq f(k) \ \text{, for } k=1, \cdots, t \ ,$$

and

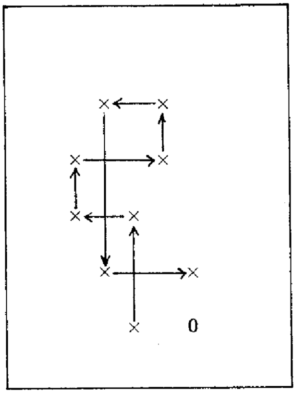
$$g(k+1) \leq f(k) \ \text{, for } k=1, \cdots, t-1 \ ,$$

then

$$x_c \in Adj_{H_i}(q_r) \ .$$

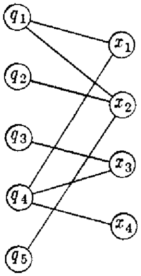
□

In terms of the matrix, the sufficient condition in Theorem 3.2 is equivalent to a path that only uses the *lower trapezoidal* portion of the matrix. This is illustrated in the following figure.



However, the condition in Theorem 3.2 is not necessary. This can be illustrated by the following example.

$$\begin{pmatrix} \times & \times & & \\ & \times & & \\ & & \times & \\ \times & & \times & \times \\ & \times & & \end{pmatrix}$$



Consider the path $(q_5,x_2,q_1,x_1,q_4,x_3)$. It does not satisfy the condition in Theorem 3.2 and it is the only path from q_5 to x_3. However, it is straightforward to verify that

$$x_3 \in Adj_{H_2}(q_5) \quad .$$

3.4. Complexity of Sparse Givens Transformation

The complexity of the orthogonal transformation of a sparse matrix by Given rotations can be expressed quite readily in terms of the sequence of bipartite graphs $\{H_i\}$. There are two quantities that are of interest in the analysis: the number of Givens rotations and the number of multiplicative operations required to effect the orthogonal transformation.

Theorem 3.3:

The number of Givens rotations required to reduce A to upper trapezoidal form is

$$\sum_{i=1}^{n} \left\{ |Adj_{H_{i-1}}(x_i)| - 1 \right\} .$$

Proof:

In the transformation from H_{i-1} to H_i, the number of nonzeros that need to be annihilated is given by

$$|Adj_{H_{i-1}}(x_i)| - 1 .$$

The result then follows by summing over i.

□

Lemma 3.4:

In the bipartite graph H_{i-1}, let

$$q_r \in Adj_{H_{i-1}}(x_i) \;, \qquad r > i \;.$$

Then the number of multiplicative operations required to annihilate this entry is given by

$$4 \left\{ \, |Adj_{H_i}(q_r)| + 1 \right\} .$$

Proof:

Left as an exercise. (Note that we assume two multiplicative operations are used to compute each new value.)

□

Theorem 3.5:

The number of multiplicative operations required to reduce A to upper trapezoidal form is

$$4 \sum_{i=1}^{n} \sum_{q_r \in Adj_{H_{i-1}}(x_i)} \left\{ |Adj_{H_i}(q_r)| + 1 \right\} .$$

Proof:

In transforming H_{i-1} to H_i, the row q_r requires processing if and only if

$$q_r \in Adj_{H_{i-1}}(x_i) \;.$$

Thus, the result follows from Lemma 3.4.

□

Corollary 3.6:

When the matrix A is n by n and full, the number of Givens rotations required is $(1/2)n^2 + O(n)$, and the number of operations required is $(4/3)n^3 + O(n^2)$.

□

4. Row Ordering Strategies

4.1. Node and Clique Separators

The analysis in Section 3 assumes that the row and column orderings of the matrix A are given or are pre-determined by other means. Since in the QR decomposition of the matrix A, R is the Cholesky factor of the symmetric and positive definite matrix A^TA, it is clear that we want to order the columns of the matrix A (that is, a symmetric ordering of the matrix A^TA) so that fill-in in factoring A^TA can be reduced.

However, the row ordering of the matrix A does not have any effect on the zero-nonzero structure of the factor R. Nevertheless, the choice of a good row ordering does have a significant impact on the arithmetic operations required to compute the QR decomposition by Givens transformations [5]. In part III of this paper, we shall illustrate a bad row ordering for the k by k grid model problem with a good column ordering.

There is existing well-developed sparse matrix software for the solution of symmetric positive definite systems [2,7]. Among the ordering schemes, the most popular are the *minimum degree* algorithm and the *nested dissection* algorithm [7]. In this section, we consider the case when the matrix A^TA (and hence the columns of A) is ordered by the nested dissection scheme. We shall investigate the different possible ways in which the rows of A can be rearranged so as to reduce the number of (non-trivial) Givens rotations and the number of arithmetic operations necessary to perform the orthogonal decomposition. In [12], Golub and Plemmons addressed this row ordering problem in the context of block orthogonal decomposition of large scale geodetic least-squares systems.

The results of Section 3 provide some insight and motivation for the selection of row ordering strategies. In view of the relationship between separators, fill-in and reachable sets in the symmetric positive definite case [7], Theorem 3.1 can be used or interpreted as follows. To reduce or control the size of the set

$$Adj_{H_i}(q_r) \quad ,$$

(which implies a *reduction* in the number of rotations and arithmetic cost), one should try to reduce the size of the reachable set

$$Reach_{H_0}(q_r, \{x_1, \cdots, x_i, q_1, \cdots, q_{r-1}\}, X) \quad .$$

This can be achieved by the use of *separators*. Since there are two types of nodes involved in the bipartite graph, notably the x's and q's, we are looking for two types of separators. The approaches discussed in the remainder of this section are based on the idea of using *separators from the column nodes* $\{x_j\}$ and *separators from the row nodes* $\{q_i\}$. By separators from the column {row} nodes, we mean subsets of column {row} nodes whose removal, together with the incident edges, disconnect the bipartite graphs $H(A)$ into disjoint pieces.

The treatment here assumes that the columns are ordered by some form of nested dissection. Because of the recursive nature of nested dissection, all the different strategies for row orderings discussed in this section will also be applied recursively.

We begin by quoting, in our terminology, some results from George and Ng [9].

Lemma 4.1:

Consider $M = A^T A$, and $G = G(M)$. Let S be a separator in the graph $G(M)$ inducing two node components C_1 and C_2 in $G(X-S)$. Then, for each q_r in $Q(A)$,

$$Adj_{H(A)}(q_r)$$

is a subset of $C_1 \bigcup S$ or $C_2 \bigcup S$.

$\square$

To facilitate our discussion, we introduce clique separators (in the symmetric graph of $A^T A$). Their introduction is motivated by Lemma 2.1, which states that each row q_r of the matrix A corresponds to a clique in the symmetric graph $G(A^T A)$.

Let G be a given connected graph, and K be a subset of cliques of G. The set K is said to be a *clique separator* of G if, when (or after) all the *edges* in each clique of K are removed from G, the resulting graph has more than one connected component. Thus, a separator from row nodes in the bipartite graph $H(A)$ is equivalent to a clique separator in the symmetric graph $G(A^T A)$. The concept of clique separators will be used in some of the row ordering strategies described in this section.

4.2. Induced Row Ordering by Nested Dissection Column Ordering

A row ordering can be induced quite naturally from the corresponding nested dissection column ordering. It is based on the observation from Lemma 4.1.

Let S be a (node) separator in $G(A^T A)$ whose removal creates connected components C_1 and C_2. The induced row ordering is as follows:

Number all those rows q_r with $Adj_{H(A)}(q_r) \subseteq C_1 \bigcup S$,

followed by the remaining rows.

The structure of the resulting matrix can be depicted as follows:

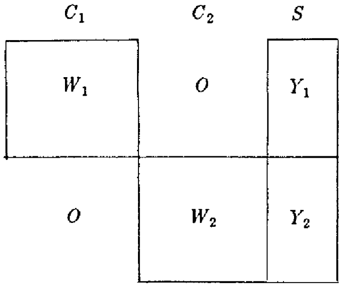


Figure 4.1: Matrix Structure on Induced Row Ordering

It is clear from Theorem 3.1 that the two zero submatrices remain zero throughout the entire orthogonalization process. However, zeros in Y_1 can become nonzeros due to those in W_1. Moreover, zeros in Y_2 can become nonzeros due to either those in submatrix W_2, or those in Y_1 (the "rectangular box effect" as discussed in the appendix). This form of blocking has been exploited by Golub and Plemmons [12] in the use of a *nested bisection* procedure for geodetic least squares systems.

Since the other strategies discussed in this section provide better alternatives, we will not consider this method further. However, one point worth noting is that, given the nested dissection column ordering, the corresponding row ordering (assuming that this idea is to be applied recursively) can be determined by simply sorting the rows in *ascending order* with respect to the *first nonzero subscripts*.

4.3. Width-1 clique separator

A possible extension to the above strategy is to exercise more care in the numbering of the rows that correspond to $C_2 \bigcup S$. The method can be viewed as identifying a clique separator from the row nodes and ordering the equations associated with this clique separator last. Indeed, the set of equations q_r, whose adjacent set $Adj_{H(A)}(q_r)$ is contained in $C_2 \bigcup S$ and intersects the separator S, forms a (width-1) clique separator. Numbering the node separator and this clique separator last helps to reduce the size of the reachable set

$$Reach_{H_0}(q_r, \{x_1, \cdots, x_i, q_1, \cdots, q_{r-1}\}, X)$$

in the same way as node separators in the solution of symmetric positive definite systems [7].

Let S be a (node) separator in $G(A^T A)$ whose removal creates connected components C_1 and C_2. This method can be described as follows:

First number all those rows q_r with $Adj_{H(A)}(q_r) \subseteq C_1 \bigcup S$.

Number next all those q_r with $Adj_{H(A)}(q_r) \subseteq C_2$.

Finally, number the remaining rows.

The structure of the resulting matrix can be depicted as follows:

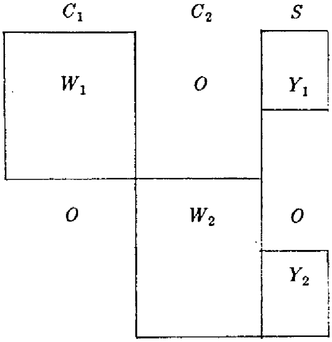


Figure 4.2: Matrix Structure By Width-1 Clique Separator

With this structure, it follows again by Theorem 3.1 that possible fill-in is limited to the submatrices W_1, W_2, Y_1 and Y_2.

4.4. Width-2 Clique Separator

A further refinement to the above strategies gives the next method. In fact, this corresponds to the heuristic method used by George and Heath [5]. This method can be regarded as applying the same idea in Section 4.2 to both the subsets $C_1 \bigcup S$ and $C_2 \bigcup S$.

Let S be a (node) separator in $G(A^T A)$ whose removal creates connected components C_1 and C_2. This method can be described as follows:

First number all those rows q_r with $Adj_{H(A)}(q_r) \subseteq C_1$

followed by those with $Adj_{H(A)}(q_r) \subseteq C_1 \bigcup S$.

Number next all those q_r with $Adj_{H(A)}(q_r) \subseteq C_2$.

Finally, number the remaining rows.

The structure of the resulting matrix can be depicted as follows. (This form of row arrangement has also been described by Golub and Plemmons [12].)

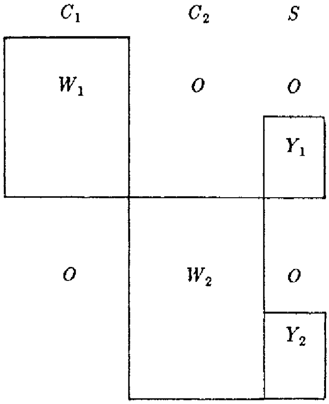


Figure 4.3: Matrix Structure for width-2 clique separator

The rows in the above structure can further be rearranged so that

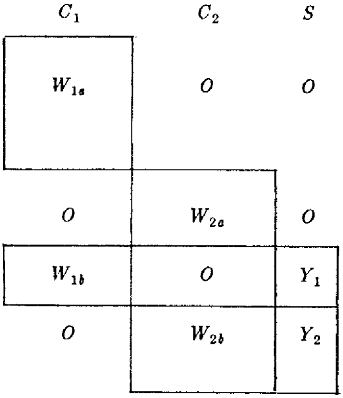


Figure 4.4: Different structure for width-2 clique separator

In other words, the row ordering strategy can be described as follows:

First number all those rows q_r with $Adj_{H(A)}(q_r) \subseteq C_1$.

Next number all those rows q_r with $Adj_{H(A)}(q_r) \subseteq C_2$.

Finally, number the remaining rows q_r. (Note that $Adj_{H(A)}(q_r)$ has a nonempty intersection with the separator S).

The strategy can also be interpreted in terms of clique separators. Indeed, the node separator S in $G(A^TA)$ can be used to identify the set of clique separators that corresponds to those rows q_r such that $Adj_{H(A)}(q_r)$ intersects S. The rows in this clique separator are to be numbered last to reduce the sizes of reachable sets.

In fact, the clique separator defined by the node separator S can be divided into two sets of clique separators, namely those in $C_1 \bigcup S$ and in $C_2 \bigcup S$. Each by itself is a clique separator. Therefore, together they form a width-2 clique separator (and hence the name).

But, how can the row ordering corresponding to a given nested dissection column ordering be determined easily? We shall consider this question in the remainder of this section.

We observe that the rows in the clique separator can be ordered "intermixably" from those in $C_1 \bigcup S$ and those in $C_2 \bigcup S$ without affecting the arithmetic cost for orthogonal decomposition. (Indeed, this can be explained by the "rectangular box" effect as described in appendix.)

In view of this observation, for a given nested dissection column ordering, the row ordering using the strategy of width-2 clique separator can be generated by sorting the rows in *ascending order of last nonzero subscript.* This may not generate the same matrix structure as depicted in Figure 4.4. However, this will be appropriate for the decomposition.

4.5. Width-2 Node Separator

In George and Ng [9], a mechanism is provided to order the columns of the matrix A, which automatically induces a good row ordering. It is based on the idea of *width-2* (*node*) *separators* [10].

A width-2 node separator is identified in the symmetric graph $G(A^TA)$, which defines a set of cliques that is "bounded" or "included" by the node separator. More specifically, the set

$$\left\{ q_r \mid Adj_{H(A)}(q_r) \subseteq S \right\}$$

defines a clique separator. This set of cliques is numbered last in the row ordering. The procedure is applied recursively to the remaining components.

Let S be a *width-2* node separator in $G(A^TA)$ with C_1 and C_2 as the induced connected components. The matrix structure of the resulting column/row ordering can be depicted as follows.

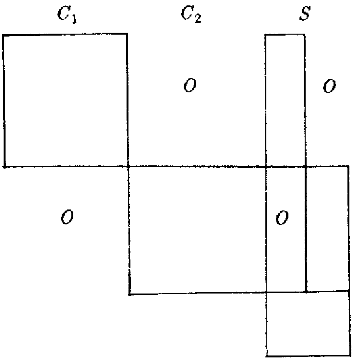


Figure 4.5: Matrix Structure for Width-2 Node Separator

The success of this scheme can again be explained by Theorem 3.1. It delays the numbering of node and clique separators in order to reduce the sizes of reachable sets.

The determination of the corresponding row ordering can be obtained by sorting the rows in ascending order of first nonzero subscript [9]. It is interesting to note that the scheme to determine the row ordering is the same as that in Section 4.3, although the two column dissection schemes are different.

5. Concluding Remarks

In this paper, we have used the bipartite graph model to study the combinatorial nature of the orthogonal reduction process by Givens rotations. The entire process can be modelled by a sequence of bipartite graphs. The number of non-trivial rotations and number of arithmetic operations can both be expressed in terms of adjacency relations in this sequence of bipartite graphs.

The result provides insight into the decomposition process. To reduce arithmetic cost (and also the number of rotations), it is important to order the row and columns

$$q_1, q_2, \cdots, q_m$$

$$x_1, x_2, \cdots, x_n$$

so that the reachable sets in the sequence:

$$Reach_{H(A)}(q_r, \{x_1, \cdots, x_i, q_1, \cdots, q_{r-1}\}, X)$$

are small in size. This motivates the introduction of clique separators (vs node separators).

The concepts of node and clique separators are used to study the various row and column ordering strategies. We have viewed the row and column orderings from "width-2" (node) nested dissection [9] in terms of these separators. We have also introduced the "width-2" clique nested dissection ordering, which can be interpreted as one with the columns ordered by the conventional (width-1) nested dissection and the rows by an induced ordering.

In part III of this paper [8], we shall provide a detailed analysis of the latter approach applied to a k by k regular grid model problem that arises in the *natural factor* formulation of the finite element method [1]. We show that this approach is better than the "width-2" (node) nested dissection strategy, both in terms of storage requirements and operation counts. The readers are referred to part III of this paper. Here, we conclude by presenting some results from experimental runs comparing these two approaches.

Matrix A			width-1			width-2		
n	m	# nonz	# nonz R	# opns	# Givens	# nonz R	# opns	# Givens
484	1764	7056	7646	2480779	27181	8181	2563009	29321
676	2500	10000	11367	4242850	41086	12676	4580162	46286
900	3364	13456	16321	6822042	59462	18397	7373369	66111

Table 5.1: Comparisons of row and column orderings using width-1 and width-2 (node) separators.

6. References

[1] J.H. Argyris and O.E. Bronlund, "The natural factor formulation of the stiffness matrix displacement method", Computer Methods in Applied Mech. and Eng. 5 (1975), pp. 97-119.

[2] S.C. Eisenstat, M.C. Gursky, M.H. Schultz, A.H. Sherman, "Yale Sparse Matrix Package I. the symmetric code", Int. J. Num. Methods in Enginerring, 18 (1982), pp1145-1151.

[3] W.M. Gentleman, "Least squares computations by Givens transformations without square roots", J. Inst. Maths. Appl. 12 (1973), pp. 329-336.

[4] W.M. Gentleman, "Row elimination for solving sparse linear systems and least squares problems", in *Proc. Dundee Conference on Numerical Analysis,* ed. G.A. Watson, Lecture Notes in Mathematics (506), Springer-Verlag (1975), pp. 122-133.

[5] J.A. George and M.T. Heath, "Solution of sparse linear least squares problems using Givens rotations", Linear Alg. and its Appl. 34 (1980), pp. 69-83.

[6] J.A. George, M.T. Heath and R.J. Plemmons, "Solution of large scale sparse least squares problems using auxiliary storage", SIAM J. Sci. Stat. Comput. 2 (1981), pp. 416-429.

[7] J.A. George and J.W.H. Liu, *Computer Solution of Large Sparse Positive Definite Systems,* Prentice-Hall Inc., Englewood Cliffs, N.J. (1981).

[8] J.A. George, J.W.H. Liu and E.G.Y. Ng, "Row ordering schemes for sparse Givens transformations III. Analysis of the grid model problem", in preparation.

[9] J.A. George and E.G.Y. Ng, "On row and column orderings for sparse least squares problems", SIAM J. Numer. Anal. 20 (1983), pp. 326-344.

[10] J.R. Gilbert, "Graph separator theorems and sparse Gaussian elimination", Report STAN-CS-80-833, Dept. of Computer Science, Stanford University, Stanford (1980).

[11] G.H. Golub, "Numerical methods for solving linear least squares problems", Numer. Math. 7 (1965), pp. 206-216.

[12] G.H. Golub and R.J. Plemmons, "Large scale geodetic least squares adjustments by dissection and orthogonal decomposition", Linear Alg. and its Appl. 34 (1980), pp. 3-28.

[13] Z. Zlatev, "Comparison of two pivotal strategies in sparse plane rotations", Comp. & Math. with Appls. 8 (1982), pp. 119-135.

7. Appendix

In this appendix, the formal proofs of the main theorems in Section 3.3 are presented. We follow the same notation as introduced in that section. In particular,

$$H_i = H(A_i) = (Q_i, X_i, B_i) \quad , \qquad i = 1,2, \cdots ,n \quad ,$$

is the sequence of bipartite graphs associated with the corresponding sequence of submatrices A_i remaining to be processed.

Lemma A.1:

For $r > i$,

$$Adj_{H_i}(q_r) = \begin{cases} Adj_{H_{i-1}}(q_r) \ , & \text{if } x_i \text{ not in } Adj_{H_{i-1}}(q_r) \ , \\ \bigcup \left\{ Adj_{H_{i-1}}(q_s) \mid i \le s \le r, x_i \in Adj_{H_{i-1}}(q_s) \right\} - \left\{ x_i \right\} \ , & \text{otherwise} \ . \end{cases}$$

Proof:

It follows from the transformation process using Givens rotations.

□

Lemma A.2:

For $r > i$, $Adj_{H_{i-1}}(q_r) \subseteq Adj_{H_i}(q_r) \bigcup \left\{ x_i \right\}$.

Proof:

It follows from Lemma A.1.

□

Corollary A.3:

For $r > i + j$, $Adj_{H_i}(q_r) \subseteq Adj_{H_{i+j}}(q_r) \bigcup \left\{ x_{i+1}, \cdots ,x_{i+j} \right\}$.

Proof:

By repeated applications of Lemma A.2.

□

Lemma A.4:

For $r > i$, $Adj_{H_i}(q_r) = Reach_{H_{i-1}}(q_r, \{x_i, q_i, q_{i+1}, \cdots, q_{r-1}\}, X)$.

Proof:

Case 1 : $x_i \notin Adj_{H_{i-1}}(q_r)$

By Lemma A.1,

$$Adj_{H_i}(q_r) = Adj_{H_{i-1}}(q_r) = Reach_{H_{i-1}}(q_r, \emptyset, X) = \text{R.H.S.} \ .$$

Case 2 : $x_i \in Adj_{H_{i-1}}(q_r)$

Consider $x_c \in Adj_{H_i}(q_r)$. If there exists $i \leq s \leq r$ such that $x_i, x_c \in Adj_{H_{i-1}}(q_s)$, then (q_r, x_i, q_s, x_c) is a path in H_{i-1} and hence $x_c \in$R.H.S.. On the other hand, if $x_c \in$R.H.S., then there exists a path $(q_r, x_i, q_{f(1)}, x_i, \cdots, q_{f(t)}, x_c)$ in H_{i-1}, with $i \leq f(k) < r$, for $k = 1, \cdots, t$. If $t = 0$, $x_c \in Adj_{H_{i-1}}(q_r) - \{x_i\} \subseteq$ L.H.S. If $t > 0$, $x_i, x_c \in Adj_{H_{i-1}}(q_{f(t)})$, for $i \leq f(t) < r$, so that by Lemma A.1, $x_c \in Adj_{H_{i-1}}(q_{f(t)}) - \{x_i\} \subseteq$ L.H.S. Thus, $x_c \in Adj_{H_i}(q_r)$.

□

Corollary A.5:

For $r > 1$, $x_c \in Adj_{H_i}(q_r)$ if and only if there exists a path of length 1 or 3 from q_r to x_c through $\{x_i, q_i, \cdots, q_{r-1}\}$ in H_{i-1}.

Proof:

By Lemma A.4, we can find a path

$$(q_r, x_i, q_{f(1)}, x_i, \cdots, q_{f(t)}, x_c)$$

in H_{i-1}, where $i \leq f(k) < r$ for $k = 1, \cdots, t$. If $t = 0$, (q_r, x_c) is a path of length 1. If $t > 0$, $(q_r, x_i, q_{f(t)}, x_c)$ is a path of length 3. The converse is obvious from Lemma A.4.

□

Corollary A.6:

For $r > i$ and $c > i$, $x_c \in Adj_{H_i}(q_r)$ if and only if either $x_c \in Adj_{H_0}(q_r)$ or for some $k \leq i$, there is a path (q_r, x_k, q_s, x_c) in H_{k-1} with $k \leq s < r$.

Proof:

only if part

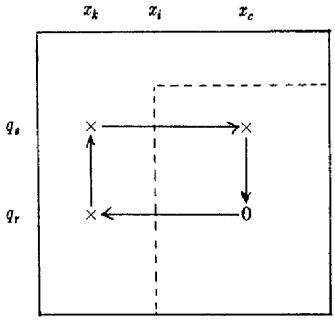
Consider the case when x_c is in $Adj_{H_i}(q_r)$ but not in $Adj_{H_0}(q_r)$. This means that the edge (q_r, x_c) must be created during the annihilation process. Let k be the smallest subscript such that $x_c \in Adj_{H_k}(q_r)$. By Corollary A.5, there is a path (q_r, x_k, q_s, x_c). in H_{k-1}, where $k < c$ and $k \leq s < r$.

if part

In the first case, by Corollary A.3, $x_c \in Adj_{H_0}(q_r) \subseteq Adj_{H_i}(q_r) \bigcup \{x_1, \cdots, x_i\}$. In the second case, by Lemma A.4, we have $k \leq i$ with $x_c \in Adj_{H_k}(q_r)$, so that again by Corollary A.3, $x_c \in Adj_{H_k}(q_r) \subseteq Adj_{H_i}(q_r) \bigcup \{x_{k+1}, \cdots, x_i\}$. In either case, $x_c \in Adj_{H_i}(q_r)$.

□

Pictorially, the result of Corollary A.6 can be illustrated as below:



The path (q_r, x_k, q_s, x_c) in H_{k-1} forms a "rectangular box" in the matrix. The conditions

$$k < c \quad \text{and} \quad s < r$$

mean that the rectangle must lie to the *left* of column c and *above* row r. Moreover, the inequality

$$k \leq s$$

implies that the upper left "corner" of this rectangle must not be above the matrix diagonal.

Another remark is that this path is in the bipartite graph H_{k-1}, which implies that the edges

$$(q_r, x_k)\ ,\ (q_s, x_k)\ ,\ \text{and}\ (q_s, x_c)$$

can either be in the original bipartite graph H_0 or can be created earlier in the process.

Proof of Theorem 3.1:

We prove this by induction on i. The result is obviously true when $i=0$, since

$$x_c \in Adj_{H_0}(q_r) = Reach_{H_0}(q_r, \emptyset, X) \Longrightarrow x_c \in \text{R.H.S.} \ .$$

Assume the result holds for values less than i. Consider $x_c \in Adj_{H_i}(q_r)$. By Corollary A.6, there are two cases.

Case 1: $x_c \in Adj_{H_0}(q_r) \subseteq$R.H.S.

Case 2: there is a path (q_r, x_k, q_s, x_c) in H_{k-1}, for some $k \leq i$ and $k \leq s < r$. This implies

$$\{q_r, x_k\} \ , \ \{q_s, x_k\} \ , \text{ and } \{q_s, x_c\}$$

are edges in the bipartite graph H_{k-1}. By inductive assumption,

$$x_k \in Reach_{H_0}(q_r, \{x_1, \cdots, x_{k-1}, q_1, \cdots, q_{r-1}\}, X) \ ,$$

$$x_k \in Reach_{H_0}(q_s, \{x_1, \cdots, x_{k-1}, q_1, \cdots, q_{s-1}\}, X) \ , \text{ and}$$

$$x_c \in Reach_{H_0}(q_s, \{x_1, \cdots, x_{k-1}, q_1, \cdots, q_{s-1}\}, X) \ .$$

Since $s < r$ and $k \leq i < c$, we can link these paths together to obtain one that goes from q_r to x_c through nodes in

$$\{x_1, \cdots, x_i, q_1, \cdots, q_{r-1}\} \ .$$

Indeed, this path goes through x_k and q_s. Hence

$$x_c \in Reach_{H_0}(q_r, \{x_1, \cdots, x_i, q_1, \cdots, q_{r-1}\}, X) \ .$$

□

Proof of Theorem 3.2:

We prove the result by induction on t. For $t=1$, we have the path

$$(q_r, x_{g(1)}, q_{f(1)}, x_c)$$

in H_0 with

$$g(1) \leq f(1) < r \ \text{ and } \ g(1) \leq i \ .$$

This path is also in the bipartite graph $H_{g(1)-1}$. Then by Corollary A.5, $x_c \in Adj_{H_{g(1)}}(q_r)$. Hence, by Corollary A.3, we obtain the result.

Assume that the result holds for values less than t. Consider the given path

$$(q_r, x_{g(1)}, q_{f(1)}, x_{g(2)}, q_{f(2)}, \cdots, x_{g(t)}, q_{f(t)}, x_c) \ .$$

Let $q_{f(k^*)}$ be the node in this path that has the largest subscript among the q's; that is, $f(k) \leq f(k^*)$, for $k=1, \cdots, t$. Furthermore, let $x_{g(h^*)}$ be the one with the largest subscript among the x's; that is, $g(h) \leq g(h^*)$, for $h=1, \cdots, t$. Then

$$(q_r, x_{g(1)}, q_{f(1)}, \cdots, x_{g(h^*)}) \ ,$$

$$(x_{g(k^*)}, q_{f(h^*)}, \cdots, q_{f(k^*)}) \ ,$$

and

$$(q_{f(k^*)}, x_{g(k^*+1)}, \cdots, x_c)$$

are three paths in H_0 that are of length less than t and satisfy the induction conditions. By the inductive assumption, we have

$$x_{g(h^*)} \in Adj_{H_u}(q_r) \ ,$$

$$x_{g(h^*)} \in Adj_{H_v}(q_{f(k^*)}) \ ,$$

and

$$x_c \in Adj_{H_w}(q_{f(k^*)}) \ ,$$

for some $u, v, w < g(h^*)$. By Corollary A.3, we have

$$x_{g(h^*)} \in Adj_{H_{g(h^*)-1}}(q_r) \ ,$$

$$x_{g(h^*)} \in Adj_{H_{g(h^*)-1}}(q_{f(k^*)}) \ ,$$

and

$$x_c \in Adj_{H_{g(h^*)-1}}(q_{f(k^*)}) \ .$$

In other words, $(q_r, x_{g(h^*)}, q_{f(k^*)}, x_c)$ is a path in $H_{g(h^*)-1}$ with $g(h^*) \leq f(k^*) < r$. By Corollary A.6,

$$x_c \in Adj_{H_{g(h^*)}}(q_r) \ ,$$

and again by Corollary A.3,

$$x_c \in Adj_{H_t}(q_r) \ .$$

$\square$