COMPUTATIONAL COMPLEXITY ANALYSIS OF FFT PRUNING - A MARKOV MODELING APPROACH

Robert J. Baxley and G. Tong Zhou

School of Electrical & Computer Engineering
Georgia Institute of Technology
Atlanta, GA 30332-0250, USA

ABSTRACT

The Fourier transform is instrumental in many signal processing applications such as digital filtering, spectral analysis and communications. In 1965, Cooley and Tukey demonstrated that the discrete Fourier transform (DFT) can be computed using the fast Fourier transform (FFT) algorithm with reduced computational complexity. When the input vector to the FFT contains mostly zeros (i.e., is sparse), it is possible to realize computational savings over a full FFT by only performing the arithmetic operations on non-zero elements. That is, the FFT is "pruned" so that only the useful computations are performed. In this paper, we derive the (non-stationary) Markov process that describes the number of occupied (i.e. non-zero) paths at each stage of a pruned FFT. With the probability distribution of the number of non-zero paths at each FFT stage, we then determine the probability distribution of the number of multiplications and additions necessary to compute the FFT of an input vector with a given sparsity distribution.

1. INTRODUCTION

The Fourier transform is instrumental in many signal processing applications such as digital filtering, spectral analysis and communications. In [1], Cooley and Tukey demonstrated that the discrete Fourier transform (DFT) can be computed using the fast Fourier transform (FFT) algorithm, which reduces the computational complexity of an N-point DFT from $\mathcal{O}(N^2)$ to $\mathcal{O}(N \log N)$, where the base of the logarithm is the radix of the FFT. However, there are some applications where the input vector to the FFT, $\{x[n]\}_{n=0}^{N-1}$, has a relatively small number of non-zero values. For example, signal interpolation can be performed using an "oversampled" FFT, where the FFT of a vector padded with a block of M zeros is encountered.

When the input vector to the FFT is sparse, it is possible to realize computational savings over a full FFT by only performing the arithmetic operations on non-zero elements. Equivalently, this can be thought of as "pruning" the zero paths in the FFT flow graph. FFT pruning was first studied in [2] and several algorithms have been proposed to implement pruned FFTs [2–7]. However, most of these algorithms and analyses are based on pruned FFTs where the zeros in the input vector are contiguous.

There are applications which require the FFT of unstructured sparse vectors. That is, the input vector to the FFT will contain a large number of zeros that are not organized in any systematic way. The FFT of a sparse vector is called an input-pruned FFT. Conversely, the FFT of a full vector where only several output values need to be calculated is called an output-pruned FFT [2]. Input- and output-pruned FFTs occur in diverse applications including crest factor reduction in multicarrier communications [8], cognitive radio [9] and genetic sequence alignment applications [10].

Fig. 1 is a flow graph of an example input-pruned FFT, where the input vector contains only two non-zero values $x[4]$ and $x[13]$. The bold paths in the flow graph indicate the paths that need to be considered in the calculation of the FFT output. The paths with a power of $W = e^{-j2\pi/N}$, $j = \sqrt{-1}$, above them indicate that a complex multiplication is required to compute the value of that path. Also, the bold paths that terminate in a filled circle indicate that an addition is necessary to compute the value at the end of that stage, whereas an empty bold circle indicates that the value at the end of that stage is simply a copy or a sign-reversed copy of a value at the beginning of the stage. That is, the bold circles indicate an addition is necessary and the empty bold circles indicate that no arithmetic operation is necessary. With this, Fig. 1 shows that multiplications [1] are only necessary at stages one and three, while additions are only necessary at stage three.

In [10] upper and lower bounds on the number of butterfly computations required to compute the FFT of an arbitrarily sparse input vector were given, where one butterfly operation requires the computation of one multiplication and two addi-

This work was supported in part by the U. S. Army Research Laboratory under the Collaborative Technology Alliance Program, Cooperative Agreement DAAD19-01-2-0011 and by the National Science Foundation Graduate Research Fellowship Program.

[1] Since a complex sign change can be implemented with a two bit flipping operations we do not take into account sign changes in our complexity analysis.

1-4244-0535-1/06/$20.00/©2006 IEEE

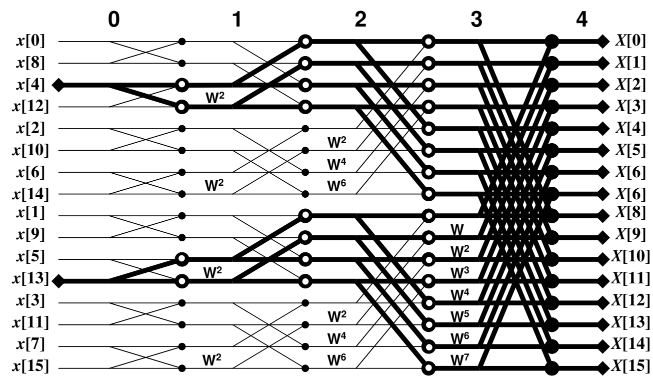

Fig. 1. Signal flow graph for an input-pruned FFT ($N = 16$).

tions. Those complexity bounds were derived assuming that the proportion of non-zero entries was known (i.e. the input vector density), but that the positions of these non-zero input bins were random. Specifically, in [10] the bounds were justified by counting the most and least possible number of occupied butterflies for a given input vector density.

In this paper, we are interested in finding a more precise characterization of the number of multiplications and additions required to compute an input-pruned FFT. Instead of bounding the number of occupied butterflies at each FFT stage, we derive the Markov process that describes the number of occupied (i.e. non-zero) paths at each FFT stage for a given input vector density.

In basing our complexity analysis on the number of occupied paths instead of the number of occupied butterflies we are able to derive a more precise estimate of the number of multiplications and additions required to compute the pruned FFT. By only examining the number of occupied butterflies, the analysis in [10] is unable to distinguish between butterflies that require full complexity (i.e. one multiplication and two additions) and butterflies that require no computation (i.e. a half filled butterfly whose single non-zero element is not on the multiplied half of the butterfly). Thus, the butterfly-based complexity analysis in [10] has slightly less resolution than our path-based analysis.

With the probability distribution of the number of non-zero paths at each FFT stage, we can determine the probability distribution of the number of multiplications and additions for an FFT with a given sparseness. From these distributions, we can gain valuable insights into the computational complexity required to compute a pruned FFT.

In the next section we will explain our procedure for deriving the Markov model that describes the number of paths in each stage. In Section III, we derive the conditional distributions for the number of additions and multiplications required at each stage given the number of paths occupied at the beginning of the stage. In Section IV, we show how the stage-wise conditional distributions can be used to determine the pdf of the total number of additions and multiplications required to compute a pruned FFT with a given input vector sparseness. Finally, we verify our results in Section V by comparing them with Monte Carlo simulations.

Notational summary:

$Q^{(k)}$	Number of occupied paths at stage k
$A^{(k)}$	Number of additions at stage k
$M^{(k)}$	Number of multiplications at stage k
$x_{i,s}^{(k)}$	Entry in row i, column s of the matrix $\boldsymbol{X}^{(k)}$

2. MARKOV MODEL FOR PATH PROPAGATION

Our approach to determining the complexity of a pruned FFT is based on modeling the number of non-zero paths at each stage of the FFT. Then, using this model, we derive the number of arithmetic operations (i.e. additions and multiplications) at each stage, and finally by adding the results of each stage together we determine the distribution of the number of operations for the entire pruned FFT. In this section we outline the procedure for determining the Markov model that describes the number of non-zero paths at each FFT stage. We will first show a few simple examples and then use inductive reasoning to extrapolate to a more general case. For purposes of simplicity, we will consider the radix-2 FFT in the remainder of the paper.

At any stage k, the number of occupied paths at the beginning of that stage, $Q^{(k)}$, can take on values from 1 to N. In order to construct the transition matrix from stage zero to stage one let us look at some particular cases. If $Q^{(0)} = 1$, then the only possibility for the number of paths at stage $k = 1$, $Q^{(1)}$, is $Q^{(1)} = 2$. To clarify, $Q^{(k)}$ describes the number of occupied paths, but does not contain any information about where these are. Since there is only one occupied path, knowing the butterfly structure of the FFT, we can easily conclude that the second stage of the FFT will contain exactly two occupied paths.

If, on the other hand, $Q^{(0)} = 2$, then there are two cases to consider. The first is when both of the non-zero entries occupy the same butterfly, which can happen in $\binom{N/2}{1}$ different ways. In this case, the values from each of the two paths will be linearly combined to produce exactly two occupied paths in the second stage, i.e. $Q^{(1)} = 2$. If, however, both non-zero entries are on different butterflies, then each will be paired with a zero path and $Q^{(1)} = 4$. There are $\binom{N}{2}$ ways to choose two paths among the N possible paths, $\binom{N/2}{2}$ way to choose the two butterflies and 2^2 ways to arrange the non-zero paths amongst the two chosen butterflies. Thus,

$$\begin{aligned}
\Pr(Q^{(1)} = 2 | Q^{(0)} = 2) &= \frac{\binom{N/2}{1}}{\binom{N}{2}} \\
\Pr(Q^{(1)} = 4 | Q^{(0)} = 2) &= \frac{\binom{N/2}{2} 2^2}{\binom{N}{2}}.
\end{aligned}$$

Notice that these are the only two possibilities when $Q^{(0)} =$

2. As a matter of fact, the number of output paths will always be at least as large as the number of input paths. Furthermore, the difference $Q^{(k)} - Q^{(k-1)}$ will either be zero or will be divisible by 2^k.

By combining these observations the probability transition matrix for the number of non-zero paths from stage $k-1$ to stage k is

$$\begin{aligned} p_{i,s}^{(k)} &= \Pr\left(Q^{(k)} = 2^k s \middle| Q^{(k-1)} = 2^{k-1} i\right) \\ &= \begin{cases} \binom{N2^{1-k}}{i}^{-1}\binom{N2^{-k}}{i-s}\binom{N2^{-k}-i+s}{2s-i}2^{2s-i}, & \Omega \\ 0, & \text{else}, \end{cases} \\ &\triangleq \boldsymbol{P}^{(k)}, \end{aligned} \tag{1}$$

where $\Omega = \{1 \le s \le 2^{-k}N,\ s \le i \le 2s,\ 1 \le k \le \log_2 N\}$ is the set of non-zero entries in the matrix $\boldsymbol{P}^{(k)}$. Notice that $\boldsymbol{P}^{(k)} \in \mathbb{R}^{N2^{1-k} \times N2^{-k}}$ is a tall stochastic matrix. The matrix is tall because the possibilities for the number of occupied paths is halved from one stage to the next.

3. CONDITIONAL NUMBER OF ADDITIONS AND MULTIPLICATIONS

From the butterfly structure, we can see that half of the paths are multiplied by a power of

$$W = e^{-j2\pi k/N}. \tag{2}$$

But when $k = 0$ no multiplication is necessary. Therefore, we only consider the multiplications with $\{W^k\}_{k=1}^{N/2-1}$ in the following complexity analysis. Also, since we assume that the positions of the non-zero input paths are uniformly distributed, then the blocks of non-zero paths at each stage are also uniformly distributed.

In order to derive the distribution of the number of multiplications conditioned on the number of input paths at the start of stage $k-1$, it is convenient to start by also conditioning on the number of paths at the start of stage k. By doing this, we can see that the number of multiplications with these conditions follows a binomial distribution. Specifically, we have the probability of the number of multiplications in stage $k-1$ conditioned on the number of paths in stage k and stage $k-1$ is

$$\begin{aligned} &\Pr\left(M^{(k-1)} = \left(\mu + i - s\right)\left(2^{k-1} - 1\right) \middle| Q^{(k)} = 2^k s,\ Q^{(k-1)} = 2^{k-1} i\right) \\ &= \begin{cases} 2^{i-2s}\binom{2s-i}{\mu}, & 1 \le s \le 2^{-k}N,\ s \le i \le 2s, \\ & 2 \le k \le \log_2 N,\ 0 \le \mu \le 2s - i \\ 0, & \text{else}. \end{cases} \end{aligned} \tag{3}$$

Next we can use the law of total probabilities to get

$$\begin{aligned} c_{i,\mu}^{(k-1)} &= \Pr\Big(M^{(k-1)} = (\mu + i - s)\left(2^{k-1} - 1\right) \Big| Q^{(k-1)} = 2^{k-1} i\Big) \\ &= \sum_{s=1}^{N2^{-k}} \Pr\Big(M^{(k-1)} = (\mu + i - s)\left(2^{k-1} - 1\right) \Big| Q^{(k)} = 2^k s,\ Q^{(k-1)} = 2^{k-1} i\Big) p_{i,s}^{(k)} \\ &= \sum_{s=1}^{N2^{-k}} \frac{\binom{N2^{-k}}{i-s}\binom{N2^{-k}-i+s}{2s-i}\binom{2s-i}{\mu}}{\binom{N2^{1-k}}{i}} \\ &\triangleq \boldsymbol{C}^{(k-1)}. \end{aligned} \tag{4}$$

Now, let us proceed to find the distribution of the number of additions. To do this we first derive the distribution of the number of additions in stage $k-1$ conditioned on the number of paths in stages $k-1$ and k. Finding this conditional distribution is straightforward because the number of additions is deterministic when the number of paths in stage $k-1$ and k are specified. Once we derive the conditional distribution we can use (1) and the law of total probabilities to find the number of additions in stage $k-1$ conditioned only on the number of paths in stage $k-1$. The conditional probability distribution on the number of additions in stage k, $A^{(k)}$, is

$$\begin{aligned} &\Pr\left(A^{(k)} = 2^k \alpha \middle| Q^{(k)} = 2^k s,\ Q^{(k-1)} = 2^{k-1} i\right) \\ &\qquad = \begin{cases} 1, & \Omega,\ \alpha = i - s \\ 0, & \text{else}, \end{cases} \end{aligned} \tag{5}$$

where $\Omega = \{1 \le s \le 2^{-k}N,\ s \le i \le 2s,\ 1 \le k \le \log_2 N\}$, as defined in (1). Now using the law of total probabilities,

$$\begin{aligned} b_{i,\alpha}^{(k)} &= \Pr\left(A^{(k)} = 2^k \alpha \middle| Q^{(k-1)} = 2^{k-1} i\right) \\ &= \sum_{s=1}^{N2^{-k}} \Pr\Big(A^{(k)} = 2^k \alpha \Big| Q^{(k)} = 2^k s,\ Q^{(k-1)} = 2^{k-1} i\Big) p_{i,s}^{(k)} \\ &= p_{i,(i-\alpha)}^{(k)} \\ &\triangleq \boldsymbol{B}^{(k)}, \end{aligned} \tag{6}$$

where $1 \le k \le \log_2 N,\ 0 \le \alpha \le 2^{-k}N,\ \max(1, 2\alpha) \le i \le N2^{-k}$. In other words, $\boldsymbol{B}^{(k)}$ is just a permuted version of $\boldsymbol{P}^{(k)}$, in (1).

4. DISTRIBUTION OF ADDITIONS AND MULTIPLICATIONS FOR A PRUNED FFT

From Markov theory, the probability transition matrix for the number of non-zero paths in stage k given $Q^{(0)}$ is

$$t_{i,s}^{(k)} = \Pr\left(Q^{(k)} = 2^k s \middle| Q^{(0)} = i\right) = \prod_{d=1}^{k} \boldsymbol{P}^{(d)}. \quad (7)$$

With the cumulative transition matrix $\boldsymbol{T}^{(k)}$ from (7) and the law of total probabilities, we can write the probability distribution for the number of multiplications conditioned on the input vector sparseness (i.e. $Q^{(0)}$) as

$$\Pr\left(M^{(k)} = (\mu + i - s)\left(2^k - 1\right) \middle| Q^{(0)} = i\right) = \boldsymbol{T}^{(k)}\boldsymbol{C}^{(k)},$$

for $1 \leq k \leq \log_2 N - 1$. Finally, to find the total number of multiplications, we have to find the distribution of $\sum_{k=1}^{\log_2 N-1} M^{(k)}$, which can be done through the row-wise multiple convolution of $\boldsymbol{T}^{(k)}\boldsymbol{M}^{(k)}$. Specifically we have

$$\Pr\left(\sum_{k=1}^{\log_2 N-1} M^{(k)} = \Psi_s \middle| Q^{(0)} = i\right) = \bigotimes_{k=1}^{\log_2 N-1} \boldsymbol{T}^{(k)}\boldsymbol{C}^{(k)} \triangleq y_{i,s}, \quad (8)$$

where $\bigotimes$ is the multiple row-wise convolution operator and Ψ_s is the sth entry of the set containing all possible values of $\sum_{k=1}^{\log_2 N-1} M^{(k)}$.

Finding the distribution of the number of additions is more difficult because the distribution of the number of additions is not Markovian. That is,

$$\Pr\left(A^{(k)} = s \middle| A^{(k-1)} = i\right) \neq \Pr\left(A^{(k)} = s \middle| A^{(k-1)} = i,\ A^{(k-2)} = d\right).$$

Nevertheless, we propose that an approximation to the distribution of the total number of additions can be made by assuming that the number of additions *is* Markovian. With this assumption we can write

$$\Pr\left(\sum_{k=1}^{\log_2 N} A^{(k)} = \Theta_s \middle| Q^{(0)} = i\right) \approx \bigotimes_{k=1}^{\log_2 N} \boldsymbol{T}^{(k)}\boldsymbol{B}^{(k)} \triangleq x_{i,s}, \quad (9)$$

where Θ_s is the sth entry of the set containing all possible values of $\sum_{k=1}^{\log_2 N} A^{(k)}$.

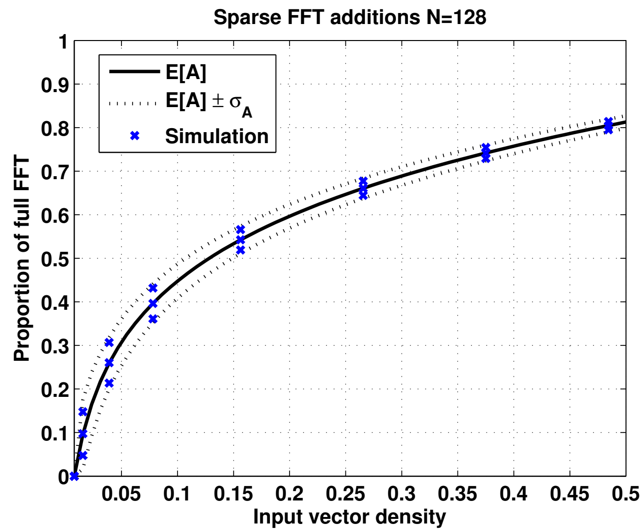

Fig. 2. Proportion of the number of additions required at a given sparseness to the number of additions required for a full FFT.

5. SIMULATION VERIFICATION

In this section we will corroborate our theoretical findings through Monte Carlo simulation results. For our simulation we choose $N = 128$ and implemented a mock pruned FFT. At each level of sparseness (or, correspondingly, each $Q^{(0)}$) we ran 30,000 Monte Carlo simulations to find the distribution of the total number of additions and multiplications. We quantified the distributions through the first and second order centered moments.

Fig. 2 is a plot of the ratio of the mean number of additions for a given input vector sparseness to the number of additions required for a full FFT. Also plotted are the lines plus and minus one standard deviation from the mean. The plot verifies that our theoretical results, even with the simplifying assumption, match very closely to the simulation results. The mean seems to be exact (which is to be expected as Markovianess is not required) while the theoretical standard deviation is slightly larger than the empirical standard deviation.

Fig. 3 is similar to Fig. 2 except that it pertains to the number of multiplications instead of the number of additions. The plot shows that the simulation results closely match our theoretical results. By comparing the two plots we observe that the number of additions grows much more slowly with the input vector sparseness than does the number of multiplications. This implies that most of the computational savings realized by implementing a custom pruned FFT is gained through reducing the number of additions and not by reducing the number of multiplications.

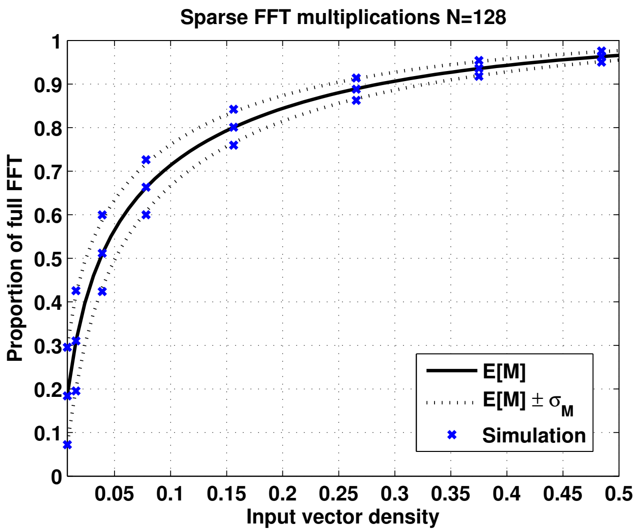

Fig. 3. Proportion of the number of multiplications required at a given sparseness to the number of multiplications required for a full FFT.

6. CONCLUSIONS

In this paper we have presented a method for determining the complexity of a pruned FFT. The proposed method is based on a Markov model of the number of occupied path at each FFT stage. This Markov path model, in conjunction with the conditional distribution of the number of multiplications and additions at each stage of the FFT, was then used to theoretically determine the total number of additions and multiplications necessary to compute a pruned FFT of a given sparseness. We verified our theoretical results through Monte Carlo simulations and showed that, despite a simplifying assumption in the derivation of the distribution of the number of additions, our results are very close to the simulated values.

7. REFERENCES

[1] J. W. Cooley and J. W. Tukey, "An algorithm for the machine calculation of Fourier series," *Math. Comput.*, vol. 19, pp. 297 – 301, 1965.

[2] J. D. Markel, "FFT pruning," *IEEE Trans. Audio Electroacoust.*, vol. 19, pp. 305 – 311, Dec. 1971.

[3] K. Nagai, "Pruning the decimation-in-time FFT algorithm with frequency shift," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 34, pp. 1008 – 1010, Aug. 1986.

[4] H. V. Sorensen and C. S. Burrus, "Efficient computation of the DFT with only a subset of input or output points," *IEEE Trans. Signal Processing*, vol. 41, pp. 1184 – 1200, Mar. 1993.

[5] T. V. Sreenivas and P. V. S. Rao, "FFT algorithm for both input and output pruning," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 27, pp. 291 – 292, Jun. 1979.

[6] C. Roche, "A split-radix partial input/output fast Fourier-transform algorithm," *IEEE Trans. Signal Processing*, vol. 40, pp. 1273 – 1276, May. 1992.

[7] S. R. Rangarajan and S. Srinivasan, "Generalized method for pruning an FFT type of transform," *IEE Proc. Vision, Image and Signal Processing*, vol. 144, pp. 189 – 192, Aug. 1997.

[8] S. Muller and J. Huber, "OFDM with reduced peak-to-average power ratio by optimum combination of partial transmit sequences," *IEE Electronics Letters*, vol. 33, no. 5, pp. 368 – 369, 1997.

[9] S. Haykin, "Cognitive radio: brain-empowered wireless communications," *IEEE Journal on Selected Areas in Communications*, vol. 23, pp. 201–220, March 2005.

[10] Z. Hu and H. Wan, "A novel generic fast Fourier transform pruning technique and complexity analysis," *IEEE Trans. Signal Processing*, vol. 53, pp. 274 – 282, Jan. 2005.