Algorithmic Transforms for Efficient Energy Scalable Computation

Amit Sinha, Alice Wang, and Anantha P. Chandrakasan
Department of EECS
Massachusetts Institute of Technology
Cambridge, MA 02139
e-mail : sinha@mit.edu, aliwang@mit.edu, anantha@mtl.mit.edu

Abstract - We introduce the notion of energy scalable computation on general purpose processors. The principle idea is to maximize computational quality for a given energy constraint. The desirable energy-quality behavior of algorithms is discussed. Subsequently the energy-quality scalability of three distinct categories of commonly used signal processing algorithms (viz. filtering, frequency domain transforms and classification) are analyzed on the StrongARM SA-1100 processor and transformations are described which obtain significant improvements in the energy-quality scalability of the algorithm.

I. Introduction

In embedded systems, energy is a precious resource and must be used efficiently. Therefore, it is highly desirable that we structure our algorithms and systems in such a fashion that computational accuracy can be traded off with energy requirement. At the heart of such transformations lies the concept of *incremental refinement* [1]. Consider the scenario where an individual is using his laptop for a video telephone application. Based on the current battery state and overall power consumption model [2] the system should be able to predict its uptime. If the battery life is insufficient, the user might choose to tradeoff some quality/performance and extend the battery life of his laptop.

Consider another scenario where a distributed sensor network [3] is being used to monitor seismic activity from a remote basestation. Sensor nodes are energy constrained and have a finite lifetime. It would be highly desirable to have energy scalable algorithms and protocols running on the sensor network. The remote basestation should have the capability to dynamically reduce energy consumption (to prolong mission lifetime if uninteresting events have occurred) by altering the throughput and computation accuracy. This type of behavior necessitates algorithmic restructuring so that every computational step leads us incrementally closer to the output.

Energy-Quality (*E-Q)* tradeoffs have been explored in the context of encryption processors [4]. A large class of algorithms, as they stand, do not render themselves to such *E-Q* scaling. Using simple modifications, the *E-Q* behavior of the algorithm can be modified such that if the available computational energy is reduced, the proportional hit in quality is minimal. However, one must ensure that the energy overhead attributed to the transform is insignificant compared to the total energy consumption. It may be possible to do a significant amount of preprocessing such that the *E-Q* behavior is close to perfect but we might end up with a situation where the overall energy consumption is higher compared to the unscalable system. This defeats the basic idea behind having a scalable system viz. overall energy efficiency.

II. Energy scalability example

Consider the simple power series shown in Equation 1. Such power series are frequently encountered in Taylor expansions used to evaluate transcendental functions.

$$y = f(x) = 1 + k_1 x + k_2 x^2 + \ldots + k_N x^N \qquad (1)$$

A standard implementation of the algorithm would have an *N*-step loop that would multiply the current value of the computed power of *x* with *x* and accumulate the result in *y*. Let us assume we have to compute *f*(2) for *N*=100. If the k_i's are similar, even after *N*-1 steps in the loop, the value accumulated in *y* would be approximately 50% off from the final value since $2^N / f(2) \approx 1/2$. In terms of *E-Q* performance, the algorithm does not do well. Assuming that the amount of energy required to evaluate *f*(2) on a processor is E_{max}, and that each step dissipates the same amount of energy (ignoring inter-instruction effects etc.), we have about 50% computational accuracy after dissipating $(N\text{-}1)/N.E_{max}$ energy. However, if we had to evaluate f(0.5), the most significant terms would occur in the first few steps in the loop and the *E-Q* behavior would be better. Based on the above analysis, we can conclude that transforming the algorithm, as shown in Table I, will result in the most significant computations occurring early in the loop as a result of which, the computational energy could be reduced, without taking a significant hit in accuracy.

Table I : Power series computation

Original Algorithm	Transformed Algorithm

`xpowi = 0.0; y = 1.0;`
`for( i=1; i<N; i++ ) {`
`  xpowi *= x;`
`  y += xpowi*k[i];`
`}`
	`if( x>1.0 ) {`
	`  xpowi = pow(x,N);`
	`  y = k[N]*xpowi+1;`
	`  for( i=N-1; i>0; i-- ) {`
	`  xpowi /= x;`
	`  y += xpowi*k[i]; }`
	`}`
	`else { // original algo`
	`}`

Fig. 1 shows the *E-Q* graphs for the original and modified power series algorithm. It captures the all the basic ideas. (i) *E-Q* behavior is in general data dependent. It is possible to come up with pathological cases where the transformed algorithm would have a *E-Q* behavior very close to the original. However, from an energy efficiency perspective, its the average *E-Q* performance that matters. (ii) It is desirable to have an *E-Q* graph above the baseline (*E=Q* on a normalized scale). This would imply that marginal returns in accuracy from successive units of computational energy is diminishing. Therefore, if the available energy is reduced by 10%, the quality degradation is less that 10%, the lesser, the better. (iii) There is an energy overhead associated with the transform which should be insignificant compared to the total energy.

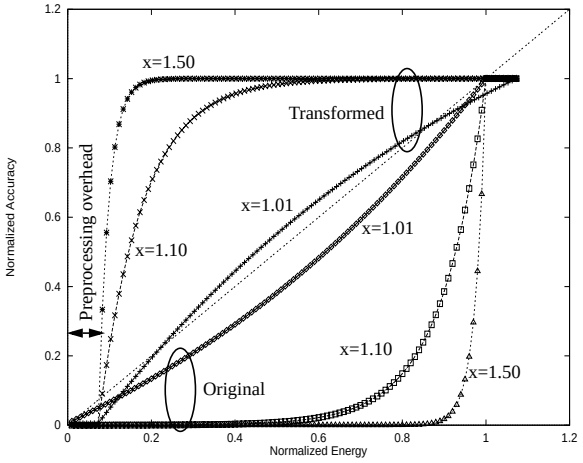

Fig. 1. *E-Q* performance of power series algorithm

III. Formal notions for scalability

We now formalize the notion of a desirable *E-Q* behavior of a system. The *E-Q* graph of an algorithm is the function $Q(E)$, representing some quality metric (e.q. mean-square error, peak signal-to-noise ratio etc.) as a function of the computational energy $0 \le E \le E_{max}$. There may exist situations where the notion of a quality metric is unclear. However, in this paper, we are dealing with signal processing algorithms where the notion of a quality metric is usually unambiguous. Consider two algorithms (I and II) that perform the same function. Ideally, from an energy perspective, II would be a more efficient scalable algorithm compared to I if

$$Q_{II}(E) > Q_I(E) \quad \forall E \tag{2}$$

In most practical cases, Equation 2 will not hold over all energy values. As shown in Table I, there might be a preprocessing overhead as a result of which the maximum energy consumptions might be different for the two cases (i.e. $E_{max,\,II} > E_{max,\,I}$). Nevertheless, as long as the Equation 2 holds over a significant range of computational energies, overall efficiency is assured.

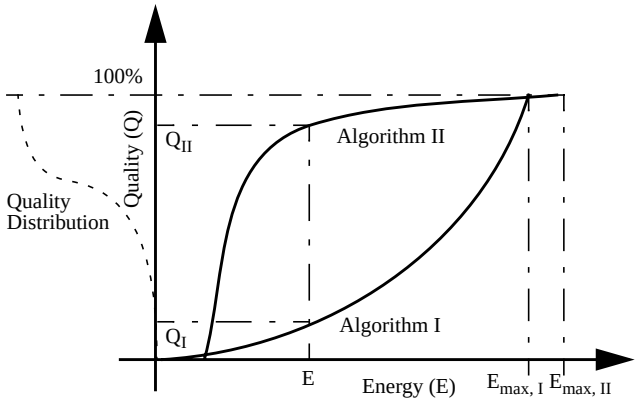

Fig. 2. *E-Q* formal notions

Let us assume that there exists a quality distribution $p_Q(x)$, i.e. from system statistics we are able to conclude that the probability that we would want a quality x is $p_Q(x)$. A typical quality distribution is shown in Fig. 2. The average energy consumption per output sample can then be expressed as

$$\bar{E} = \int p_Q(x) E(x) dx \tag{3}$$

where $E(Q)$ is the inverse of $Q(E)$. When the quality distribution is unknown, we would like the *E-Q* behavior to be maximally concave downwards (with respect to the energy axis), i.e.

$$\frac{\partial^2 Q(E)}{\partial E^2} \le 0 \tag{4}$$

The *E-Q* behavior suggested by Equation 4 is not always attainable globally i.e. across $0 \le E \le E_{max}$ as we will see subsequently. However, on an average case, for a given energy availability E, we would like the obtainable quality $Q(E)$ to be as high as possible.

IV. Energy scalable transformations

A. Filtering Application

Finite Impulse Response (FIR) filtering is one of the most commonly used Digital Signal Processing (DSP) operations. FIR filtering involves the inner product of two vectors one of which is fixed and known as the impulse response, $h[n]$, of the filter [6]. An N-tap FIR filter is defined by Equation 5.

$$y[n] = \sum_{k=0}^{N-1} x[n-k]h[k] \tag{5}$$

Various low power and energy efficient implementations of the FIR filter have been proposed and implemented [7]. The approximate processing techniques proposed in [8] reduce the total switched capacitance by dynamically varying the filter order based on signal statistics.

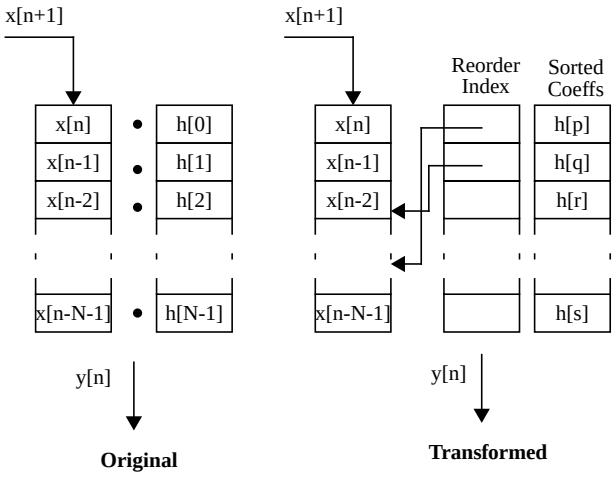

Fig. 3. FIR filtering with coefficient reordering

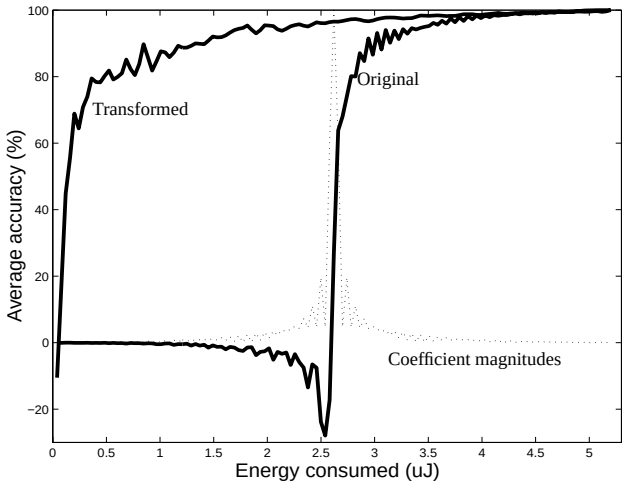

Fig. 4. *E-Q* graph for original and transformed FIR filtering

However, when we analyze the FIR filtering operation from a pure inner product perspective, it simply involves *N* multiply and accumulate (MAC) cycles. For desired *E-Q* behavior, the MAC cycles that contribute most significantly to the output $y[n]$ should be done first. Each of the partial sums, $x[k]h[n-k]$, depends on the data sample and therefore its not apparent which ones should be accumulated first. Intuitively, the partial sums that are maximum in magnitude (and can therefore affect the final result significantly) should be accumulated first. Most FIR filter coefficients have a few coefficients that are large in magnitude and progressively reduce in amplitude. Therefore, a simple but effective *most-significant-first transform* involves sorting the impulse response in decreasing order of magnitude and reordering the MACs such that the partial sum corresponding to the largest coefficient is accumulated first as shown in Fig. 3. Undoubtedly, the data sample multiplied to the coefficient might be so small as to mitigate the effect of the partial sum. Nevertheless, on an average case, the coefficient reordering by magnitude yields a better *E-Q* performance than the original scheme (See Appendix A for proof). Fig. 4 illustrates the scalability results for a low pass filtering of speech data sampled at 10kHz using a 128-tap FIR filter whose impulse response (magnitude) is also outlined. The average energy consumption per output sample (measured on the StrongARM SA-1100 [5] operating at 1.5V power supply and 206MHz frequency) in the original scheme is 5.12μJ. Since the initial coefficients are not the ones with most significant magnitudes the *E-Q* behavior is poor. Sorting the coefficients and using a level of indirection (in software that amounts to having an index array of the same size as the coefficient array), the *E-Q* behavior can be substantially improved. It can be seen that fluctuations in data can lead to deviations from the ideal behavior suggested by Equation 4, nonetheless overall concavity is still apparent. The energy overhead associated with using a level of indirection on the SA-1100 was only 0.21μJ which is about 4% of the total energy consumption. Fig. 5 shows the ratio of the energy consumed in the unsorted system to the sorted system for a given quality.

In FIR filtering, the input data samples are unknown a priori. The partial sum which is most significant is not completely deterministic until all of them have been computed. More sophisticated schemes could involve sorting both the data samples and the coefficients and using two levels of indirection to perform the correct inner product first by picking up the partial sum corresponding to the largest coefficient, then the one corresponding to the largest data sample and so on. The overhead associated with such a scheme involves real time sorting of incoming samples. Assuming that we have a presorted data array at time *n*, the next data sample $x[n+1]$ can be inserted into the right position using a binary search type technique which can be done in $O(logN)$. The scalability gains might not be substantial compared to the simpler scheme discussed before. However, in applications such as autocorrelation which involves an inner product of a data stream with a shifted version of itself, sorting both the vectors in the inner product would yield significant improvements in *E-Q* behavior.

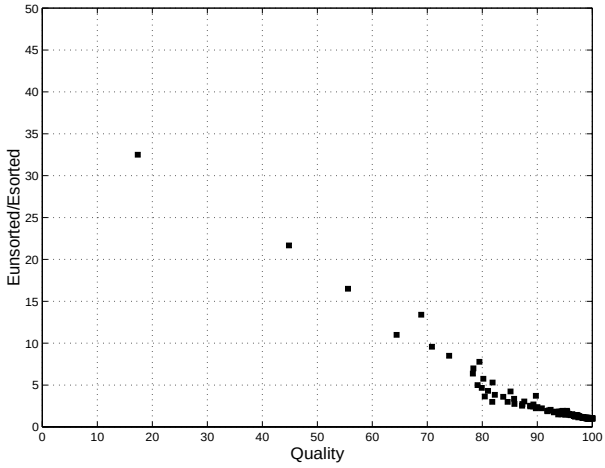

Fig. 5. Energy inefficiency of unsorted system compared to the sorted case

B. Image Decoding Application

The Discrete Cosine Transform (DCT), which involves decomposing a set of image samples into a scaled set of discrete cosine basis functions, and the Inverse Discrete Cosine Transform (IDCT), which involves reconstructing the samples from the basis functions, are crucial steps in digital video [9]. The 64-point, 2-D DCT and IDCT (used on 8x8 pixel blocks in of an image) are defined respectively as

$$X[u,v] = \frac{c[u]c[v]}{4}\sum_{i=0}^{7}\sum_{j=0}^{7} x[i,j]\cos\left(\frac{(2i+1)u\pi}{16}\right)\cos\left(\frac{(2j+1)v\pi}{16}\right) \quad (6)$$

$$x[i,j] = \frac{1}{4}\sum_{u=0}^{7}\sum_{v=0}^{7} c[u]c[v]X[u,v]\cos\left(\frac{(2i+1)u\pi}{16}\right)\cos\left(\frac{(2j+1)v\pi}{16}\right) \quad (7)$$

DCT is able to capture the spatial redundancy present in an image and the coefficients obtained are quantized and compressed. Most existing algorithms attempt to minimize the number of arithmetic operations (multiplications and additions) usually relying on the symmetry properties of the cosine basis functions (similar to the FFT algorithm) and on matrix factorizations [10]. The *E-Q* behavior of these algorithms are not good as they have been designed such that computation takes a minimal yet constant number of operations. The Forward Mapping-IDCT (FM-IDCT) algorithm, proposed in [11] can be shown to have an *E-Q* performance with is much better than other algorithms. The algorithm is formulated as follows

$$\begin{bmatrix} x_{0,0} \\ x_{0,1} \\ \vdots \\ x_{8,8} \end{bmatrix} = X_{0,0}\begin{bmatrix} c_0^{0,0} \\ c_1^{0,0} \\ \vdots \\ c_{64}^{0,0} \end{bmatrix} + X_{0,1}\begin{bmatrix} c_0^{0,1} \\ c_1^{0,1} \\ \vdots \\ c_{64}^{0,1} \end{bmatrix} + \ldots + X_{8,8}\begin{bmatrix} c_0^{8,8} \\ c_1^{8,8} \\ \vdots \\ c_{64}^{8,8} \end{bmatrix} \quad (8)$$

where $x_{i,j}$ are the reconstructed pels, $X_{i,j}$ are the input DCT coefficients, and $[c_k^{i,j}]$ is the 64x64 constant reconstruction kernel. The improved *E-Q* behavior of the FM-IDCT algorithm can be attributed to the fact that most of the signal energy is concentrated in the DC coefficient ($X_{0,0}$) and in general in the low-frequency coefficients as shown in Fig. 7. Instead of reconstructing each pixel by summing up all its frequency contributions, the algorithm incrementally accumulates the entire image based on spectral contributions from the low to high frequencies.

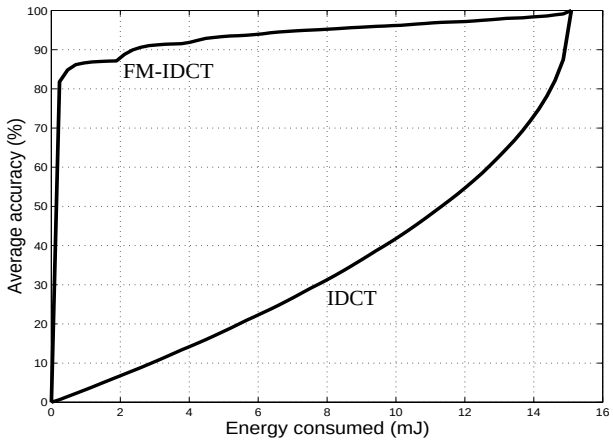

Fig. 6. *E-Q* graph for FM-IDCT vs normal IDCT

Fig. 6 and Fig. 8 illustrate the *E-Q* behavior of the FM-IDCT algorithm. It is obvious from Fig. 8 that almost 90% image quality can be obtained from as little as 25% of the total energy consumption. In terms of the overhead requirement, the only change that is required is that we now need to store the IDCT coefficients in a transposed fashion (i.e. all the low frequency components first and so on).

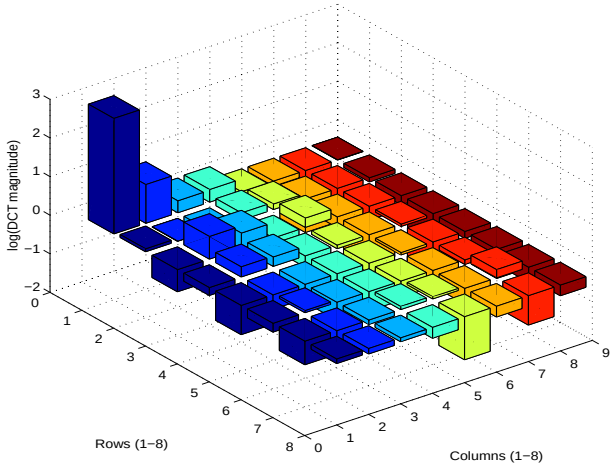

Fig. 7. 8x8 DCT coefficient magnitudes averaged over a sample image

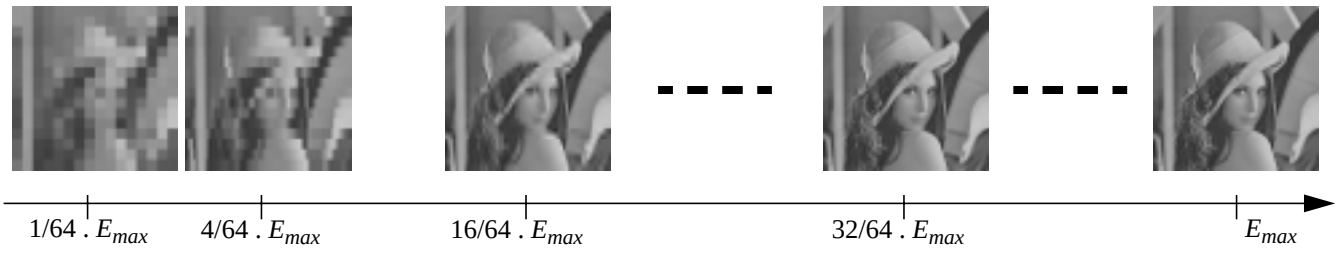

Fig. 8. Illustrating the incremental refinement property with respect to computational energy of the FM-IDCT algorithm

C. Classification using Beamforming

Beamforming algorithms can be used to aggregate highly correlated data from multiple sensors into one representative signal. The advantages of beamforming is twofold. First, beamforming is used to enhance the desired signal while interference or uncorrelated sensor noise is reduced. This leads to an improvement in detection and classification of the target. Second, beamforming reduces redundant data through compression of multiple sensor data into one signal. Fig. 9 shows a block diagram of a wireless network of M sensors utilizing beamforming for local data aggregation.

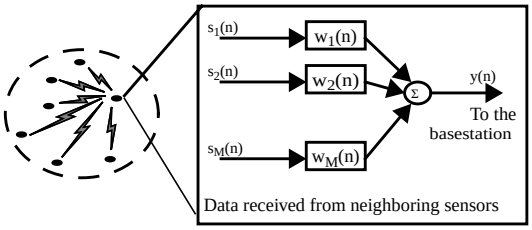

Fig. 9. Beamforming for data aggregation

We have studied various beamforming algorithms that fall under the category of "blind beamforming'' [12]. These beamformers provide suitable weighting functions, $w_i(n)$, to satisfy a given optimality criterion, without knowledge of the sensor locations. In this paper we will show energy scalability for one particular blind beamforming algorithm, the Least Mean Squares (LMS) beamforming algorithm. The LMS algorithm uses a minimum mean squared error criterion to determine the appropriate array weighting filters. This algorithm is considered an optimum algorithm, and is highly suitable for power aware wireless sensor networks [13].

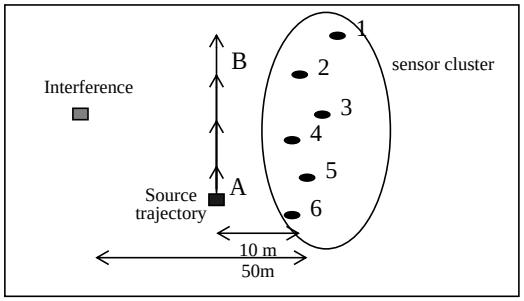

Fig. 10. Sensor testbed

We will now show how algorithmic transformations can be used to improve the *E-Q* model for LMS beamforming. Fig. 10 shows our testbed of sensors for this example. We have an array of 6 sensors spaced at approximately 10 meters, a source at a distance of 10 meters from the sensor cluster, and interference at a distance of 50 meters. We want to perform beamforming on the sensor data, measure the energy dissipated on the StrongARM SA-1100, calculate the matched filter output (quality), and provide a reliable model of the *E-Q* relationship as we vary the number of sensors in beamforming.

In Scenario 1, we will perform beamforming without any knowledge of the source location in relation to the sensors. Beamforming will be done in a pre-set order <1,2,3,4,5,6>. The parameter we will use to scale energy is n, the number of sensors in beamforming. As *n* is increased from 1 to 6, there is a proportional increase of energy. As the sensor moves from location A to B we take snapshots of the *E-Q* curve, shown in Fig. 11. This curve shows that with a preset beamforming order, there can be vastly different *E-Q* curves, which leads to a very poor energy-quality model. When the source is at location A, the beamforming quality is only at maximum when sensors 5 and 6 are beamformed. Conversely, when the source is at location B, the beamforming quality is close to maximum after beamforming 2 sensors. Therefore, for this setup, since the *E-Q* curve is highly data dependent, an accurate *E-Q* model for LMS beamforming is not possible.

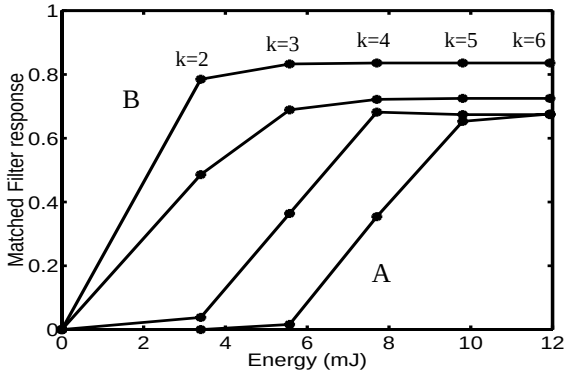

Fig. 11. *E-Q* snapshot for Scenario

An intelligent alternative is to perform some initial pre-processing of the sensor data to determine the desired beamforming order for a given set of sensor data. Intuitively, we want to beamform the data from sensors which have higher signal energy to interference energy. Using the *most-significant-first transform*, which was proposed earlier, the *E-Q* scalability of the system can be improved.. To find the desired beamforming order, first the sensor data energy is estimated. Then the sensor energies are sorted using a quicksort method. The quicksort output determines the desired beamforming order. Fig. 12 shows a block diagram of the transformed system.

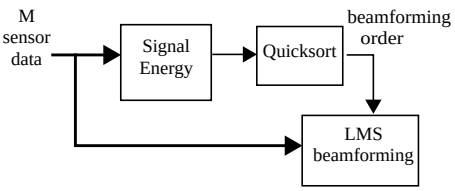

Fig. 12. "Sort by significance" preprocessing

In Scenario 2, we apply the *most-significant-first transform* to improve the *E-Q* curves for LMS beamforming. Fig. 13 shows the *E-Q* relationship as the source moves from location A to B. In this scenario, we can ensure that the *E-Q* graph be monotonically increasing, thus improving our *E-Q* models. However, there is a price to pay in computation energy. If the energy cost required to compute the correlation and quicksort was large compared to LMS beamforming, then the extra scalability is not worth the effort. However, in this case, the extra computational cost was 8.8mJ of energy and this overhead is only 0.44% of the total energy for LMS beamforming (for the 2 sensor case).

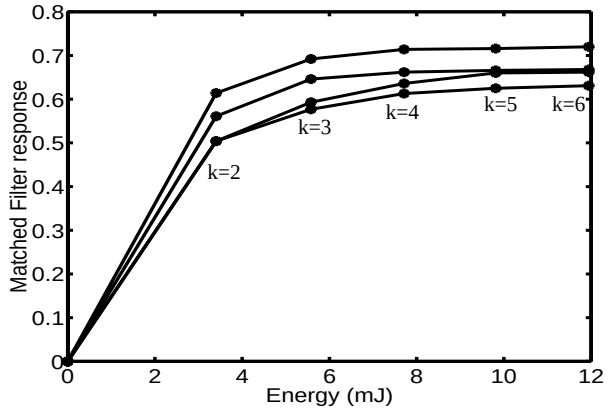

Fig. 13. *E-Q* snapshot for Scenario 2

V. CONCLUSIONS

We have introduced the notion of energy scalable computation in the context of signal processing. Algorithms that render incremental refinement of a certain quality metric such that the marginal returns from every additional unit of energy is diminishing are highly desirable in embedded applications. Using three broad classes of signal processing algorithms we have demonstrated that using simple transformations (with insignificant overhead) the Energy-Quality (*E-Q*) behavior of the algorithm can be significantly improved. In general, we have concluded that doing the most significant computations first enables computational energy reduction without significant hit in output quality.

ACKNOWLEDGEMENTS

This research is sponsored by the Defense Advanced Researc Project Agency (DARPA) and Air Force Research Laboratory, Air Force Material Command, USAF, under agreement number F30602-00-2-0551. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon[1].

APPENDIX A

The proof as to why sorting the filter coefficients would on an average produce a better *E-Q* behavior is as follows. Assume that there are N bins corresponding to our N-tap filter and as the input data streams in, the partial products corresponding to each of the taps are accumulated in the corresponding bin. Let $\{x_1, x_2, ... x_M\}$ represented the flipped and shifted data samples. For a given filter coefficient h_k, the accumulated partial products in the bin is $\sum x_i \cdot h_k$ with an expected value $h_k \sum \mathrm{E}[x_i]$ which is independent of a particular tap (assuming M is fairly large). Therefore, the magnitude of a partial product, on an average, is larger if h_k is larger and sorting the coefficients will result in the most significant partial products being accumulated first.

REFERENCES

[1] S. H. Nawab, et. al., "Approximate signal processing", Journal of VLSI Signal Processing Systems for Signal, Image, and Video Technology, vol. 15, no. 1/2, Jan. 1997, pp. 177-200

[2] A. Sinha and A. Chandrakasan, "Energy Aware Software", Proceedings of the XIII Int'l Conference on VLSI Design, Calcutta, India, Jan 2000

[3] A. Chandrakasan, et. al., "Design Considerations for Distributed Microsensor Systems", Proceedings of the IEEE 1999 Custom Integrated Circuits Conference, San Deigo, May 1999, pp. 279-286

[4] J. Goodman, A. Dancy, A. P. Chandrakasan, "An Energy/Security Scalable Encryption Processor Using an Embedded Variable Voltage DC/DC Converter", IEEE Journal of Solid-State Circuits, vol. 33, no. 11, Nov. 1998, pp. 1799-1809

[5] Advanced RISC Machines Ltd., *Advance RISC Machines Architectural Reference Manual*, Prentice Hall, New York, 1996

[6] A. V. Oppenheim and R. W. Schafer, Discrete Time Signal Processing, Prentice Hall, New Jersey, 1989

[7] A. Sinha and A. P. Chandrakasan, "Energy Efficient Filtering Using Adaptive Precision and Variable Voltage", 12th Annual IEEE ASIC Conference, Sept. 1999

[8] J. T. Ludwig, S. H. Nawab and A. Chandrakasan, "Low-Power Digital Filtering Using Approximate Processing", IEEE Journal of Solid-State Circuits, vol. 31, no. 3, March 1996, pp. 395-400

[9] N. Ahmed, T. Natarajan and K. R. Rao, "Discrete Cosine Transform", IEEE Trans. on Computers, vol. 23, Jan 1974, pp. 90-93

[10] W. H. Chen, C. H. Smith and S. C. Fralick, "A Fast Computational Algorithm for the Discrete Cosine Transform", IEEE Trans. on Communication, vol. 25, Sept 1977, pp. 1004-1009

[11] L. McMillan and L. A. Westover, "A Forward-Mapping Realization of the Inverse Discrete Cosine Transform", Proceedings of the Data Compression Conference (DCC '92), March 1992, pp. 219-228

[12] Yao, et. al., "Blind Beamforming on a Randomly Distributed Sensor Array System, IEEE Journal on Selected Areas in Communications, vol. 16, no. 8, Oct. 1998, pp. 1555-1567

[13] A. Wang, W. Heinzelman and A. Chandrakasan, "Energy-Scalable Protocols for Battery-Operated Microsensor Networks", Proceedings of SIPS '99, Oct. 1999

1. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Defense Advanced Research Project Agency (DARPA), the Air Force Research Laboratory, or the U.S. Government.