ON THE CONCEPT OF DISTRIBUTED DIGITAL SIGNAL PROCESSING IN WIRELESS SENSOR NETWORKS*

Carla F. Chiasserini
Dipartimento di Elettronica, Politecnico di Torino
Torino, Italy

Ramesh R. Rao
California Institute of Telecommunications and Information Technology
University of California, San Diego
La Jolla, California

ABSTRACT

Wireless sensors have several constraints, such as a short transmission range, poor processing capabilities, and a limited available energy. Our goal is to design an energy-efficient sensor network by exploiting the concept of collaborative signal processing. We consider a network composed of multiple sensor nodes, each of which corresponds to a processor. By using an appropriate collaborative computational algorithm and communication scheme, we make sensors operate as a Distributed Digital Signal Processor (DDSP) and generate the desired results. The benefit of such an approach is twofold: (i) the energy and computational limitations of the individual sensors can be overcome, and (ii) for a given level of processing complexity, the energy efficiency of the overall network can increase significantly. We apply the DDSP approach to the Fast Fourier Transform algorithm, and derive results showing the improvement in performance that can be obtained through the proposed technique. Moreover, we study the existing trade-off between energy saving, data latency and number of employed sensors.

INTRODUCTION

Wireless sensors can be used for remote monitoring and object-tracking in different environments and for a wide range of applications. Typically, they consist of a Micro-ElectroMechanical System (MEMS), a low-power Digital Signal Processor (DSP), a radio frequency circuit, and a battery. Due to their low-cost and low-complexity nature, sensors are characterized by several constraints, such as a short transmission range, poor computation and processing capabilities, low reliability and data transmission rates, and a limited available energy. Networks composed of multiple sensors should be designed with the aim to overcome these limitations by exploiting the synergy between distributed nodes.

Consider a network where data have to be gathered from the sensors to some distant information sink, which can be any wireless device connecting the network to a communication infrastructure. By applying data gathering techniques and routing algorithms, information can be conveyed from the remote sensors to the sink through relay nodes, regardless of the transmission range of the sensors and their distance from the sink [1]. Also, arbitrary vast areas can be covered by deploying a sufficient number of sensors, each of which will be in charge of controlling a portion of the observed area. When minimizing data transmissions is important, collaborative processing techniques, such as beamforming and data fusion, can be applied. Correlated data can be combined between nearby sensors before being transmitted to the sink node. In this way, communications are mostly restricted to data exchange between neighboring nodes while the signal quality and statistics are enhanced [2, 3]. Instead, when a high network reliability is required, redundancy in the collected data is highly desirable, and can be obtained by having multiple sensors reporting on the same target event [4].

In this paper, our objective is to provide energy efficiency in wireless sensor networks.

The major contributions to power consumption in sensor nodes are: (i) the power consumed by the digital part of the device circuitry; (ii) the power consumed by the transceiver in transmitting and receiving mode; and (iii) the output transmission power. The output transmission power of a sensor depends on its transmission range, while the total

*This work was supported by the Centro di Eccellenza per le Radio COmunicazioni Multimediali (CERCOM), Torino, Italy.

power consumption depends on the amount of transmitted, received, and processed data. Energy-efficient techniques can therefore operate in two domains: (i) signal processing and (ii) communication protocol architectures. Signal processing techniques involve energy scaling algorithms [5], design of low-power processors [6], and partitioning of algorithms among multiple sensor nodes [3]. Techniques at the protocol level include low-power network configuration algorithms [7], energy-efficient routing and medium access schemes [8, 9].

In this work, we adopt an approach which belongs to both the signal processing and the network protocol domains. We apply the concept of collaborative signal processing to implement a Distributed Digital Signal Processor (DDSP). The idea of the DDSP relies on the *divide-and-conquer* paradigm, which is often used in multiprocessor computers. According to the divide-and-conquer paradigm, a problem is divided in multiple sub-problems of smaller size. Every processor solves each subproblem by using the same algorithm, and the solution to the original problem is obtained by combining the outputs from the different processors [10]. In our scenario, each sensor corresponds to a processor. By designing appropriate communication protocols and collaborative computational schemes, we make sensors operate as a distributed DSP and generate the desired result. The benefit of such an approach is twofold: (i) the energy and computational limitations of the individual sensor nodes can be overcome; (ii) for a given level of processing complexity, the energy efficiency of the overall network can increase significantly.

We describe the DDSP concept in more detail in the next section, where we apply the DDSP approach to the Fast Fourier Transform (FFT) algorithm. Then, we present some results showing the improvement in performance that is obtained through the DDSP technique. Finally, we conclude the paper and discuss aspects that will be subject of further research.

APPLYING THE DDSP APPROACH TO THE FFT ALGORITHM

Consider a network composed of S sensors. Think of the S sensors as a set of interconnected processors, each of them having a local memory, and being able to communicate with each other by sending and receiving messages. Then, consider the Fast Fourier Transform (FFT) algorithm, which is largely used in signal processing. Our objective is to show how, in this case, processing operations can be distributed among sensors, in such a way that the energy consumption of the overall network is reduced. Also, we want to show the existing trade-off between energy gain and information latency.

As an example, below we present the case of a two-sensor network. Different schemes are introduced for computing the FFT [10], and the main issues related to distributed processing in sensor networks are highlighted in the case of this simple system scenario.

The 2-Sensor FFT Computation

Let us assume $S = 2$ and denote the two sensors by s_1 and s_2, respectively. Let r be a vector of N entries, containing the total number of data samples over which we want to compute the FFT. We focus on the decimation-in-time FFT algorithm, which has an execution time and complexity that scales as $O(N \log N)$.

The unit computational block of the FFT algorithm is called *butterfly*. In order to implement a distributed computation of the butterfly, let r be properly partitioned into two vectors of $N/2$ data samples each, denoted by u and v. We assume that u is stored at s_1 and v is stored at s_2. We indicate by w_L the column vector of weights that are needed to compute the FFT. By taking u and v as input data, the output of the 2-sensor butterfly computation is given by: $u + w_L \cdot v$ and $u - w_L \cdot v$. Observe that each sensor operates with vectors of length equal to $N/2$. Thus, the computational complexity faced by the individual node is significantly lower than in the case where all the N samples are handled by one single sensor. On the other hand, it is clear that a distributed computation requires communication between s_1 and s_2 because each of the sensors needs non-local data for performing its task.

A first approach to the 2-sensor computation is as follows.

1. s_1 sends a copy of u to s_2
2. s_2 sends a copy of v to s_1
3. s_1 computes $u + w_L \cdot v$
4. s_2 computes $u - w_L \cdot v$.

The total number of samples sent over the radio channel is equal to N, and two transmissions are performed. Note that the scheme is load balanced since each sensor has roughly the same amount of computation and communication. However, this algorithm is redundant because each sensor computes the term $w_L \cdot v$. In the following, we adopt a different version of this approach, that avoids redundancy at the expense of load balancing. The procedure is as follows.

1. s_1 sends a copy of u to s_2
2. s_2 computes $w_L \cdot v$ and sends the result to s_1
3. s_1 computes $u + w_L \cdot v$
4. s_2 computes $u - w_L \cdot v$.

Again, two transmissions are performed and a total number of N samples are exchanged between the two sensors.

We point out that an alternative approach is also possible, which yields a load balanced and redundancy-free scheme at the expense of two additional transmissions among the sensors. However, for the sake of simplicity, this approach will not be considered here.

The performance of the above schemes can be evaluated in terms of data latency by fixing:
(i) the time needed to transmit a data sample,
(ii) the time overhead due to the start up phase of the power amplifier, and
(iii) the computational time of the floating point arithmetics.
The energy consumption associated with the presented algorithms can be estimated by considering in addition the following parameters:
(iv) the geographical proximity among sensors, that determines the output transmission power of inter-sensor communications,
(v) the sensor power consumption in transmitting and receiving mode, and
(vi) the energy consumption per floating operation at the sensor.

In the next section, we fix these parameters and evaluate the performance of the computation scheme as we vary the number of sensors involved in the FFT computation, the operating frequency of the sensor's microprocessor, and the number of FFT points. Based on this study, we select the system set-up that, by trading-off between data processing at the individual sensor and inter-sensor communication, reduces the energy consumption of the whole network. Alternatively, in the case where the sensor computation must meet precise requirements in terms of data latency, we find the optimal compromise between energy gain and delay.

RESULTS

We consider a network composed of 2^{10} nodes which are randomly distributed over a $Q \times Q$ region, with $Q = 10$ m. We assume that the RF part of a sensor node dissipates 50 nJ/bit in the transmit or receive circuitry and 100 pJ/bit/m^2 for the transmit amplifier [5]. Since short distance transmissions are considered, the energy loss due to the transmission over the radio channel is assumed to scale as d^2, where d is the distance between the transmitter and the receiver. We set the start up time of the tranceiver to be equal to 100 μs [11].

S represents the number of sensors that perform the N-point FFT in a distributed manner; we set S and N to be powers of 2 and such that $N \geq S$. To apply the DDSP technique, we randomly select S sensors among all the network nodes, and divide the initial set of N samples into S

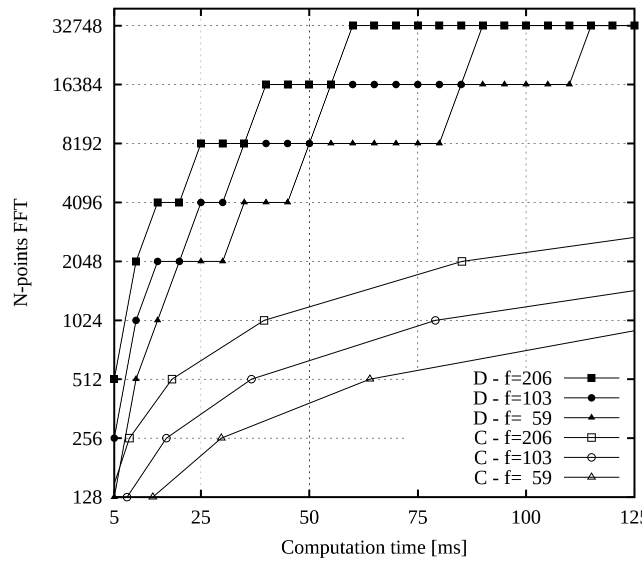

Figure 1: Number of FFT points (N) versus total computation time, for operating frequency of the microprocessor equal to f=59,103, and 206 MHz. The results obtained when the FFT is computed by a single sensor node (denoted by the label C in the plot) are compared to the results derived in the case where the computation is distributed among multiple sensors (denoted by the label D in the plot).

sets of N/S points each. The FFT is computed in $\log_2 S$ stages. Each stage includes $S/2$ butterflies that are performed as described in the previous section. Notice that, at each stage, every sensor deals with a vector of N/S samples. At the end of the computation procedure, the S sensors forward their outputs toward a central controller, where the FFT data is collected.

Each sensor is equipped with a StrongARM SA-1100 microprocessor, which is a popular low-power embedded proces-
sor. It can operate at any of the following clock frequencies: f=59,74,89,103,118,133,148,162,177,192,206 MHz [6]. By using a web-based tool, the so-called JouleTrack [6], we estimate the energy consumption and the execution time experienced by the individual sensor during the FFT computation. Clearly, the lower the operational frequency, the smaller the energy consumption and the larger the processing delay.

By summing communication and computational costs, we derive the system performance in terms of data latency and total energy expenditure as N, S, and the clock frequency vary. The results shown in the following are obtained by assuming that the packet size used for inter-sensor communications is equal to 50 bit.

Figures 1 and 2 show the N-point FFT computation time and energy consumption for different values of N

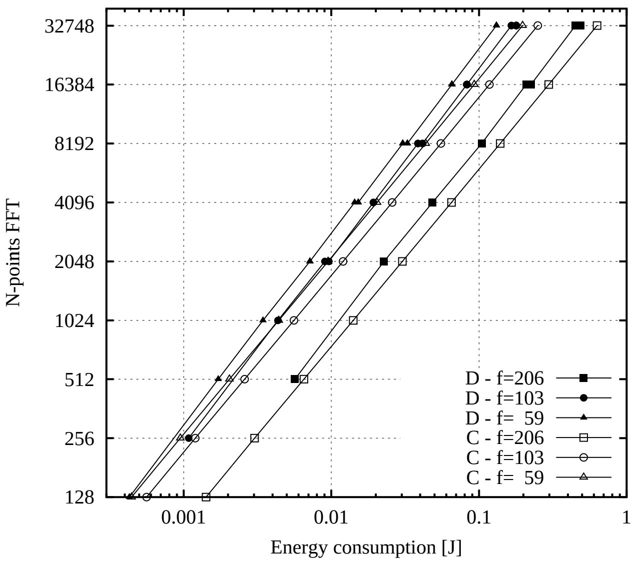

Figure 2: Number of the FFT points (N) versus the total energy consumption, for operating frequency of the microprocessor equal to f=59,103, and 206 MHz. The results obtained when the FFT is computed by a single sensor node (the curves denoted by the label C in the plot) are compared to the results derived in the case where the computation is distributed among multiple sensors (the curves denoted by the label D in the plot).

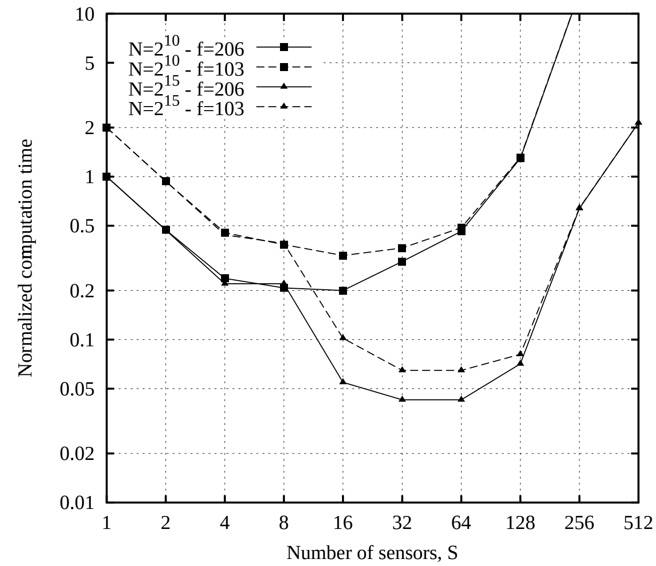

Figure 3: Total computation time normalized with respect to the case where the FFT is computed by a single sensor node and $f = 206$ MHz versus the number of sensors employed. Results are obtained for different numbers of FFT points and for operating frequency of the microprocessor equal to f=59,103, and 206 MHz.

and of the operational frequency (namely, f=59,103, and 206 MHz). The results obtained when one single sensor computes the N-point FFT (represented by the curves denoted with the label C in the plots) are compared to the results derived in the case where the computation is distributed among multiple sensors (represented by the curves denoted with the label D in the plots). In Fig. 1, the points in a curve, that are associated to the same value of N and f but to different values of computation time, correspond to different numbers of involved sensors.

Looking at Fig.s 1 and 2, we notice that a lower data latency is obtained in the case of distributed computation, regardless of which clock frequency is used for computing the distributed FFT. In fact, by applying the DDSP concept, the algorithm operations are executed in parallel at the various sensor nodes, and this may lead to a shorter computation time. Also, by fixing the clock frequency, the distributed scheme always outperforms the centralized scheme in terms of energy consumption.

Figure 3 shows the computation time of the N-point FFT as a function of the number of employed sensors. Different curves are obtained for different values of N and of the frequency f, at which the microprocessor operates. Given N and f, the computation time is normalized to the value obtained when a centralized scheme is applied (i.e., the computation is performed by one single sensor) and $f = 206$ MHz.

Looking at Fig. 3, we observe that, by fixing f, the computation time decreases as S increases until a minimum is reached. This value provides the optimal number of sensors to be used. The advantage of using the DDSP approach is more evident as N increases. Instead, for high values of S, there is not benefit in applying the DDSP concept because its communication costs outweigh the benefits in terms of computation costs. When N is fixed, a lower computation time is obtained as the clock frequency increases. Notice, however, that the benefit of using the DDSP approach with respect to the centralized scheme is more evident as f decreases. In fact, by using a distributed scheme, the maximum gain in computation time relatively to the centralized case is equal to 5 for $f = 206$ MHz, and is equal to 6.6 for $f = 103$ MHz.

In Fig. 4, we present the trade-off between computation time on one hand and energy consumption and number of employed sensors on the other hand. Energy expenditure and time delay are derived for $N = 32768$ and by varying f and S. The results are normalized to the values obtained for $S = 1$ and $f = 206$ MHz. Looking at the plot, it can be seen that, given the number of available sensors, the larger the delay, the lower the energy consumption. Conversely, by fixing the acceptable value of energy consumption, the FFT computation time can be traded-off with the number

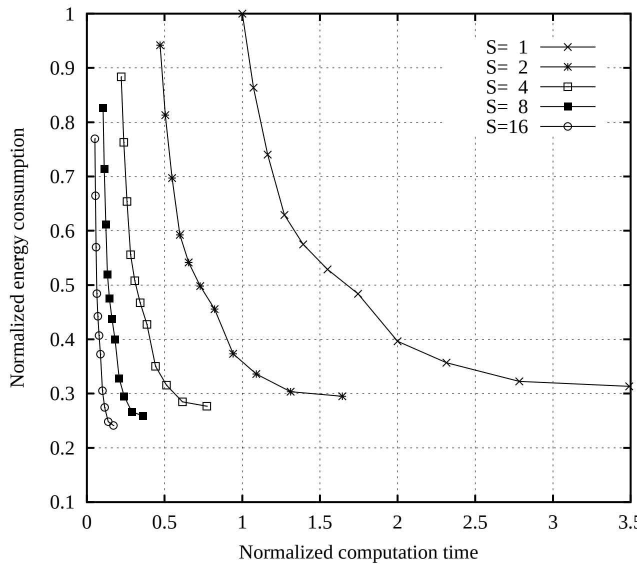

Figure 4: Trade-off between computation time and energy consumption, normalized to the values obtained when the FFT is computed by a single sensor node and the clock frequency is equal to 206 MHz. Results are shown for $N = 32768$ and for different numbers of sensor nodes (S).

of employed sensors.

CONCLUSIONS AND FUTURE WORK

In this paper, we studied a wireless sensor network whose nodes have limited energy availability and low processing capabilities. We considered each network node as a processor and extended the concept of collaborative signal processing to this scenario. By using a collaborative computational algorithm and communication scheme, we made sensors operate as a Distributed Digital Signal Processor (DDSP). We applied the DDSP approach to the Fast Fourier Transform algorithm. The results, that were presented, show the system performance for different network set-ups, and the trade-off existing between energy consumption, data latency, and number of employed sensors.

Future work will focus on the application of the DDSP concept to other signal processing algorithms that are typically used in wireless sensor applications. Moreover, general guidelines for a distributed and energy-efficient implementation of signal processing in sensor networks will be provided.

REFERENCES

[1] K. Sohrabi, J. Gao, V. Ailawadhi, and G. J. Pottie, "Protocols for Self-Organizing of a Wireless Sensor Network," *IEEE Personal Comm. Mag.,* October 2000, pp. 16–27.

[2] S. R. Blatt, "Collaborative Signal Processing for Sensor Networks," *Proc. of the SPIE - The International Society for Optical Engineering,* Vol. 4126, San Diego, CA, July 2000, pp. 183–185.

[3] A. Wang and A. P. Chandrakasan, "Energy Efficient System Partitioning for Distributed Wireless Sensor Networks," *Proc. International Conference on Acoustics, Speech, and Signal Processing (ICASSP '01),* May 2001, http://www-mtl.mit.edu/research/icsystems/uamps/.

[4] G. Hoblos, M. Staroswiecki, and A. Aitouche, "Optimal Design of Fault Tolerant Sensor Networks," *Proc. of the 2000 IEEE International Conference on Control Applications,* Anchorage, Alaska, September 2000, pp. 467–472.

[5] W. Heinzelman, A. Sinha, A. Wang, and A. P. Chandrakasan, "Energy-Scalable Algorithms and Protocols for Wireless Microsensor Networks," *Proc. International Conference on Acoustics, Speech, and Signal Processing (ICASSP '00),* June 2000, http://www-mtl.mit.edu/research/icsystems/uamps/.

[6] A. Sinha and A. P. Chandrakasan, "JouleTrack - A Web Based Tool for Software Energy Profiling," *Proc. of the 38th Design Automation Conference,* June 2001, http://www-mtl.mit.edu/research/icsystems/uamps/.

[7] M. Ajmone Marsan, C. F. Chiasserini, A. Nucci, G, Carello, and L. De Giovanni, "Optimizing the Topology of Bluetooth Wireless Personal Area Networks," *Proc. of IEEE Infocom 2002,* New York, NY, USA, June 2002.

[8] P. Bergamo, D. Maniezzo, A. Giovanardi, G. Mazzini, and M. Zorzi, "Distributed Power Control for Power-Aware Energy-Efficient Routing in Ad Hoc Networks," *Proc. of European Wireless 2002 (EW2002),* Florence, Italy, February 2002.

[9] J.-C. Chen, K. M. Sivalingam, P. Agrawal, and R. Acharya, "Scheduling Multimedia Services in a Low-Power MAC for Wireless and Mobile ATM Networks," *IEEE Transactions on Multimedia,* Vol. 1, No. 2, June 1999, pp. 187–201.

[10] E. Chu and A. George, *Inside the FFT Black Box: Serial and Parallel Fast Fourier Transform Algorithms,* CRC Press, Computational Mathematics Series, Boca Raton, 2000.

[11] S.-H. Cho and A. P. Chandrakasan, "Energy Efficient Protocols for Low Duty Cycle Wireless Microsensor Networks", ICASSP 2001, May 2001.