Power-Efficient Software Allocation in Wireless Sensor Networks

C.Brandolese, W. Fornaciari, L.Rucco, D. Zoni
Politecnico di Milano, DEI – Piazza Leonardo Da Vinci, 32 – 20133 Milano (Italy)

Abstract

While designing the applicative domain of a Wireless Sensor Network, minimizing the energy consumption is crucial to maximize its overall lifetime. We propose a model and an heuristic for determining an optimal and fast sub-optimal functional allocation, guaranteeing applicative completeness and availability, under both functional and non-functional constraints.

ILP Model & Optimization Goal

We model a WSN as composed by:

- A set of N nodes $\mathbf{N}=\{n_1, ..., n_N\}$
- A set of F functions $\mathbf{F}=\{f_1, ..., f_F\}$

Functions may *statically* resides on the node memory or *dynamically* be loaded from the base station or the cluster head, to cope with memory bounds. Nodes and functions are characterized by the parameters listed in Table 1 and Table 2.
Most real applications rely on a *hierarchical routing tree* with cluster-based topology.
We thus define:

- *Clusters* as partitions on the set of nodes
- *Tasks* as set of functions

The optimization goal can be expressed as:

$$\min q \mid q \geq \frac{E_{cons,i}}{E_i}, \forall i \in \{1, N\}$$

where:

$$E_{cons,i} = E_{sys,i} + E_{sta,i} + E_{dyn,i} + E_{route,i} + E_{oh,i}$$

$$E_{sta,i} = \sum_{j=1}^{F} E_{ex,j} \cdot \phi^{s}_{j,i} \quad \forall i \in \{1 \ldots N\}$$

$$E_{dyn,i} = \sum_{j=1}^{F} \left(E_{ex,j} + E_{ld,j}\right) \cdot \phi^{d}_{j,i} \quad \forall i \in \{1 \ldots N\}$$

$$E_{route,i} = \sum_{j=1}^{F} E_{fw,j} \cdot \phi^{d}_{j,i} \quad \forall h \in H_i, \forall i \in \{1 \ldots N\}$$

$$E_{oh,i} = \sum_{j=1}^{F} E_{rx,j} \cdot \phi^{d}_{j,i} \quad \forall n \in N_i, \forall i \in \{1 \ldots N\}$$

The contribution have the following meanings:

- $E_{sta,i}$, $E_{dyn,i}$: Energy for running the functions statically or dynamically allocated to node i.
- $E_{route,i}$: Energy for forwarding a dynamic function to node i.
- $E_{oh,i}$: Energy associated to neighboring nodes over-hearing.

The output of the problem is described by the variables $\phi^{s|d}_{j,i}$ indicating the execution frequency of each function on node i.

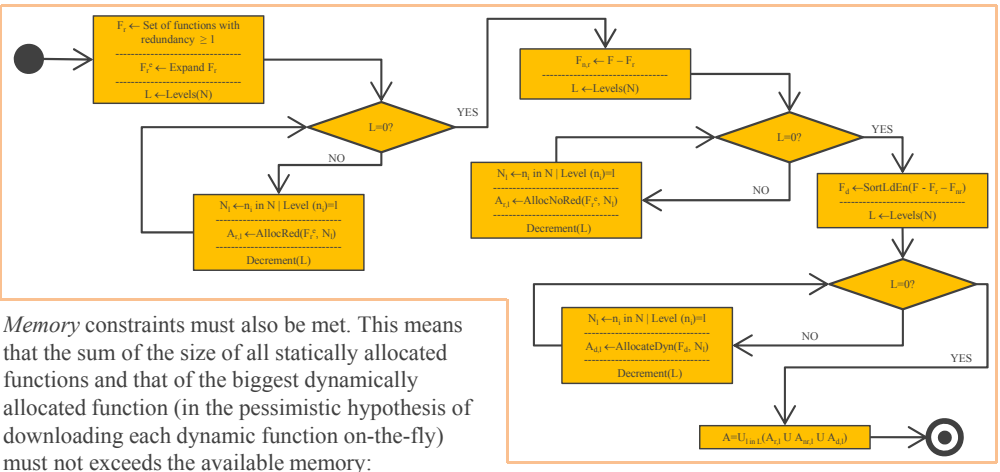
ILP Model Constraints

Several constraints need to be imposed for the model to be correct and significant.
First of all, the energy consumed by each node must not exceed the *available energy*:

$$E_{cons,i} \leq E_i$$

To implement *fault resilience* we require that some functions are statically allocated with a minimum *redundancy*:

$$\sum_{i=1}^{N} s_{j,i} \geq R_j \quad \forall j \in \{1 \ldots F\}$$



Memory constraints must also be met. This means that the sum of the size of all statically allocated functions and that of the biggest dynamically allocated function (in the pessimistic hypothesis of downloading each dynamic function on-the-fly) must not exceeds the available memory:

$$d_{k,n} \cdot S_k + \sum_{j=1}^{F} s_{j,i} \cdot S_j \leq M_i \quad \begin{cases} \forall i \in \{1 \ldots N\} \\ \forall k \in \{1 \ldots F\} \end{cases}$$

The *completeness* of the task is then required, to assure that all functions are allocated at least once, either statically or dynamically:

$$\sum_{i=1}^{N} s_{j,i} + d_{j,i} \geq 1 \quad \forall j \in \{1 \ldots F\}$$

The *correctness* of the task functionality must be guaranteed by ensuring that the sum of distributed frequencies $\phi^{s|d}$ for each function corresponds to the required frequency imposed by the designer:

$$\sum_{i=1}^{N} \left(\phi^{s}_{j,i} + \phi^{d}_{j,i}\right) = \Phi_j \quad \forall j \in \{1 \ldots F\}$$

Finally, some additional constraints are required to enforce non-negativity, frequency upper bounds, mutual exclusion between a static and a dynamic allocation of a given function, for each node:

Parameter	Meaning
M_i	Available memory
E_i	Energy capacity
$E_{sys,i}$	Energy consumed by OS and system functions
H_i	Set of successors in the routing tree
N_i	Set of neighboring nodes

Table 1: Node's parameters

Parameter	Meaning
S_i	Memory footprint
Φ_i	Execution per hour
$E_{ex,i}$	Execution energy
$E_{ld,i}$	Loading energy
$E_{tx,i}$	Transmission energy
$E_{rx,i}$	Recetion energy
R_i	Minimum required redundancy

Table 2: Function's parameters

Minimization Heuristic

Since the ILP problem often requires very long computation times, we have defined an heuristic algorithm for lifetime maximization under the constraints just described.
It is outlined by the pseudo-code of Algorithm 1 and is structured into three steps:

1. Critical functions ($R_i > 1$) first, the other next, are statically allotted to the nodes in a reverse and iterated breadth-first order on the routing three. The remaining functions are finally allocated dynamically in top-down breadth-first order,
2. Multiple instances of each function are then allocated to nodes in order to scatter the execution frequencies among more nodes.
3. Order nodes according to the estimated remaining energy and swap, when possible, allocated functions from nodes with lower remaining energy to nodes with higher. This last phase balances nodes' load.

Results

A complete and integrated optimization flow has been implemented using GNU Octave, GLPSOL and custom tools developed on purpose.
It is divided in three main phases:

1. Generation of random test instances
2. Execution of the ILP model solver and the heuristic algorithm
3. Verification of feasibility of the solution found

Test instances are generated drawing data from a real dataset, obtained in a previous work.
All the energy consumption parameters have been estimated combining the methodology in [1], with devices characterization figures found in [2].
The results obtained demonstrated very good results in *balancing energy*, since the gap between the node with more remaining energy and the one with less, is always less than 3% for the ILP model and under the 7% for the heuristic.
The heuristic, in particular, proved to be *very fast* and *accurate* if compared to the optimal solution:

- It runs 6 order of magnitude faster than ILP
- It produces results wits a relative error less than 3.3% w.r.t. the ILP solution.

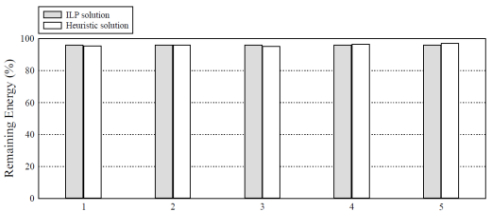


Figure 1: Solutions for a 5-nodes, 10-functions problem

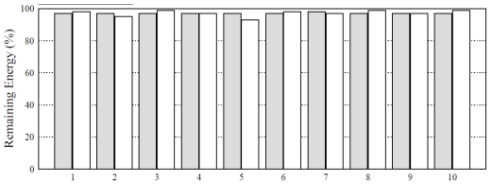


Figure 2: Solutions for a 10-nodes, 20-functions problem

References

[1] Brandolese, Corbetta, Fornaciari, "Software energy estimation based on statistical characterization of intermediate compilation code," *Proc. of the 17th IEEE/ACM International Symposium on Low-Power Electronics and Design,* ISLPED'11, Fukuoka, Japan, 2011.

[2] Polastre, Szewczyk, Culler, "Telos: Enabling ultra-low power wireless research," *Proc. of the 4th IEEE International Symposium on Information Processing in Sensor Networks,* IPSN'05, Los Angeles (CA), USA, 2005.