Dynamic Auto-Configuration in Wireless Sensor Networks

1 Introduction

Wireless sensor networks (WSN) are composed by many nodes equipped with a radio transceiver. Nodes are normally equipped with low energy processors or microcontrollers because they are battery powered. They do not have then the full capabilities and computational power of a PC connected to a Wi-Fi network.

Sensor nodes can also be positioned in places that are difficult to be accessed or can be mounted on autonomous vehicles (UAV) such as quad-copter drones or submarine drones. It is very difficult or even impossible sometimes to find an optimal static configuration that guarantees high robustness and network performance while preserving a low energy mode of operations. With a static configuration there is a high chance that a single node failure can compromise the full network operation.

Node positioning plays also an important role in determining network connectivity and energy consumption. Since it's very difficult to predict radio propagation in environments where sensor nodes are typically used, it is very important to correctly select which node just acts as a sensor node and which other node also acts as data aggregator and forwarding node. One solution that emerged along the years is Mesh networking. In mesh networks, each node acts as both data source and router. When a mesh node receives a message, if it is not the designated receiver, it broadcasts it to all its visible neighbours. While this solution is very robust towards nodes failures, it is definitely not energy efficient and suffers from high latency and low throughput since the radio channel is constantly flooded with transmissions, most of which are basically redundant.

An alternative to mesh networks is "traditional" Wi-Fi. In recent years many low power embedded wi-fi nodes have appeared on the market from different manufacturers like Texas Instruments, Silicon Labs, Analog devices, etc… All these W-Fi modules share the fact that they are extremely small and low power at the cost of implementing only a reduced set of capabilities if compared to a "PC-grade" Wi-Fi adapter.

Embedded Wi-Fi modules can only operate in a star network configuration and they can be configured as stations (STA) or as access points (AP) (or both at the same time). When in AP mode, nodes consume more power than when they operate in station mode only because they need to constantly listen and reroute packets from their associated stations. Due to their limited capabilities, they can only accommodate a limited amount of connections (i.e. the ESP8266 from Espressif can only manage five stations connected to it; Wi-Fi modules from Texas Instruments can have up to sixteen stations connected to them). It is therefore necessary to establish how many access points are needed to have full network connectivity while minimizing energy consumption.

The goal of this study is to determine a multi agent system architecture that autonomously decides the optimal network configuration. The idea is that once the nodes are deployed, they communicate and coordinate with each other in order to collectively decide which one will operate as AP and which one will operate as STA. The main objective is to simplify network deployment, increase robustness and guarantee a high degree of flexibility while using as little power as possible.

2 Multi agent architecture

The performance of two possible architectures are evaluated in this study: Reactive and Belief-Desires-Intentions (BDI). Once the strength and weakness of both architectures are evaluated, a hybrid configuration can be proposed which encompasses the strongest points of both while trying to compensate for their weak points.

Reactive architectures act as finite state machines. The output of the reactive loop depends on both the input and the internal state of the agent. While this architecture is potentially very simple, it is not very flexible and does not cope very well with unexpected behavior. The designer of reactive agents must have a very detailed knowledge of the domain of operation and keep the environment under control. More flexibility comes with a high price in complexity of the agent. The number of states of the internal agent grows exponentially with the number of inputs making this architecture unusable for very complex environments.

BDI architectures are based on practical reasoning. A BDI agent works towards its goals (desires) putting in practice a plan as a combination of partial plans (intentions) depending on its knowledge and inputs (beliefs). If the agent stops believing that a plan is feasible, it can change plan. The agent can sometimes also change goal if necessary. This means that instead of acting constantly in the same way it needs to periodically stop to analyze its beliefs versus its intentions and verify that the current plan in act is still valid. A compromise between the time spent deliberating and the time spent acting must be found. If too much time is spent deliberating, the probability that the environment changes in the meantime, thus invalidating the current plan, increases.

To summarize:

- reactive agents are fast, but not flexible.
- BDI agents are flexible, but slow.

Hybrid agents combine a reactive component with a BDI component to have the best of both worlds. The reactive component takes care of simple tasks (like responding to a ping packet) and the BDI component works towards the end goal performing more complicate coordination tasks.

3 Energy consumption

The AP typically consumes more energy than STA nodes because it has to act as router for the messages it receives and as gateway towards external networks.

3.1 Received Signal Strength (RSS) and hidden nodes problem

Depending on their location, speed and on external interference the nodes receive signal from each other with different strength. If the signal strength is too low, the link between two radio stations is open. The idea is to select the node that guarantees the best reception to all the nodes. To have optimal communication, it is possible that more than one AP node must be selected. In case for example of hidden nodes, they can select one of their neighbours as a local access point that bridges the gap between the hidden subnetwork and the main AP.

3.2 Self-Recovery

AP nodes can disappear for different reasons like empty battery, accidental damage or because they move away (vehicular networks). In case the AP nodes disappear, the nodes should select a new configuration as quickly as possible to minimize the communication disruption.

3.3 Traffic congestion

The sensor nodes have very limited processing power; to guarantee a good throughput and avoid congestion within the network, it is possible that more than one AP node is required. Having an access point routing all the traffic also affects its energy consumption in contrast to the fairness statement.

3.4 Assumptions

- The network has a star topology, with the access point as central node of the star.
- All the nodes have the same capability and can become stations, access point or both at the same time.
- No external technology (LoRa, Bluetooth, etc.) used for communication between the nodes.
- The nodes have limited capabilities and processing power compared to a PC with a Wi-Fi card.
- All the nodes can communicate at least with one other node belonging to the network.
- Self-containment / no gateway: for the purpose of this study the constraints given by an external network connection are not taken into account

4 Research questions

- What are the best agent architectures in case of moving nodes vs static nodes configuration?
- How do the nodes coordinate to avoid conflicts? (e.g. same SSID name for more than one AP)
- How much information (and which kind of information) do the nodes need to coordinate their behaviour?
- Can the coordination be done only with local communication or every node needs to know the full state of the network?
- Can the network adapt locally if the environment influence only affect part of it?