Energy harvesting IoT

Mobile and Cyber-Physical Systems

Stefano Chessa

Learning objectives

Energy harvesting IoT architectures

Properties of the energy sources, battery and load

Energy neutrality

Motivations (I)

Finite battery capacity:

- Finite lifetime of the device (and thus of its applications)
- Large batteries to provide longer lifetimes but they increase size, weight, cost and complexity to regularly replace the batteries
- Low-power design (with a low-power processor and radio) extends the device lifetime, but it implies reduced computation ability and limited communications (smaller transmission ranges, low throughput,…)

Motivations (II)

In general, finite battery capacity calls for a **tradeoff between performance and lifetime** of IoT devices

In alternative:

Energy Harvesting

Energy Harvesting

Energy harvesting converts energy from one form to another.

Energy from external sources can be harvested to power the devices and to increase their lifetime and capability.

If the harvested energy source is large and periodically / continuously available, a device can last «forever».

Allows to tune the device parameters for a better performance

Definitions (I)

- **Energy source**: source of energy to be harvested
  - Sun, wind, etc.
- **Harvesting source**: any available harvesting technology, (solar cells, wind turbines, piezo-electric harvesters etc…) that extracts energy from the environment.
- However…
  - the energy production varies with time and on environmental conditions
  - this is typically out of the control of the designer

Definitions (II)

- **Load**: consumption of energy in a device due to its activities
  - A device has many subsystems (processor/memory, radio, storage, transducers/ADC), each with its own states and consumption
  - The load varies over time:
    depends on the specific activities the device is executing at a time
- **Harvesting system:**
  - a system that supports a variable load from a variable energy-harvesting source
  - … also when the instantaneous power supply levels from the harvesting source do not match the load.

Matching energy supply and load

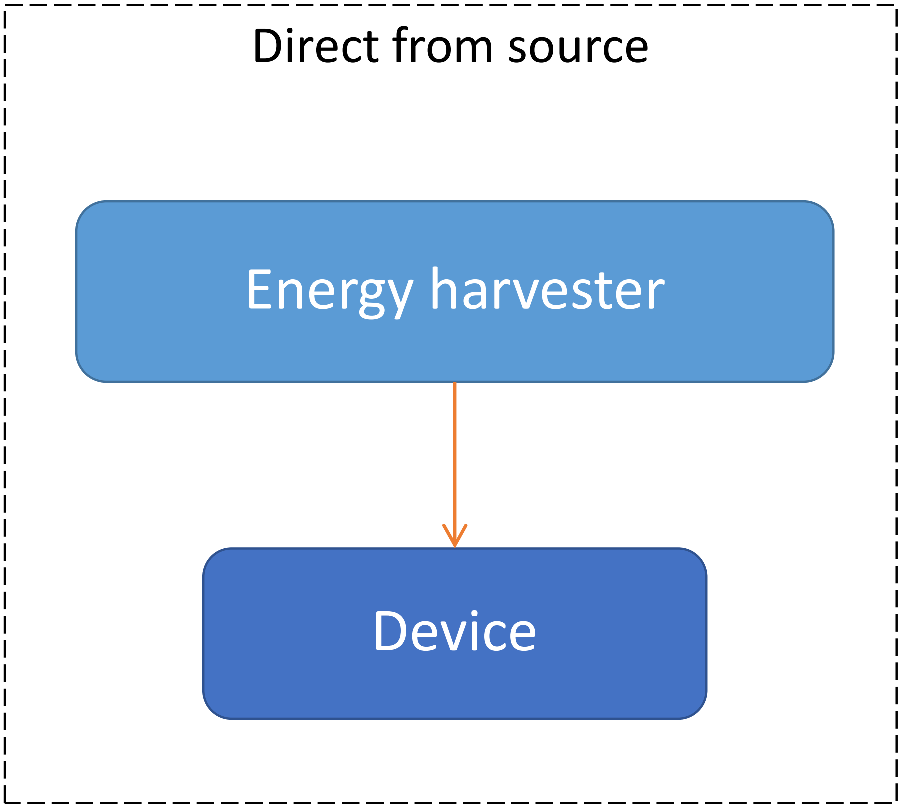
- Two main approaches:
  - **adapt the load** to the actual energy supply
  - use an **energy buffer**
    - i.e. a rechargeable battery or a supercapacitor

- In practice, neither of these approaches may be sufficient:
  - The load cannot be reduced arbitrarily
  - The buffer is not ideal (no infinite storage, energy leaks)

Harvesting architecture

Harvest-Use:

- Energy is harvested just-in-time for use
- Power output has to be above the minimum operating point (else the device turns OFF)
- Abrupt variations may cause the device to oscillate between ON and OFF states.



Modeling harvest-use systems

No energy buffers to store energy:

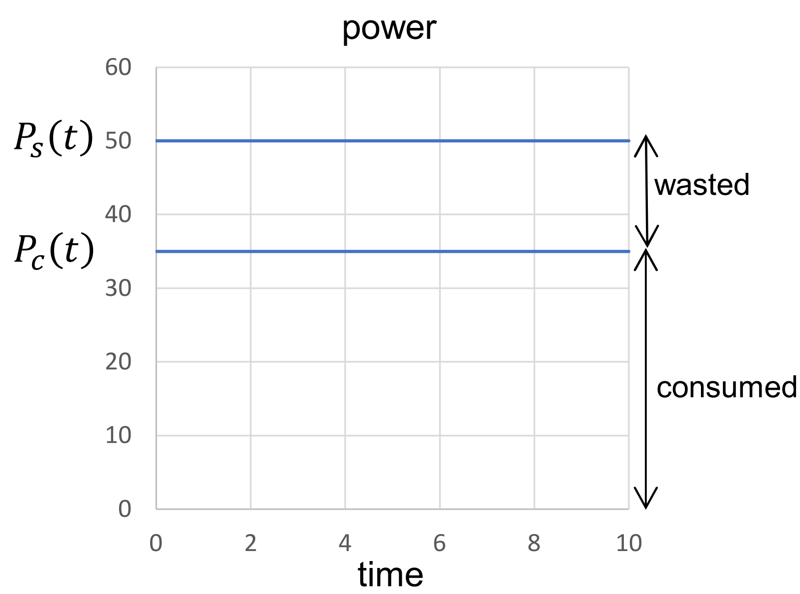
- the power produced is immediately consumed (e.g. watermills, RFID, …)

The device can operate at any time t when:

$$P_s(t) \geq P_c(t)$$

Waste of energy:

- whenever $P_s(t) < P_c(t)$: device off, energy insufficient
- whenever $P_s(t) > P_c(t)$: $P_s(t) - P_c(t)$ is in excess



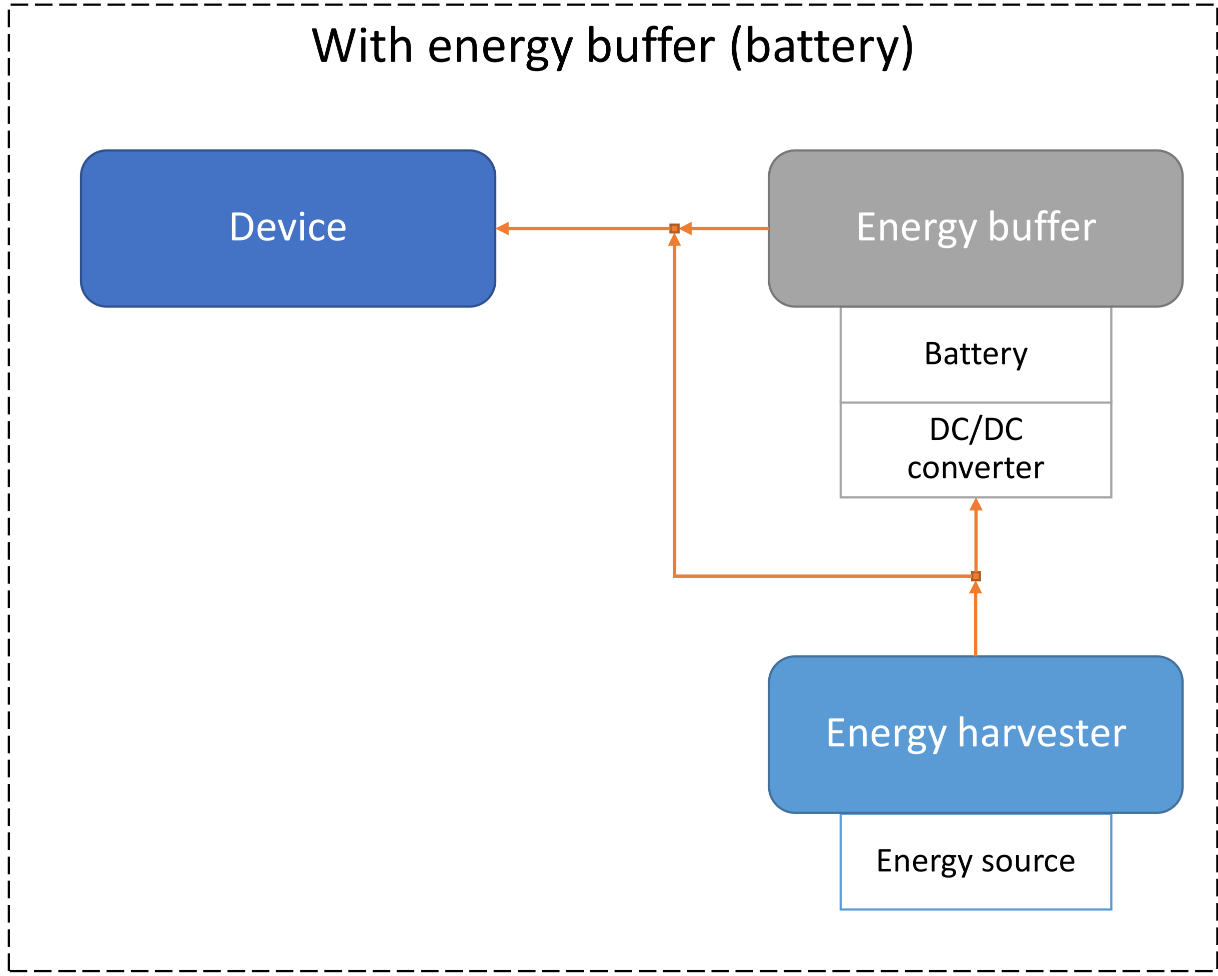
$P_s(t)$: power harvested at time t

$P_c(t)$: power consumption on the load at time t

Harvesting architecture

Harvest-Store-Use:

- Energy is harvested whenever possible and stored for future use.
- Residual energy is stored and it is used later when either there are no harvesting opportunities or the device tasks require more energy



Modeling harvest-store-use
(ideal energy buffer)

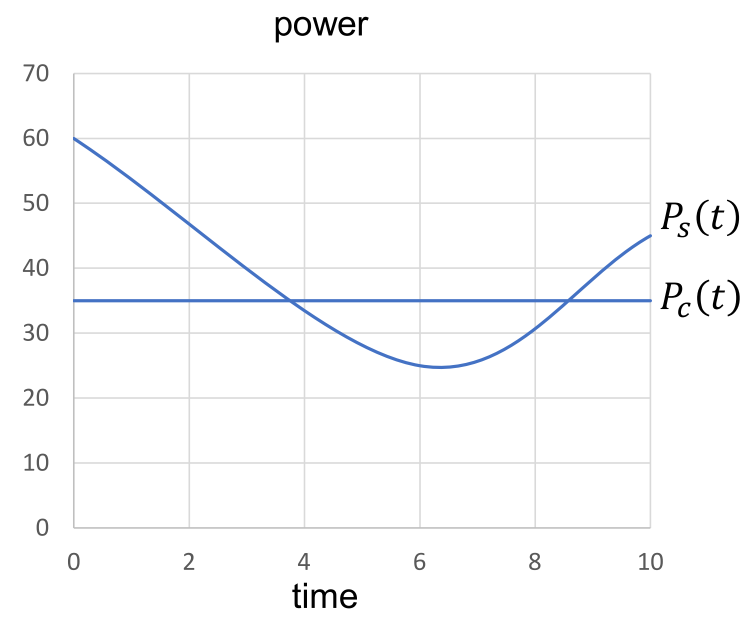
No longer energy waste (… really?)

Ideal energy buffer:

- can store any amount of energy
- does not leak energy over time
- has a charging efficiency $\eta = 1$

The device can operate at any time if, for every time interval $(0, T]$:

$$\int_0^T P_c(t)dt \leq \int_0^T P_s(t)dt + B_0 \quad \forall\, T \in (0, \infty]$$

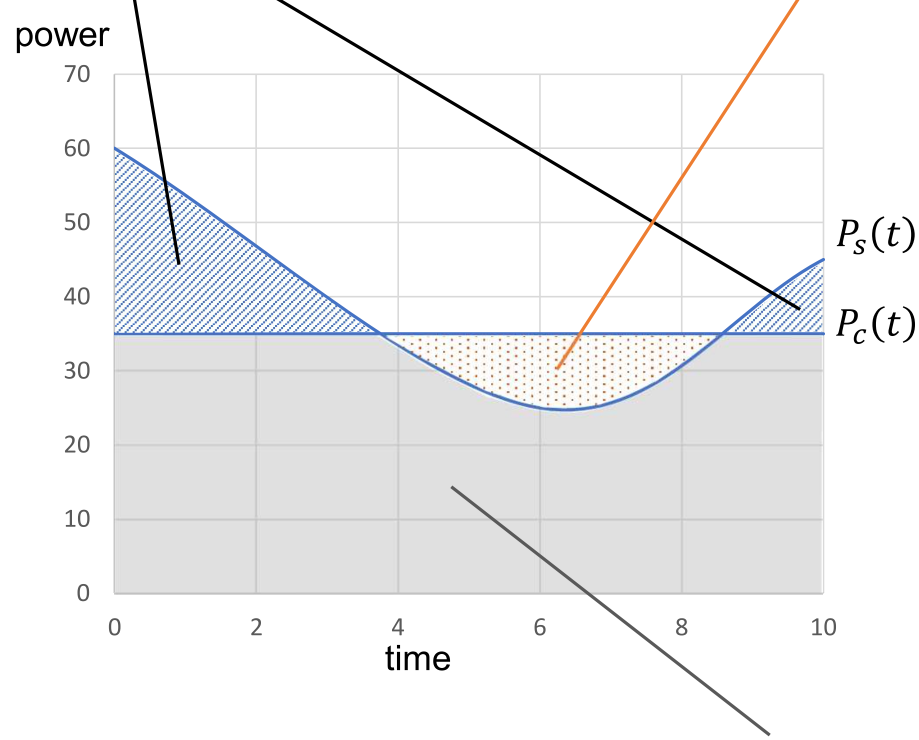


$P_s(t)$: power harvested at time t
$P_c(t)$: power consumption on the load at time t
B_0 : initial charge of the ideal buffer

Battery: power stored and power consumed

$$\int_0^T [P_s(t) - P_c(t)]^+ dt$$

$$\int_0^T [P_c(t) - P_s(t)]^+ dt$$



Power that is produced and immediately consumed

$$* \; x^+ = \begin{cases} x & x > 0 \\ 0 & x \leq 0 \end{cases}$$

harvest-store-use with non-ideal energy buffer

Non-ideal buffers:

- Have limited storage capacity
- Have a charging efficiency $\eta < 1$
- Leak energy

Symbols

- B_{max} : maximum battery capacity (energy buffer size)
- B_t : battery charge at time t
- $P_{leak}(t)$ be the leakage power of the battery at time t
- $\eta < 1$ be the charging efficiency of the battery

- $P_s(t)$: power harvested at time t
- $P_c(t)$: load at time t
- B_0 : initial charge of the buffer

- $x^+ = \begin{cases} x & x > 0 \\ 0 & x \leq 0 \end{cases}$ is a «rectifier function»

harvest-store-use with non-ideal energy buffer

The energy conservation leads to (Eq. 1):

$$B_T = B_0 + \eta \int_0^T [P_s(t) - P_c(t)]^+ dt +$$
$$- \int_0^T [P_c(t) - P_s(t)]^+ dt - \int_0^T P_{leak}(t) dt \geq 0$$

$$\forall T \in (0, \infty]$$

where:

- The first integral accounts for the energy produced in excess (thus not consumed) that goes to buffer
- The second accounts for the energy consumed from the buffer (when the production was insufficient)
- The last integral accounts for buffer energy leak

harvest-store-use with non-ideal energy buffer

Buffer is limited in charge leads to (Eq. 2):

$$B_{max} \geq B_0 + \eta \int_0^T [P_s(t) - P_c(t)]^+ dt +$$
$$- \int_0^T [P_c(t) - P_s(t)]^+ dt - \int_0^T P_{leak}(t) dt$$

$$\forall T \in (0, \infty]$$

Note that:

- Equation 1 (energy conservation) is necessary and sufficient to guarantee energy conservation
- Equation 2 (finite buffer capacity) is sufficient to guarantee thet buffer capacity is not exceeded, but it is not necessary
  - It is necessary if waste is not possible. However a device may also waste the eccess of production

Question

Consider an harvest-store-use device with a non-ideal energy buffer:

- In the interval $[0, 10sec]$ the energy production is constant and produces $P_s(t) = 80\ mA$
- In the interval $[0, 4sec]$ the load is $P_c(t) = 150\ mA$
- In the interval $[4sec, 10sec]$ the load is $P_c(t) = 20\ mA$

Furthermore:

- The charging efficiency is $\eta = 95\%$
- The battery charge at time 0 is $400\ mAh$
- The energy leak of the battery is negligible

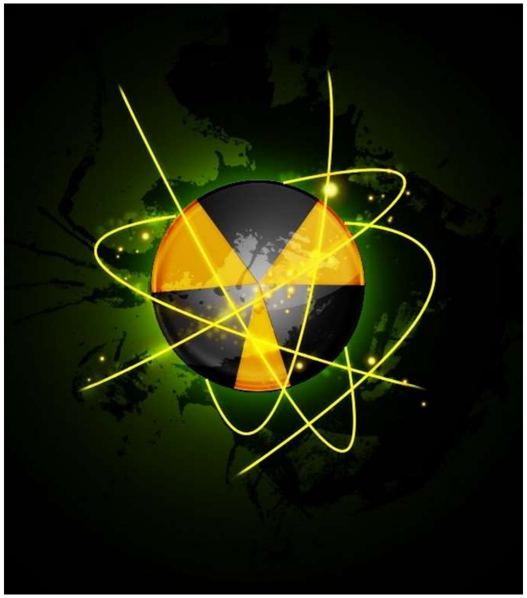
Compute the battery charge at times $4sec$ and $10sec$

Solution

Energy sources and storage technologies

Energy sources

- Sun
- Wind
- Vibrations
- Thermal
- Radioactivity
- Motion
- Radio frequency emitters
- …



Energy sources classification

Controllability:

- **Fully controllable** energy sources can provide harvestable energy whenever required.
  - Example self-power flashlights: the user may shake to generate energy whenever needed
- **Partially controllable** energy sources may be influenced by the system design, but the result may not be deterministic
  - Example: RF energy source installed in a room and RFID’s may extract energy from it. However, the energy produced at a node depends on RF propagation in the room that cannot be fully controlled
- **Non-controllable** energy sources cannot be activated on demand
  - In these cases the energy must be harvested whenever available.
  - Example: wind, sun, etc…

Energy sources classification

In turn, uncontrolled energy sources can be classified according to their **Predictability**:

- **Uncontrolled but predictable** energy sources are those for which there exist reliable models that forecast the energy availability
  - Example: Sun cannot be controlled but it can be predicted (to some extent)
  - Day/night production, summer/winter, weather forecasts

- **Uncontrolled and unpredictable** energy sources are those for which there not exist reliable models that forecast the energy availability
  - Example: vibration due to sources for which there are no models available (earthquakes etc…)

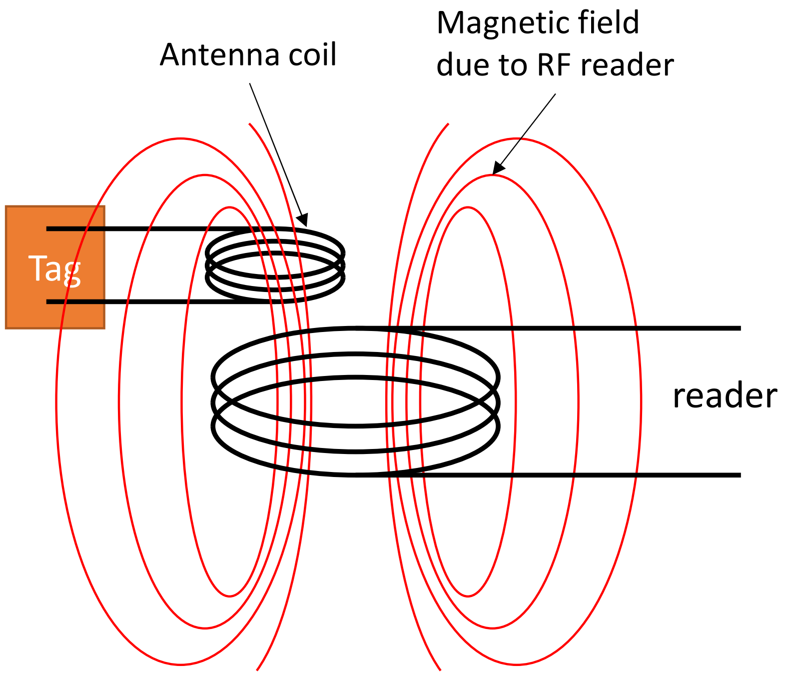
Main features of energy sources

- Solar Panels: generated energy proportional to size/area and intensity of incident light (Harvest-Store-Use).
- Wind Rotors and Turbines convert circular motion into electrical energy (Harvest-Store-Use).
- Piezo-electric films and ceramics deform and generate electricity. Larger the size, larger the electricity generated. Push buttons (Harvest Use).
- No single energy source is ideal for all applications.

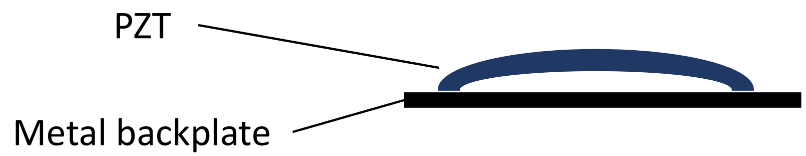
Energy source	Characteristics	Amount of energy available	Harvesting technology	Conversion efficiency	Energy harvested
Solar	Ambient, uncontrollable, predictable	100 mW/cm^2	Solar cells	15%	15 mW/cm^2
Wind	Ambient, uncontrollable, predictable		Anemometer		1200 mWh/day
Vibrations (industrial)	Controllable		Piezoelectric, induction		100 μW/cm^2
Motion (human)	Controllable	19 mW – 67 W	Piezoelectric	7.5% – 11%	2.1 mW – 5 W
Thermal (human)	Uncontrollable, predictable	20 mW/cm^2	Thermocouple		30 μW/cm^2
Thermal (industrial)	Controllable	100 mW/cm^2	Thermocouple		1-10 mW/cm^2
Radio frequency (base station)	Controllable	0.3 μW/cm^2			0.1 μW/cm^2
radioactivity		60 mW/cm^3	Radioactive decay		

Harvesting sources – Radio Frequency (RF)

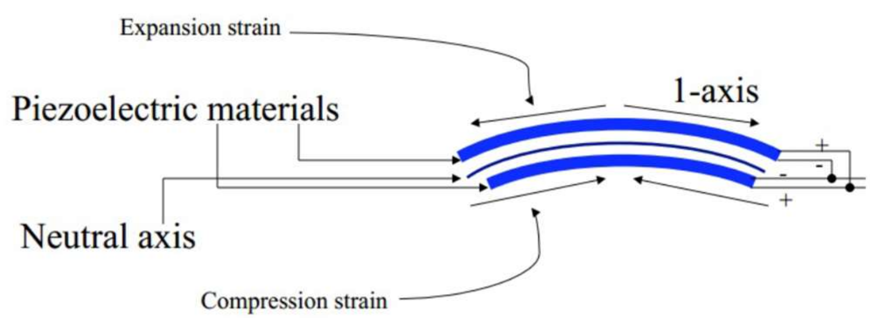
- When electromagnetic radio frequency (RF) field passes through an antenna coil, an AC voltage is generated across the coil.
- A passive RF tag powers itself by using RF energy transmitted to it (active RF tags have own battery).
- Example (RFID used to identify, locate and track people, assets and animals)
  - RFID reader queries an RFID tag by sending RF signal
  - RFID tag is entirely powered by the energy harvested by the antenna coil. This energy is just sufficient to send back a reply.



Harvesting sources – Piezo-electric

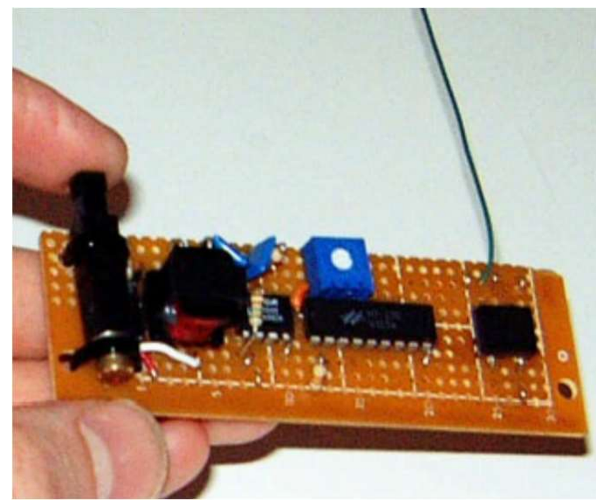


- Use mechanical force to deform a piezo-electric material, which results in an electric potential difference.
  - Piezo-electric films: PVDF (PolyVinylidene Fluoride)
  - Piezo-electric ceramic: PZT (Lead Zirconate Titanate)
  - PVDF is more flexible than PZT



Harvesting sources – Piezo-electric

- Shoe-powered RF Tag System
  - Active RFID tag wireless transmitter that sends a 12-bit identification code over short distances, while the bearer walks.
- Push-Button Controller
  - Can wirelessly transmit a digital code to a distance of 15 meters on a single button push.



Harvesting sources – Wind turbines

- Current wind turbines:
  - Many different «form-factors»
  - Many scales, even very small
  - Different operative ranges of wind speed

- For IoT applications:
  - Small size
  - Low height
  - Operative range with weak winds

Harvesting sources – Wind turbines

Specifications of a micro wind turbine generator

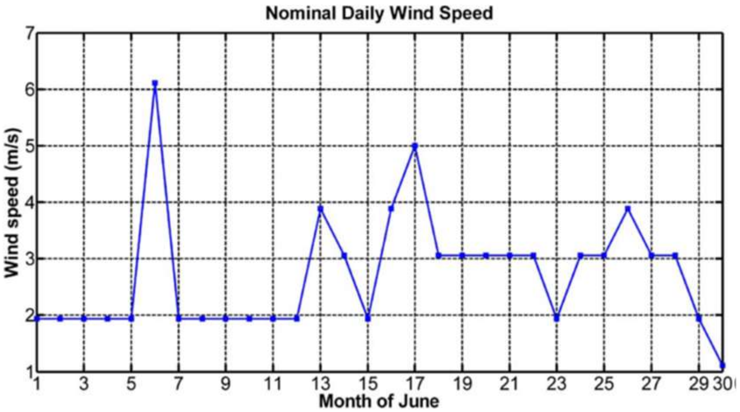
Yen Kheng Tan, Sanjib Kumar Panda, “Optimized Wind Energy Harvesting System Using Resistance Emulator and Active Rectifier for Wireless Sensor Nodes”, IEEE TRANSACTIONS ON POWER ELECTRONICS, VOL. 26, NO. 1, JANUARY 2011

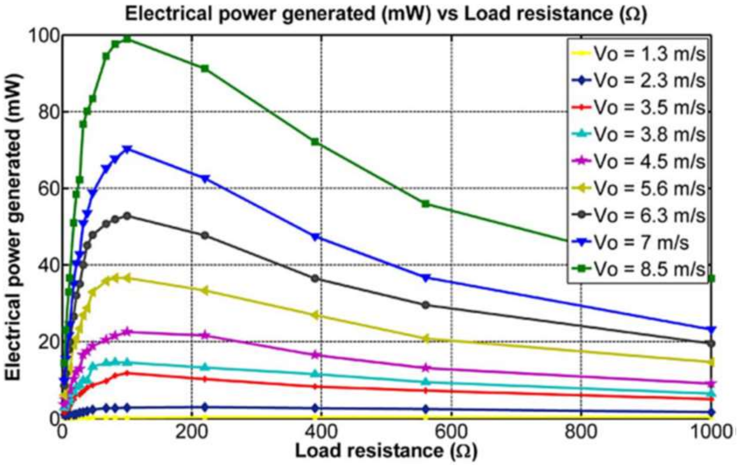
Geometrical parameters	unit	value
Radius of wind turbine	cm	3
Volume of AC generator	cm^3	1
Mass of wind turbine generator	g	100

Mechanical & electrical parameters	unit	value
Maximum wind speed	m/s	10
Rated generator speed	Rpm	3000
Rated generator frequency	Hz	50
Rated electrical power	mW	40
Open-circuit voltage	V	4
Internal resistance	Ω	100

Wind turbines: example

- Measured wind speed
- Electrical power generated for different wind speed and load applied
  - Maximum power generated when the load resistance maches the internal resistance of the generator

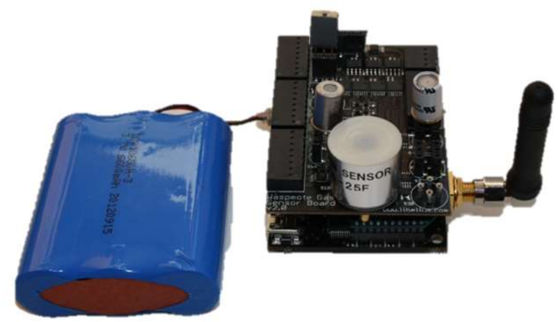


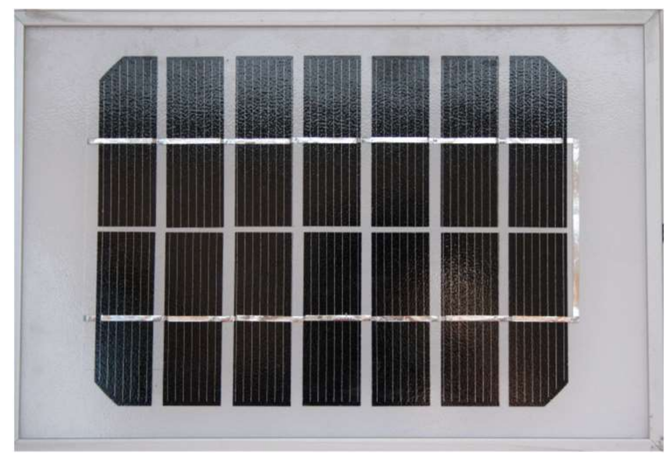


Harvesting sources:
solar energy

Example: KL-SUN3W

- Peak Power (Pout) 3W
- Maximum Power Voltage (Vmp) 5.82V
- Maximum Power Current (Imp) 0.52A
- Open Circuit Voltage (Voc) 7.38V
- Short Circuit Current (Isc) 0.55V
- Size (in mm) 225 × 155 × 17
- Operating Temperature: -40 to +85 ◦C



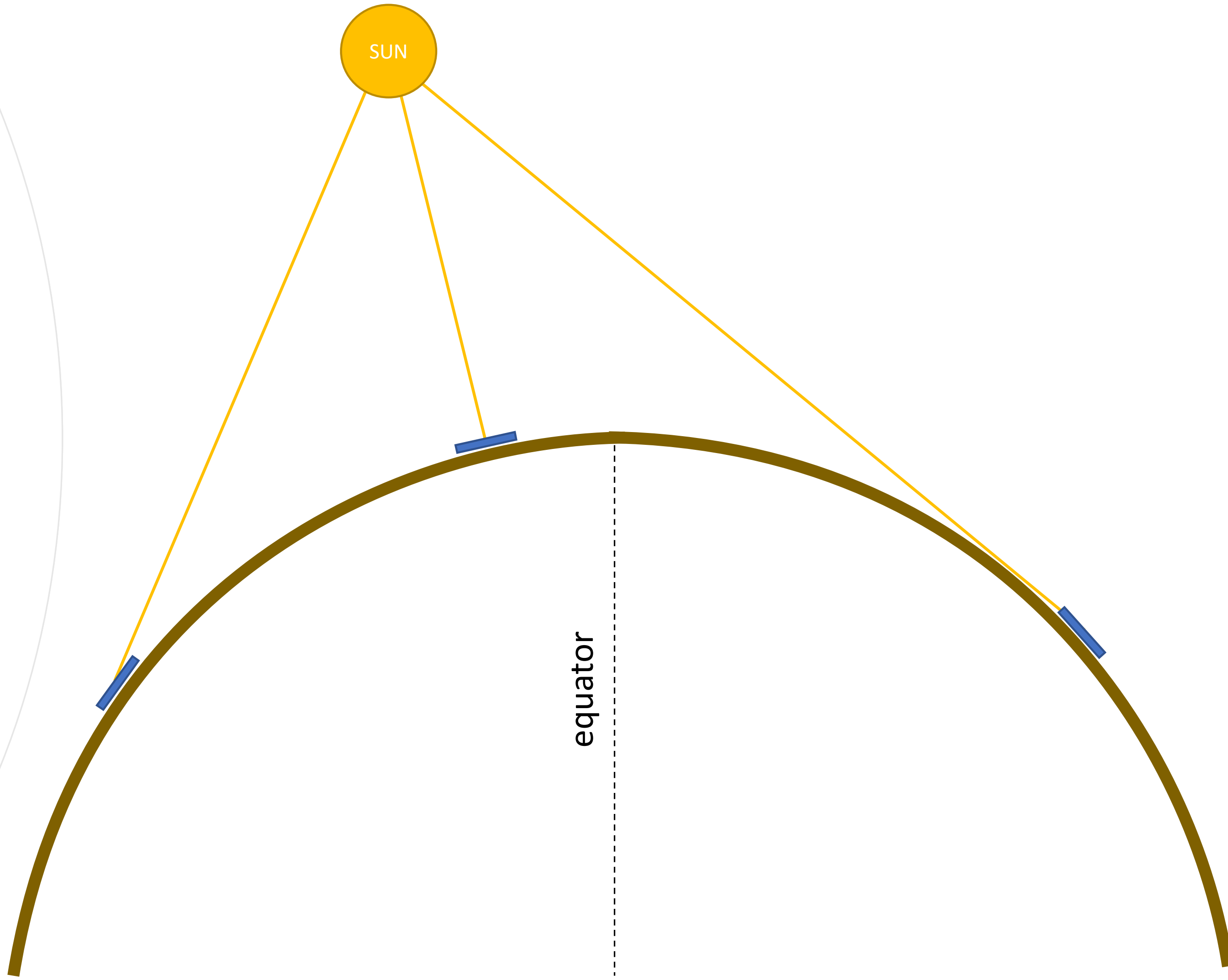


Harvesting sources: solar energy

The production of the panel depends on several factors:

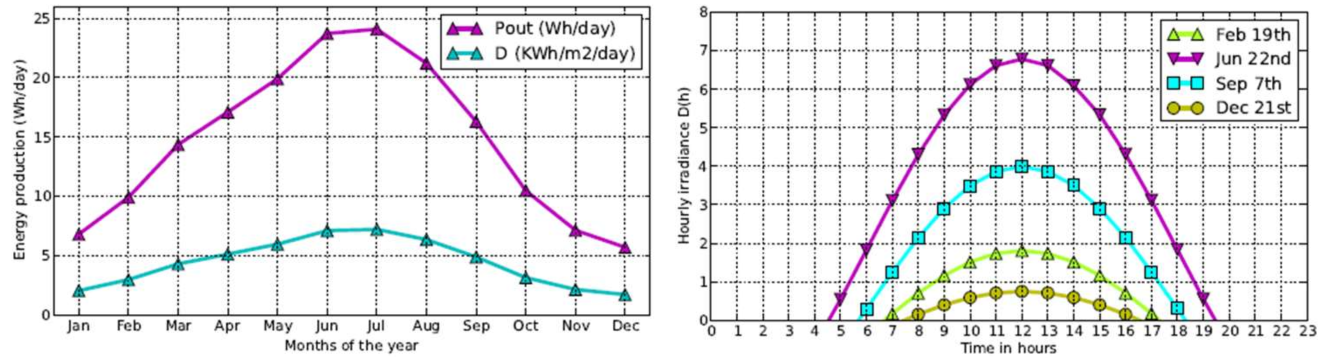
- Sun irradiance:
  - depends on the angle drawn by the Sun with respect to the vertical axis of the Earth surface …
  - … that depends on the geographical position of the panel…
  - … and on the day in the year
  - it also depends on the weather conditions
- Size of the panel
- Charging efficiency:
  - how much of the solar energy is actually available to charge the battery
  - depends on the electronics and on the materials of the panel

Sun irradiance...

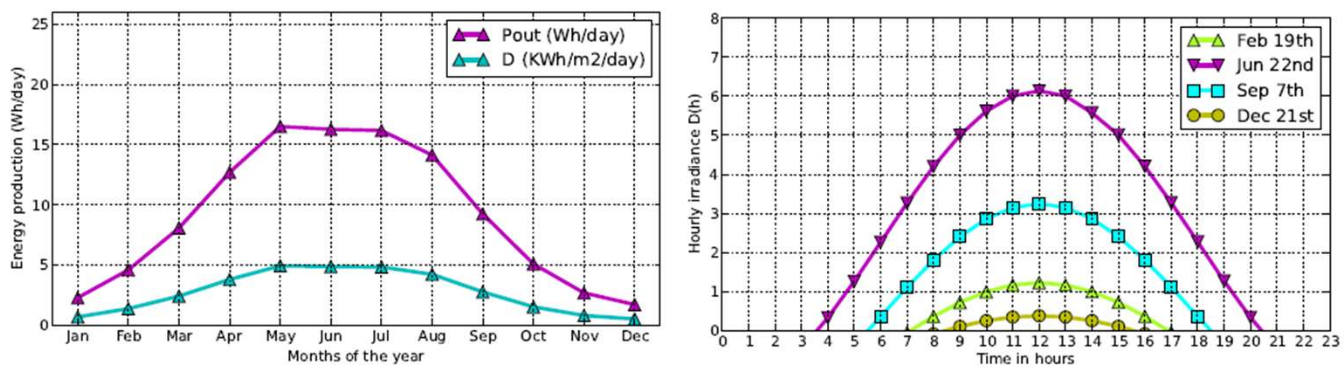


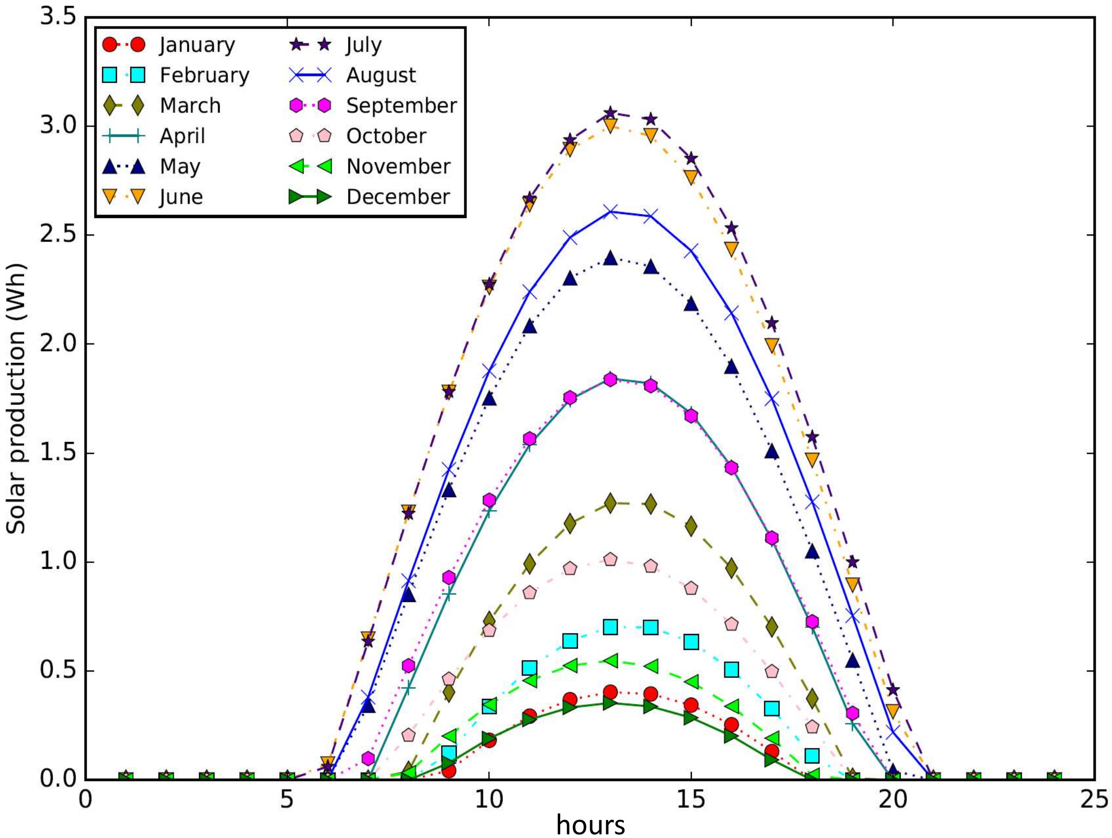
Harvesting sources: solar energy

Daily power output in Madrid and its distribution per hour



Daily power in Hamburg and its distribution per hour



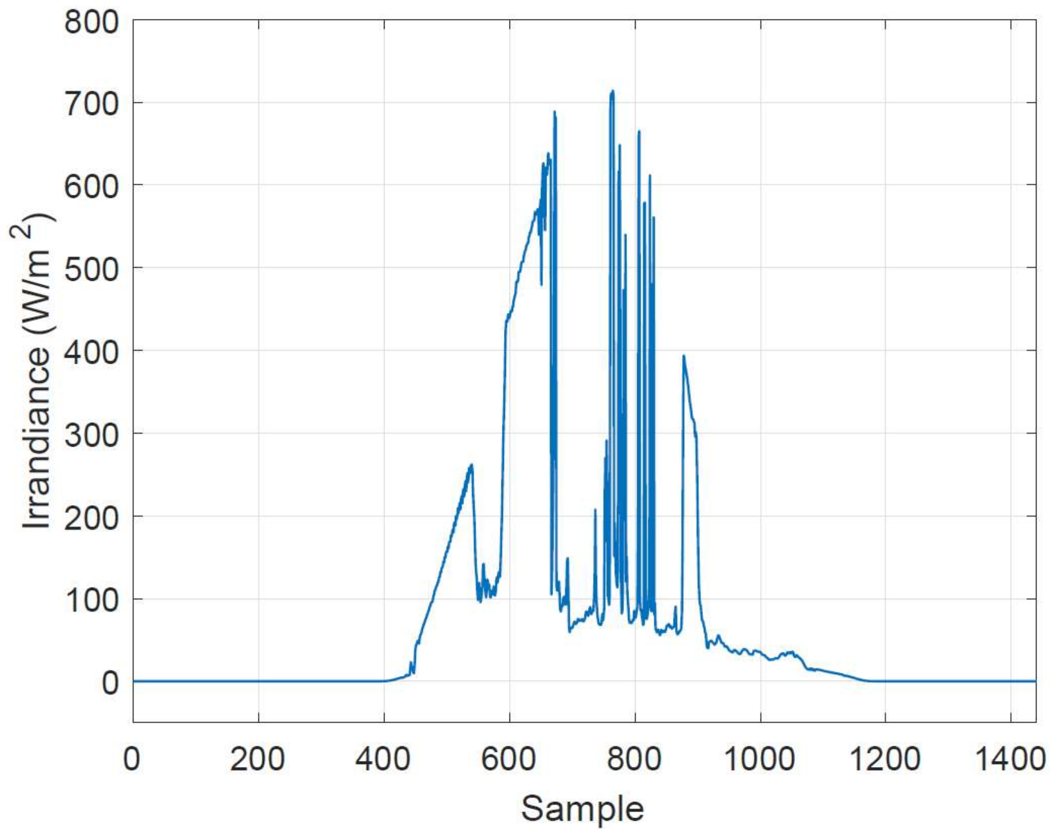


Ideal energy production

- Solar panel KL-SUN3W
- Expected energy production in Madrid
- Time frame of 24 hours, from 0:00 to 23:59

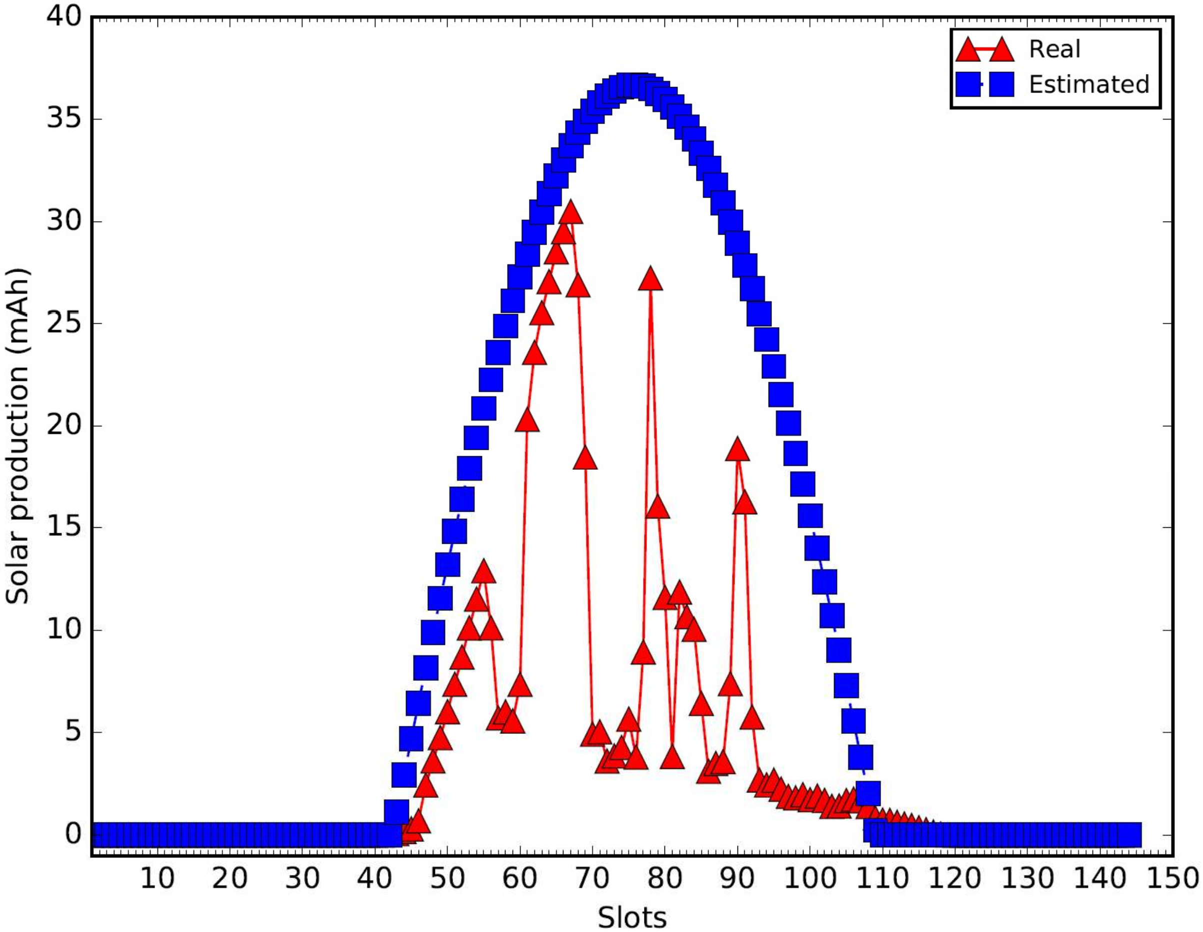
Considerations:

- availability of harvested energy varies very much in different periods of the year … (e.g December vs June)
- Usual solution: oversize the harvesting subsystem and the battery to match with the worst conditions



Irradiance

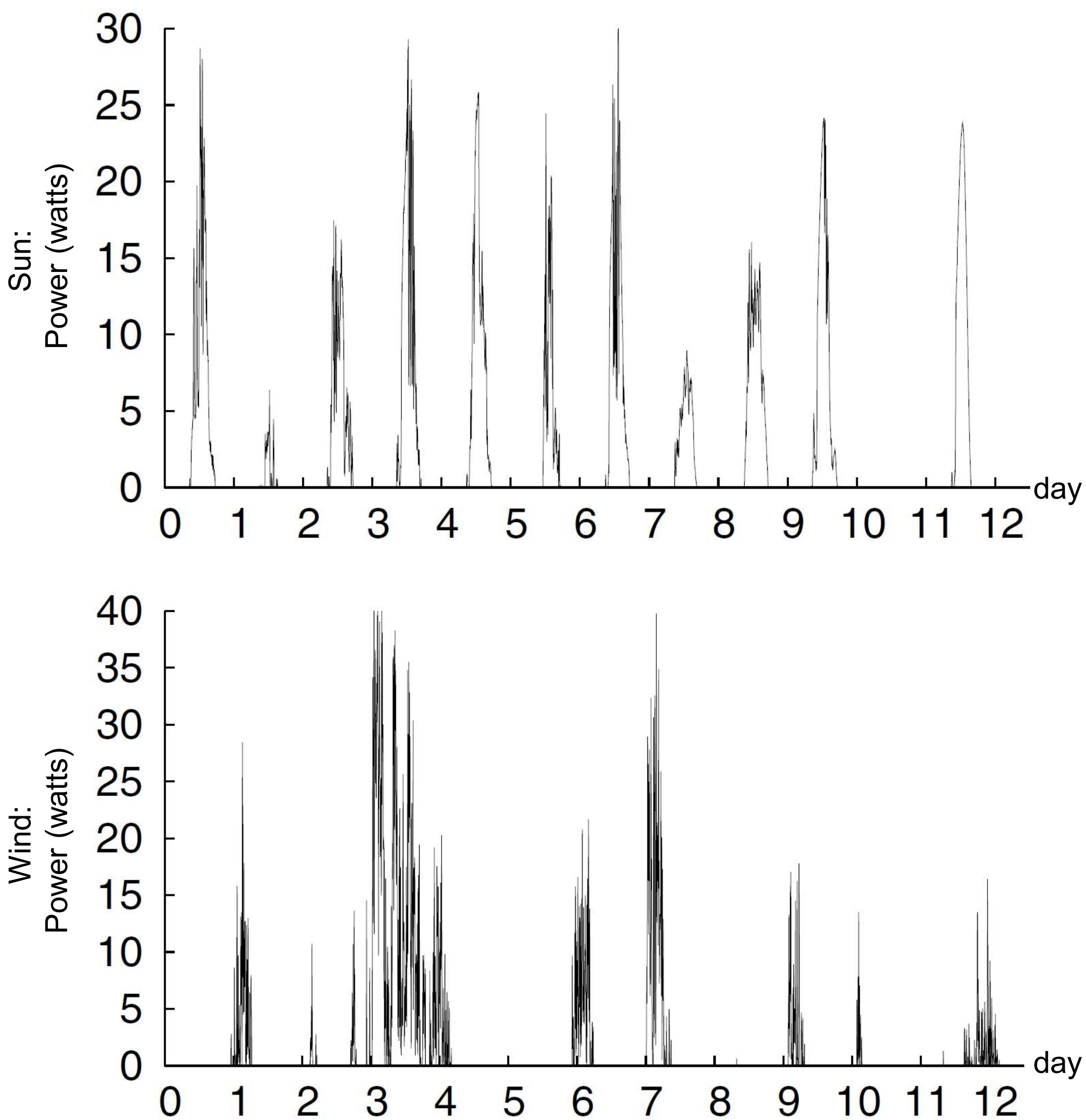
Measurements taken in Ciudad Real (Spain), 27th of March 2017



Real vs estimated production

- Solar energy production per 10-minutes slot in a day:
  - Measurements taken on the 27th of March 2017 in Ciudad Real (Spain)
  - Production in mAh

… cloudy day…



Solar vs wind energy production

- Power (watts) over a time frame of 12 days:
  - A) solar panel
  - B) wind turbine

Navin Sharma, Jeremy Gummeson, David Irwin, and Prashant Shenoy, “Cloudy Computing: Leveraging Weather Forecasts in Energy Harvesting Sensor Systems”, proceedings of IEEE Secon 2010

Limitations…

Energy harvesting does not mean that a device is always operational!

Consider a device harvesting solar & wind energy:

- In a night without wind there's no energy production
- The device relies only on the residual battery charge
- If the residual charge is insufficient …
- … it may drop below the minimum…
- … and the device turns off!
- It will turn on again in the morning…

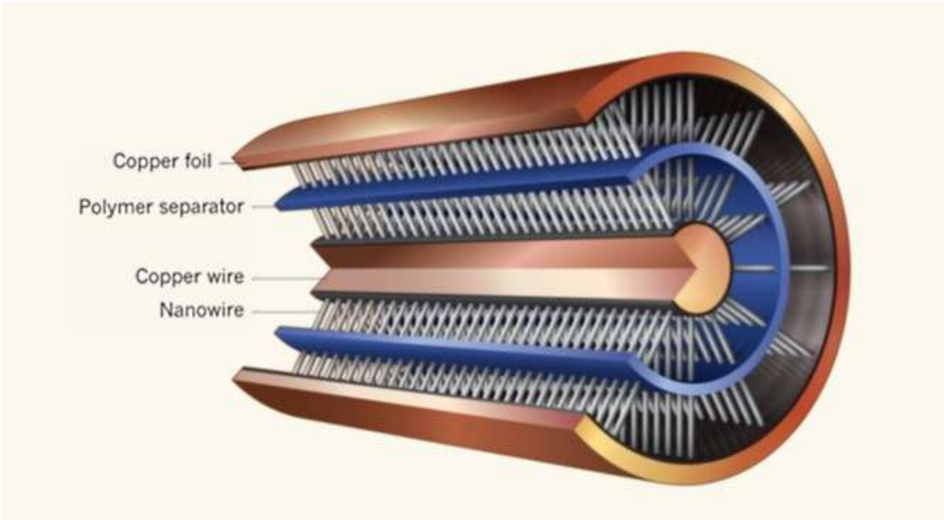
Storage technologies

Storage technologies: rechargeable batteries

Storage cells that can be charged by reversing the internal chemical reaction.

Battery type	Nominal voltage (V)	Capacity (mAh)	Weight energy density (Wh/kg)	Power density (W/kg)	Efficiency (%)	Self discharge (%/month)	Memory effect?	Charging method	Recharge cycles
SLA	6	1300	26	180	70-92	20	no	trickle	500-800
NiCd	1.2	1100	42	150	70-90	10	yes	trickle	1500
NiMh	1.2	2500	100	250-1000	66	20	no	trickle	1000
Li-ion	3.7	740	165	1800	99.9	<10	no	pulse	1200

- Sealed Lead Acid (SLA)
- Nickel Cadmium (NiCd)
- Nickel Metal Hydride (NiMH)
- Lithium Ion (Li-ion)

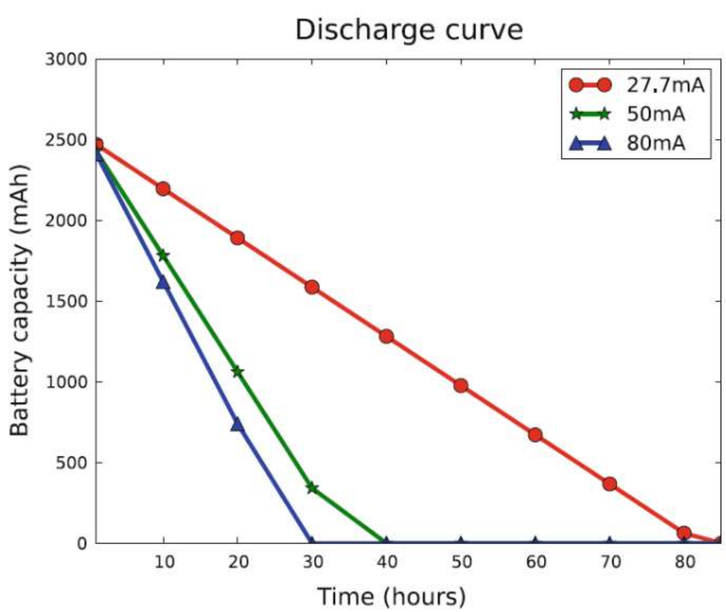


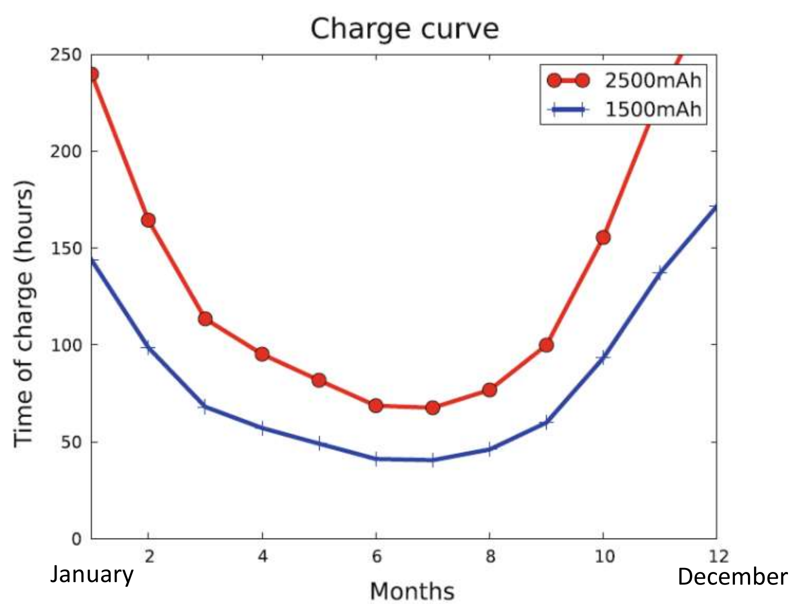
Super capacitors	
Weight energy density (Wh/Kg)	5
Efficiency (%)	97 – 98
Self discharge	5.9% per day
Memory effect	no
Charging method	trickle
Recharge cycle	infinite

Storage technologies: supercapacitors

- Do not need complex charging circuit
- Useful where ample energy is available at regular interval
- Can be used to buffer energy if energy source is jittery
  - trickle charge: when no-load, allows to charge at the same rate of discharge to keep the capacitor at full charge level

Battery: discharge and charge





Charge by means of a solar panel

measuring the battery charge & energy production

- All the methods for power management in energy harvesting assume that the device has fresh information about:
  - The battery charge
  - The actual energy production from the harvesting source
- The accurate measurement of the actual energy production is rather complex and most devices are not equipped for this…
- … and the same holds for the measurement of the battery charge.

- Here we see just a simple method

measuring the energy production

- Assume we have a method for measuring the battery charge $E_b(t)$ at time t…
- … and that the power consumption $p_c(t)$ in a time frame $[t_1, t_2]$ is known

- then the energy E_e produced in the same time frame is given by:

$$E_e = \left[\int_{t_1}^{t_2} p_c(t) \cdot dt + E_b(t_2) - E_b(t_1) \right]^+$$

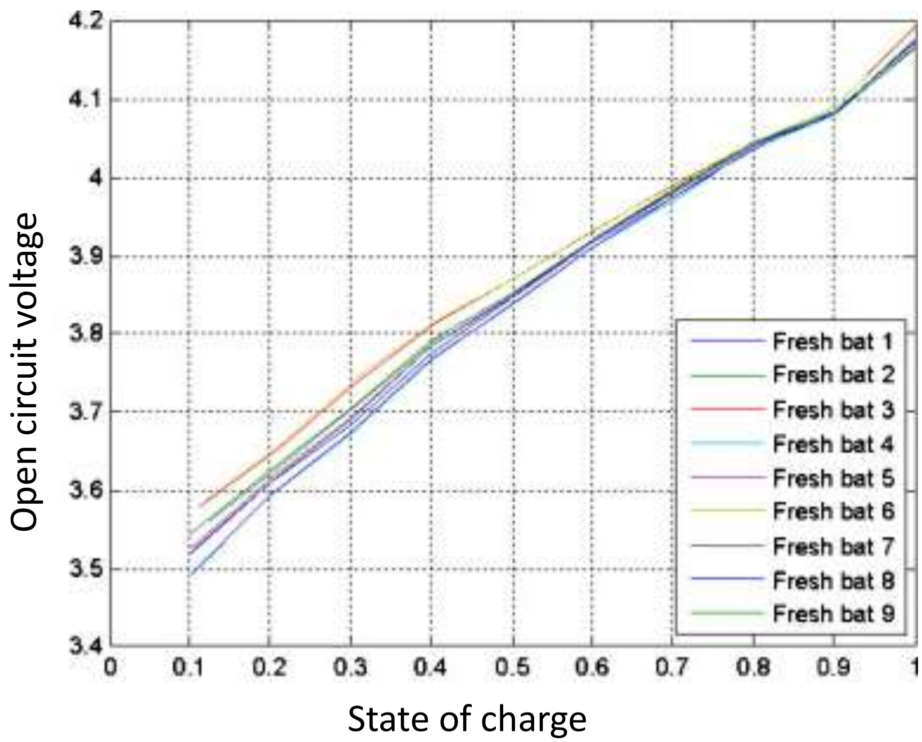
  - where $[x]^+ = 0$ if $x \leq 0$; $[x]^+ = x$ if $x > 0$
- Comes from the energy conservation principle
  - Assumes an ideal energy buffer… see later

measuring the energy production

- Drawbacks:
  - errors in the measurement of the battery charge imply errors in the estimation of energy production
  - if $[t_1, t_2]$ is too large it is difficult to know when the energy was available
  - The accurate estimation of the energy consumption $p_c(t)$ may not be easy

- A better way: use electronic circuits to measure the current flowing out of the harvesting source and its voltage

measuring the battery level

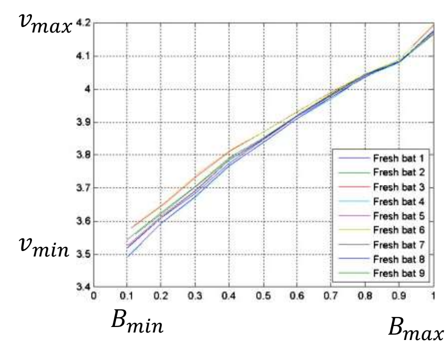


- Battery charge and voltage are related
  - the voltage drops down with the charge
- in the operational range of the battery the relationship can often assumed to be **linear**
  - ** not always! Depends on the specific technology

measuring the battery level

consider a device that measures voltage by means of a d-bits ADC

- let B_{min} , B_{max} be the minumum and maximum battery charges, respectively
- let v_{min}, v_{max} be the corresponding battery voltages



Then, the values x_{min} and x_{max} corresponding to v_{min}, v_{max} are:

$$x_{max} = 2^d - 1$$

$$x_{min} = ROUND\left[\frac{v_{min}}{v_{max}} \cdot (2^d - 1)\right]$$

measuring the battery level

let v be the current battery voltage, then, the value x read from the ADC is:

$$x = ROUND\left[\frac{v}{v_{max}} \cdot (2^d - 1)\right]$$

and the corresponding current battery level B is:

$$B = B_{min} + \frac{B_{max} - B_{min}}{x_{max} - x_{min}} \cdot (x - x_{min})$$

- assumes a linear scale voltage/charge

examples

Note: this table assumes $B_{min} = 300$ mAh

platform	v_{mi} **(volts)**	v_{max} **(volts)**	v_{ref} **(volts)**	**quantization levels (bit)**	x_{min}	x_{max}	**size of battery (mAh)**	x **(voltage sampling)**	**Battery charge (mAh)**
RPI	3,5	5	0,004883	10	716	1023	3000	1000	2798
Arduino	7	9	0,008789	10	795	1023	3000	800	359
Tmote	1,8	3	0,000732	12	2457	4095	3000	3277	1652

Question

Consider a device that samples the battery output voltage with an analog to digital converter (ADC) at 10 bits.

The battery has a maximum charge of 2000 mAh, and its maximum voltage (when fully charged) is 10 Volts.

When the battery reaches a voltage of 8 Volts the battery charge is 200mAh and it becomes insufficient to power the device.

Compute the battery charge when the ADC outputs:

- 1000
- 830
- 1023

Solution

Maximum battery charge: 2000 mAh; maximum battery voltage: 10 Volts.

Minimum battery charge: 200mAh that corresponds to an output voltage of 8 Volts.

Since the ADC operates with 10 quantization levels, and the max. voltage is 10 Volts, follows that:

- x_{max}=____________________
- $x_{\min}$=____________________________

Hence, when the ADC outputs:

- 920 → battery charge = ________________________________
- 830 → battery charge = ________________________________
- 1023 → battery charge = ________________________________

Solution

Towards energy neutrality

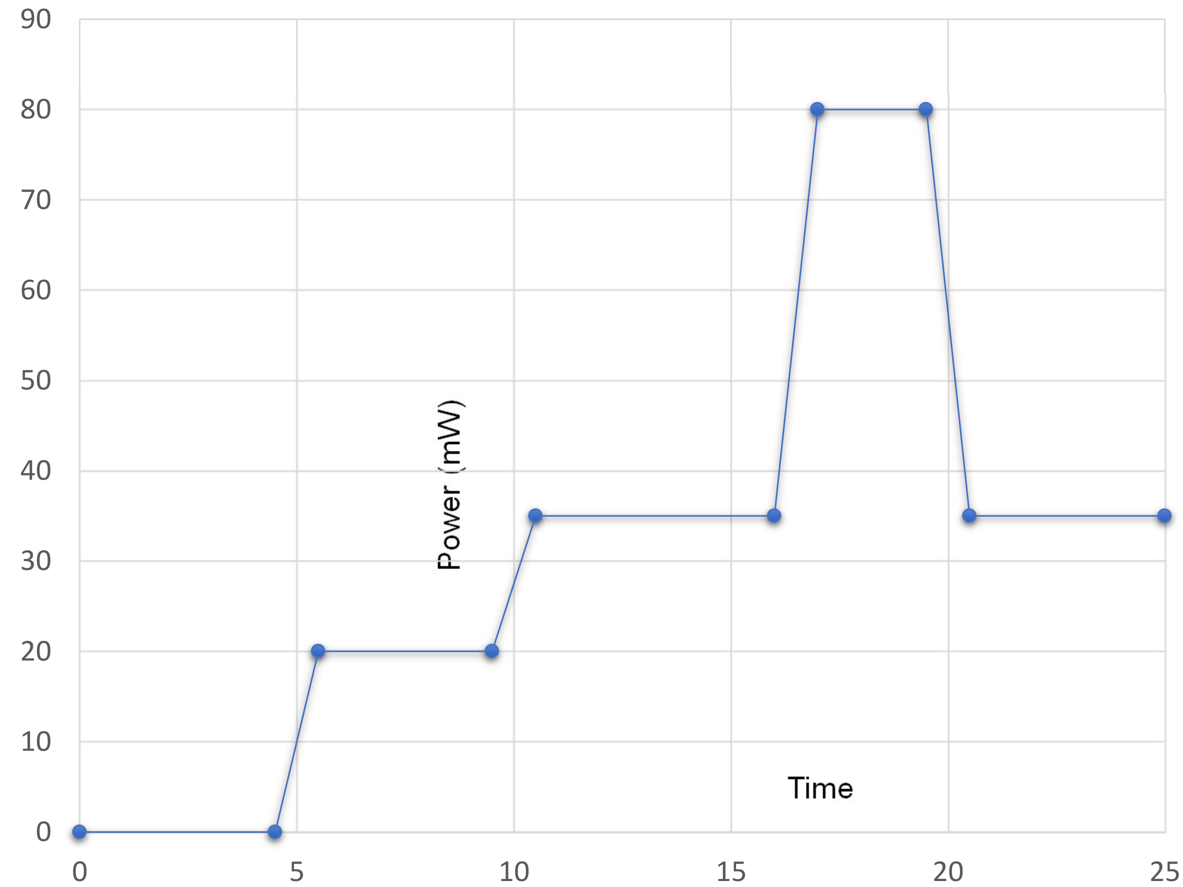
Modulating the load

- An IoT device has several components, each characterized by a power consumption (cost) per unit of time
- Components can be turned on/off by the processor when not in use
- Even the processor can be put in low-power mode

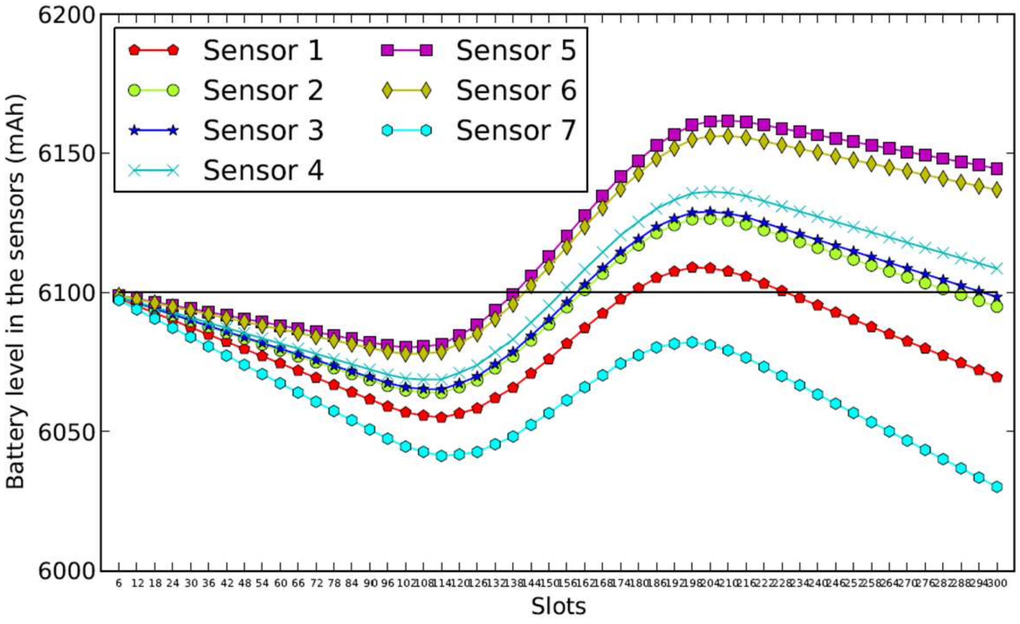
- Load modulation: reduce the amount of work to reduce the energy consumption
- For example a device may vary the sampling frequency of its sensor:
  - High sampling frequency implies high energy cost (but also high utility)
  - Low sampling frequency implies low energy cost and low utility

Load

- Power consumption varies according to the operational mode of a device
- Some activities like sampling, transmitting, receiving may initially wake the sensor node from sleep, start the processor and the radio
- Some activities may be background services:
  - routing a data packets
  - network management
  - etc.



Solar energy load vs supply



Battery level in sensors running different tasks, from 0:00 A.M. to 12:00 P.M.

- Sensors recharged with solar panels and running tasks with different duty cycles (and thus different loads)
- Sensors 1 and 7 consume more energy than that available
- Sensor 6 and 5 do not exploit all the energy produced

Energy neutrality

A device is **energy neutral** if it can keep the desired performance level «**forever**»

Much different than the classical approach of maximizing device's lifetime

Gives rise to two design considerations:

- Energy neutral operation
- Maximum performance

Energy neutrality

- Energy-Neutral Operation:

  How to operate such that, in any give time frame,
  the energy used is always less than the energy harvested?

- Note that the IoT systems may consists of multiple distributed devices:
  - each harvesting its own energy.
  - In this case, the performance does not depend only on the spatio-temporal profile of the available energy…
  - … but also on how this energy is used to deliver network-wide performance guarantees.

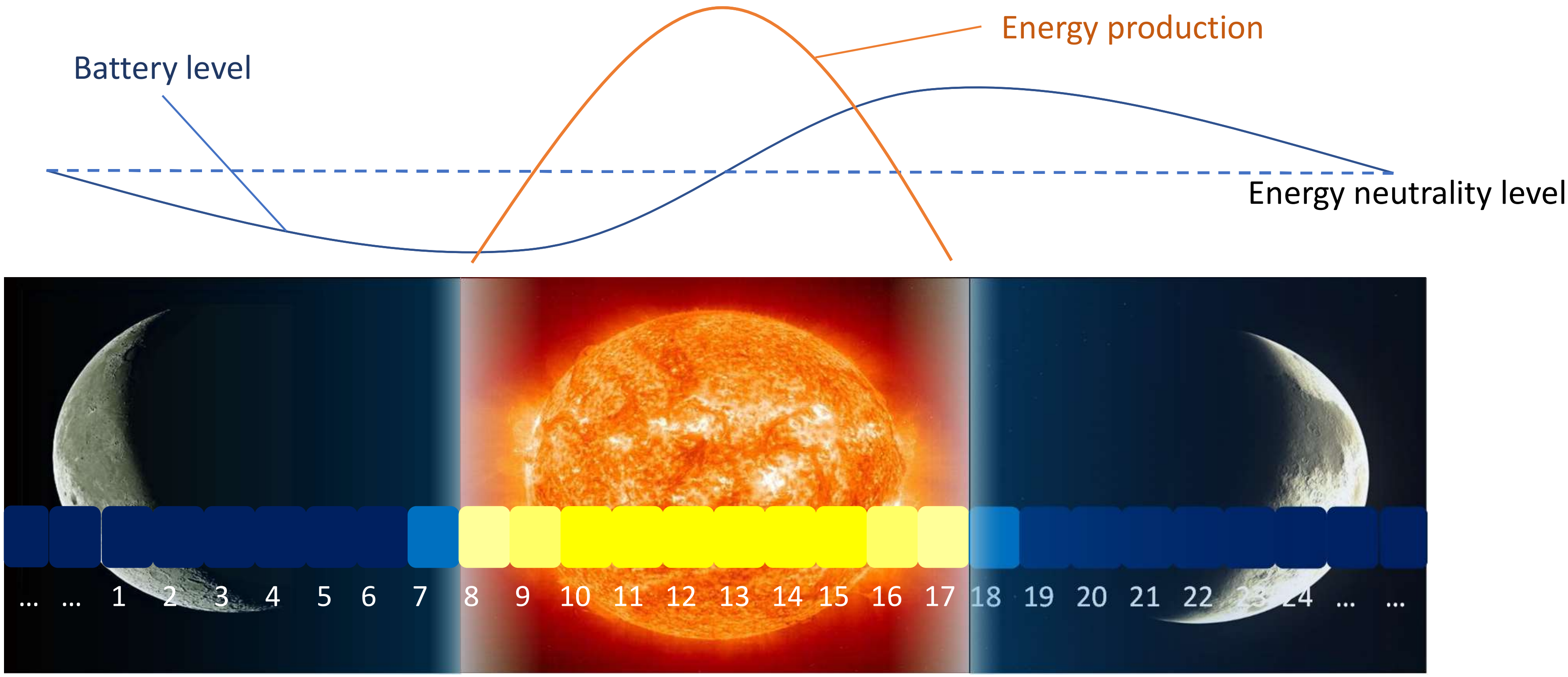
Energy neutrality

- Maximum Performance:

While ensuring energy-neutral operation,
what is the maximum performance level that can be supported
in a given harvesting environment?

- Again, this depends on the harvested energy at multiple distributed components.

Energy neutrality: load vs supply



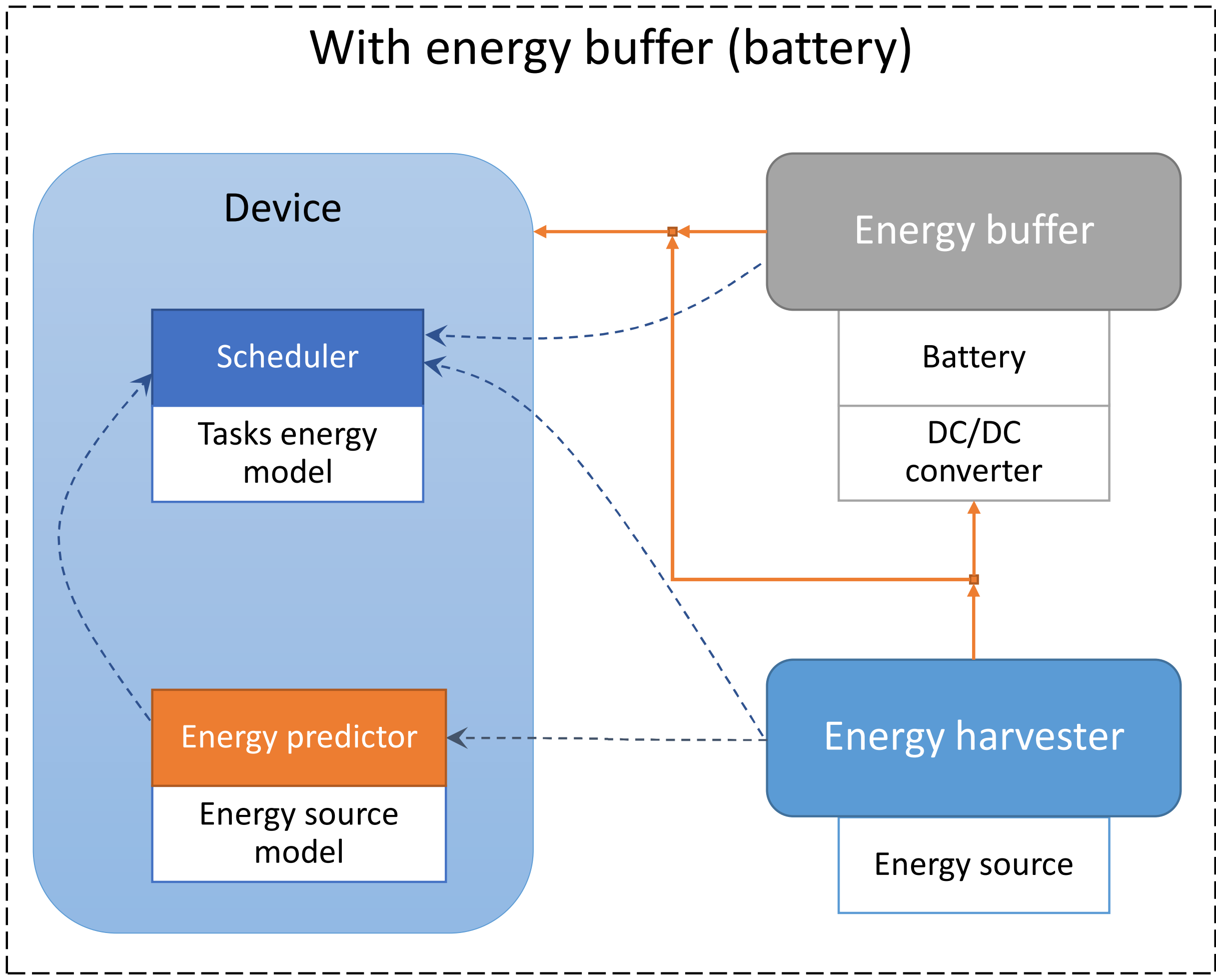
Ensuring energy neutrality

- Main elements:
  - The amount of harvested energy from (each) source
  - The current charge of the battery
  - The power consumption of the device (the load), which is modulated to ensure energy neutrality
  - The future energy production

- Future energy production estimation:
  - By means of weather forecasts or other methods
  - Subject to errors
  - Usually under pessimistic assumptions: errors maximized and expected production minimized

Harvest-Store-use architecture revisited

- Energy is harvested whenever possible and stored for future use.
- Energy is used later when either there are no harvesting opportunities, or the sensor tasks require more energy



Kansal’s approach to energy neutrality

Power management

- considers the case of predictable but uncontrollable sources

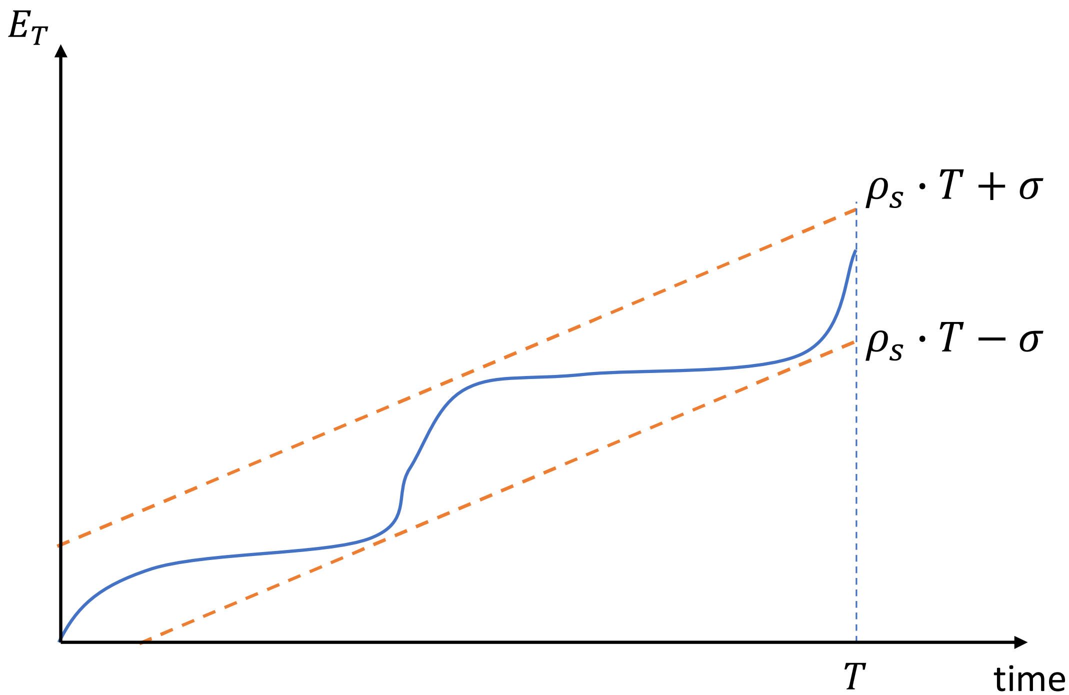
- why not unpredictable sources:
  - the problem is worse
  - it is rather difficult, if ever possible, to define guarantees on the performance
  - the problem is less interesting as well

- why not controllable sources:
  - the problem is straightforward
  - when you need energy just produce it!

Power management

- objective:
  - keep the system energy neutral
  - ensure the system never fails due to energy depletion
  - maximize the node's performance
- approach:
  - take current end expected battery charge into account
  - dynamically tune the device’s performance (and thus the load)
  - make sure the device neither operates below minimum performance levels nor switches OFF before the next recharge cycle

Condition 1: energy production



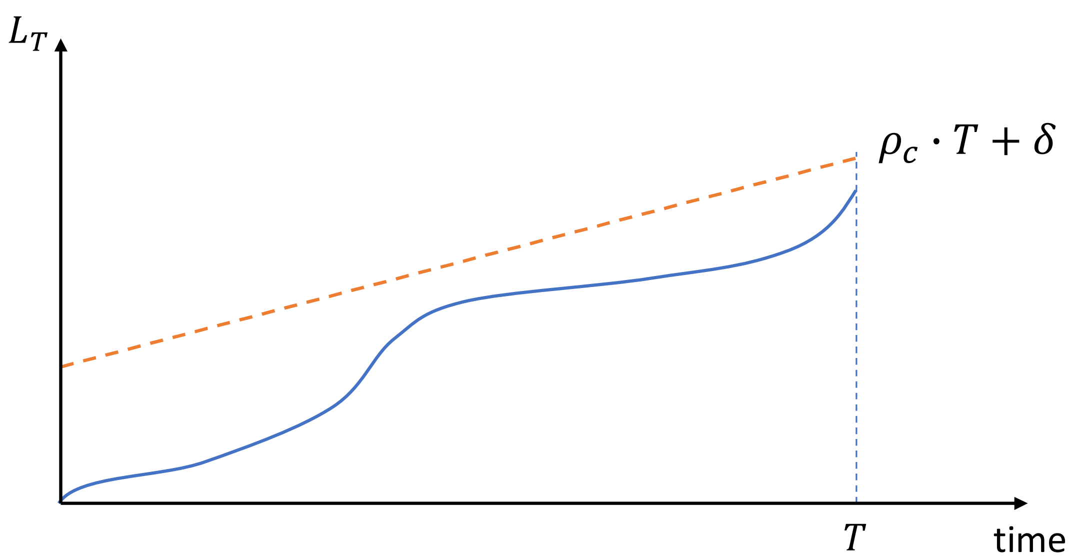
$$E_T = \int_T P_s(t)\, dt$$

It's the amount of energy produced in an interval $[0, T]$

$$\rho_s \cdot T - \sigma \le E_T \le \rho_s \cdot T + \sigma$$

for some ρ_s, σ real numbers

Condition 2: load



$$L_T = \int_T P_c(t)\, dt$$

It's the amount of energy consumed in an interval $[0, T]$

$$0 \leq L_T \leq \rho_c \cdot T + \delta$$

for some ρ_c, δ real numbers

Kansal's theorem

if the previous assumptions on E_T and L_T hold,

and the energy buffer is characterized by:

- charging efficiency η
- leakage ρ_{leak}

A sufficient condition for energy neutrality of the system is:

$$\begin{cases} \eta\rho_s \geq \rho_c + \rho_{leak} \\ B_0 \geq \eta\sigma + \delta \\ B_{max} \geq B_0 \end{cases}$$

The trend of production (considering the charging efficiency) should exceed the trend of load and of leak together

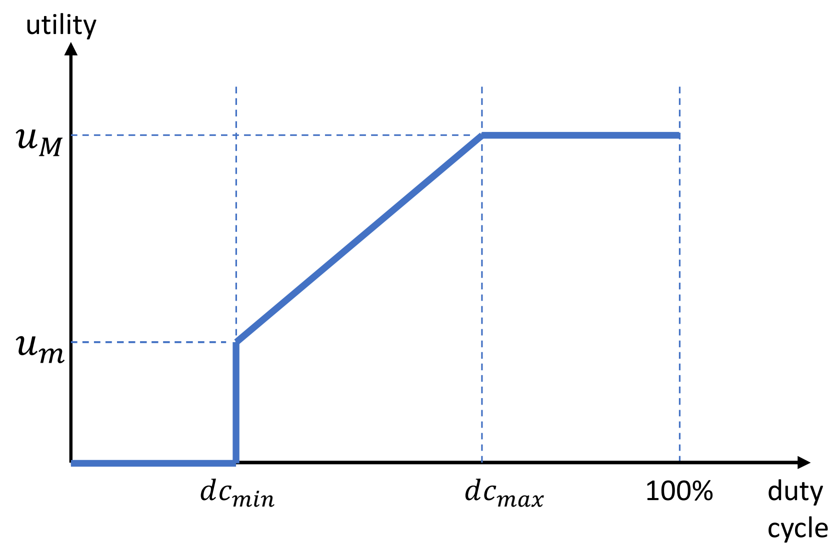
The initial battery charge should be sufficient in the worst case

The required initial battery charge should be admissible (less than the battery capacity)

Kansal's approach
utility and duty cycle

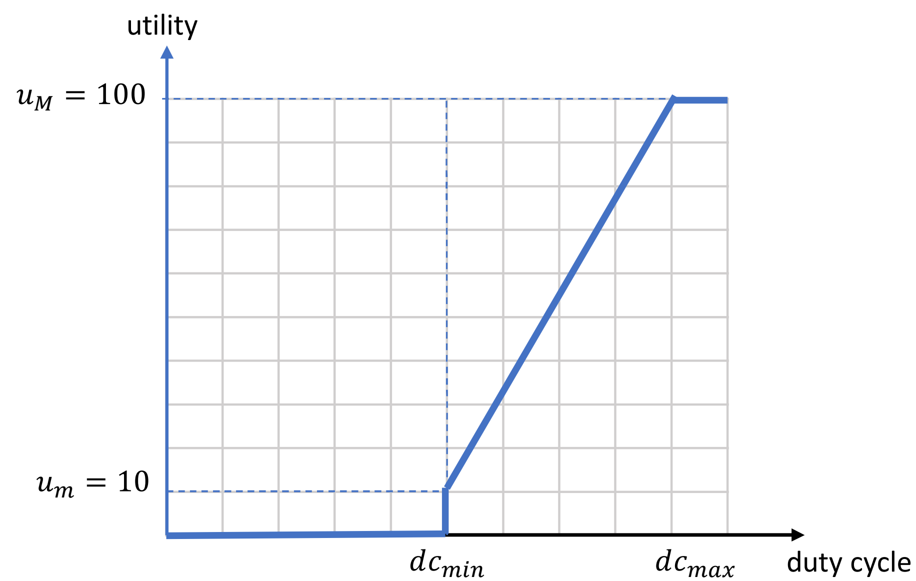
- Makes sure the system meets the two conditions:
  - Condition 1 on power production: just assumes it fits
  - Condition 2 on the load: adjusts dynamically the duty cycle
- About duty cycle:
  - reducing duty cycle reduces power consumption (there's almost a linear relationship)
  - hence a duty cycle =0% would meet the conditions… but the device would not do anything!
- Defines a relationship between duty cycle dc and utility $u(dc)$ of the application:

$$\begin{cases} u(dc) = 0 & if\ dc < dc_{min} \\ u(dc) = \alpha \cdot dc + \beta & if\ dc_{min} \leq dc \leq dc_{max} \\ u(dc) = u_M & if\ dc > dc_{max} \end{cases}$$



Example

$$\begin{cases} u(dc) = 0 & if\ dc < dc_{min} \\ u(dc) = \alpha \cdot dc + \beta & if\ dc_{min} \le dc \le dc_{max} \\ u(dc) = u_M & if\ dc > dc_{max} \end{cases}$$



Let:

- $dc_{ma} \ = 90\%$
- $dc_{min} = 50\%$
- $u_M = 100$
- $u_m = 10$

hence:

$$\alpha = \frac{u_M - u_m}{dc_{max} - dc_{min}} = \frac{90}{40} = 2.25$$

$$\beta = u_m - \alpha \cdot dc_{min} = -102.5$$

and:

$$u(dc) = 2.25 \cdot dc - 102.5$$

Example

Assume now

- $dc_{max} = 90\%$ corresponds to a consumption
$$p_{max} = 5\ mA$$
- $dc_{min} = 50\%$ corresponds to a consumption
$$p_{min} = 1\ mA$$

… and that the (average) power consumption is linear with the duty cycle. Hence:

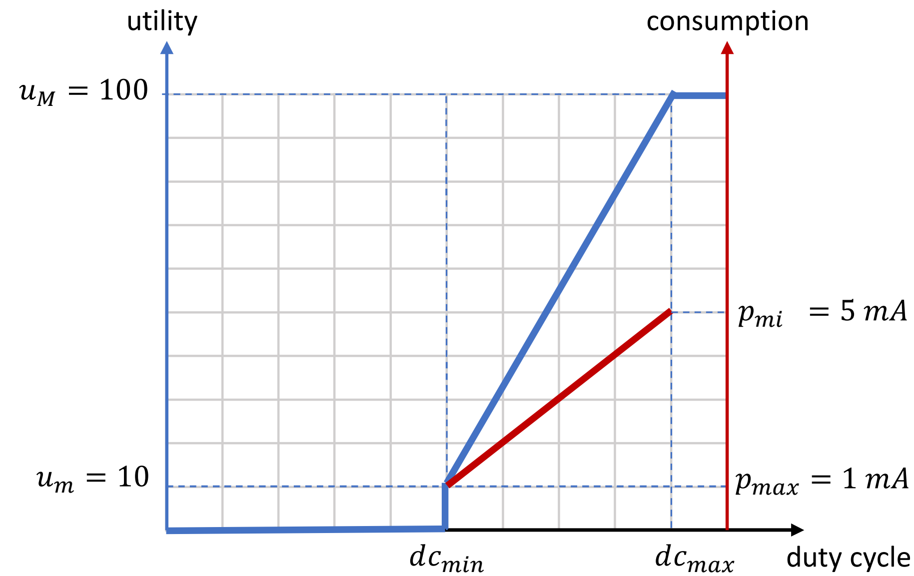
$$p(dc) = \rho \cdot dc + \sigma$$

where:

$$\rho = \frac{p_{max} - p_{mi}}{dc_{max} - dc_{min}} = \frac{4}{40} = 0.1$$

$$\sigma = p_{min} - \rho \cdot dc_{min} = -4$$

$$\begin{cases} u(dc) = 0 & if\ dc < dc_{min} \\ u(dc) = \alpha \cdot dc + \beta & if\ dc_{min} \leq dc \leq dc_{max} \\ u(dc) = u_M & if\ dc > dc_{max} \end{cases}$$



Kansal’s approach
utility and duty cycle

- This model of utility and duty cycle enables a modulation of the load on the energy harvesting device
- Consider for example a system designed to detect intruders:
  - The minimum and the maximum duty cycles imply a range of possible sampling frequencies
  - The higher the frequency the better the ability to detect intruders and the higher is thus the utility
  - The lower the frequency the lower is the ability to detect the intruders and thus the utility
  - Below dc_{min} the detection ability is impaired
  - Above dc_{max} it is useless to sample (the intruder is not that fast…)

Kansal's approach
optimization problem

- This leads to an optimization problem where:
  - the production varies over time (uncontrolled), but it is predictable (by assumption)
  - the load varies over time according to the duty cycle (controlled)
  - the battery charge varies over time accordingly

Since the energy source is predictable we have to
find the load so that the system is energy neutral
and the overall utility is maximized

Kansal's approach
optimization problem

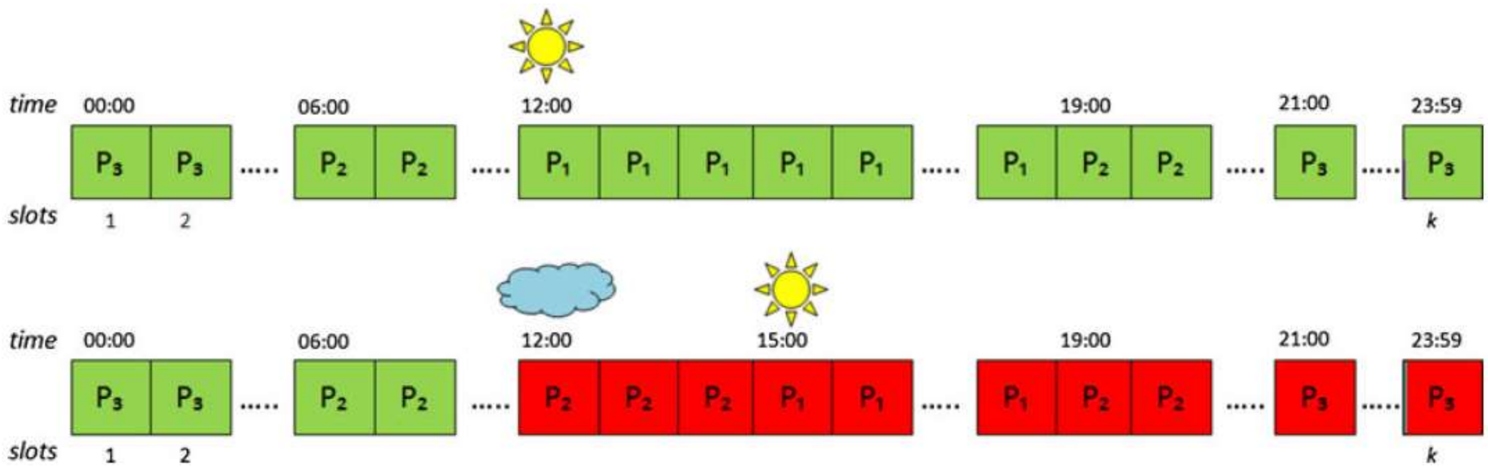
- This approach works well if the source is «well» predictable. This happens, for example, for the Sun

- Relevant aspects:
  - It is impractical to adjust dynamically the duty cycle too often
    - … afterall there is a system to manage below
    - and the risk would be to manage data sampled at irregular intervals
  - The Sun has a natural cycle of 24 hours

- hence…

Kansal's approach
optimization problem

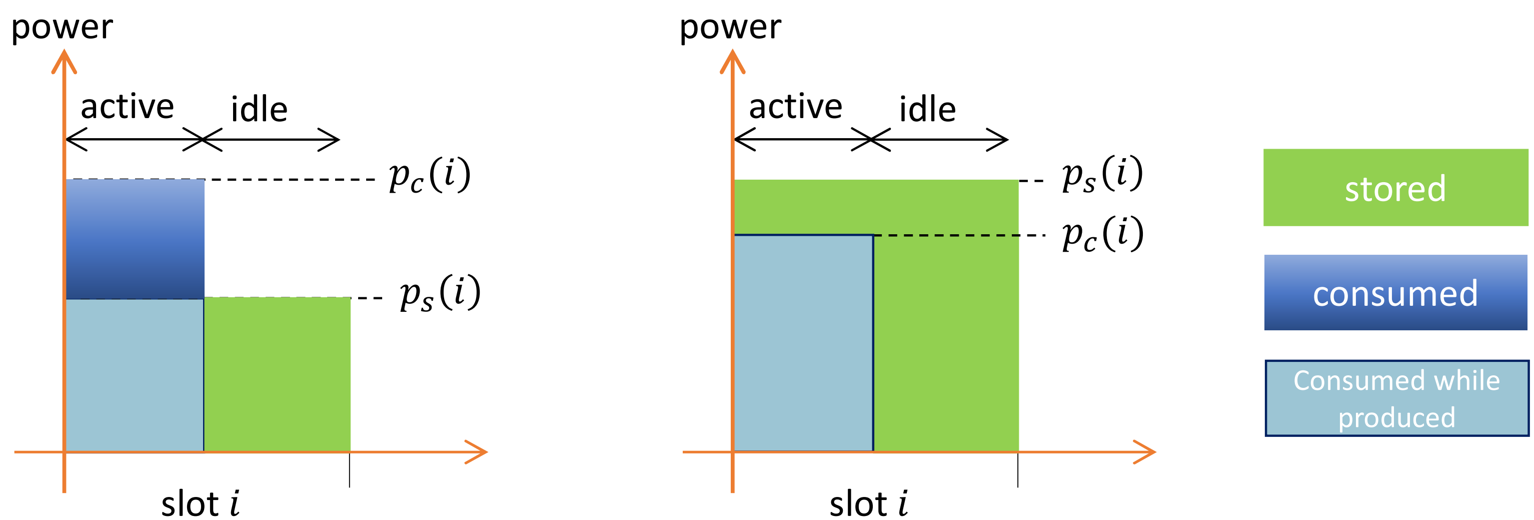
- Assume time is slotted (at each time slot set one duty cycle),
  - this assumption simplifies the job of scheduling and makes the system more «stable»
- Consider a time frame of 24 hours (after that, optimize again for the next day)
- Use forecasts to estimate the energy production of an entire day



Kansal’s approach
optimization problem

power harvested, stored and consumed in a generic slot i
- assume power consumed in idle is 0
- assume power production is constant in the slot

- $p_s(i)$: power harvested in slot i
- $p_c(i)$: power load in slot i (depends on the duty cycle used in the slot)



Kansal’s approach
optimization problem

- assigns a duty cycle at each slot
- does not requires the knowledge of the future production, but just an estimation
- builds over three components:
    - forecasts future power production based on past power production
    - uses a simple polynomial-time algorithm (suitable for low-power processors) to solve the optimization problem
    - performs a reoptimization (dynamic adaptation) if the actual power production deviates significantly from the expected one

Kansal’s approach
optimization problem

Let:

- k be the number of slots in a day
- $B(i)$ be the battery charge at the beginning of slot i
- $B(k+1)$ be the battery level at the end of slot k (i.e. at the end of the day)

Based on the expected power production $\tilde{p}_s(i)$ at each slot $i \in [1, k]$…

…assigns a duty cycle $dc(i)$, and hence a utility $u(i)$ to each slot

while requiring that $B(k+1) \geq B(1)$

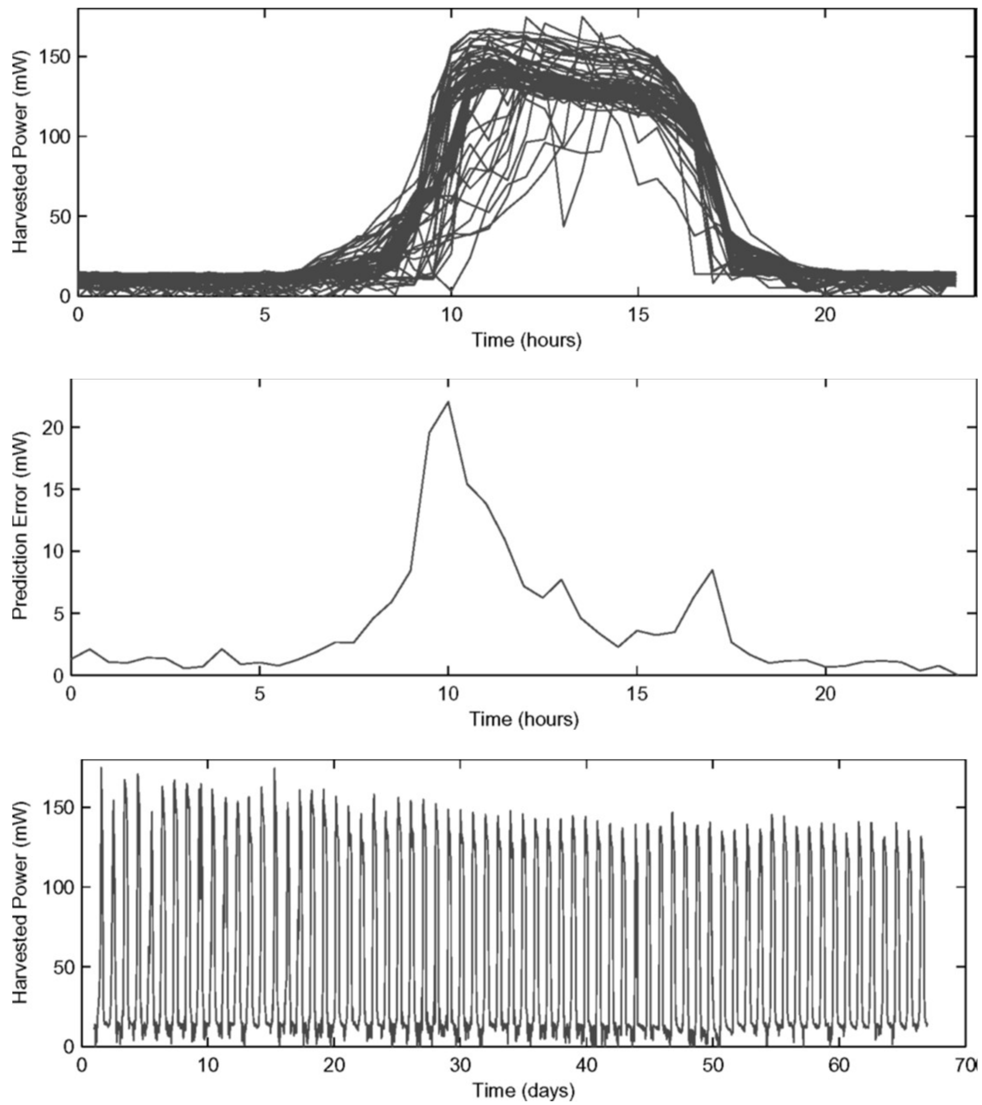
- i.e. that the system must be energy neutral

Forecast of power production

- Adopts a **exponentially weighted moving average (EWMA)** filter
- assumes that the production in a slot in a day will be similar to that of the day before…
  - Let $p_s^j(i)$ be the actual power production in slot i in day j (measured)
  - Let $\tilde{p}_s^j(i)$ be the estimated power production in slot i in day j
- The estimated power production in the same slot i in the next day $j+1$ is computed as:

$$\tilde{p}_s^{j+1}(i) = \alpha \cdot \tilde{p}_s^j(i) + (1-\alpha)p_s^j(i)$$

- where $\alpha < 1$ is a parameter (either a constant or it is adjusted dynamically based on the errors in the estimated vs actual production)
- hence the average of power production in a slot is the weighted average of the past power production in the same slot



Kansal’s experiments with an «Heliomote»

1. Harvested energy in a day (for many days)
2. Error of the EWMA-based prediction method
3. Energy harvested on several days using an heliomote sensor node

Kansal’s algorithm

Assumes that the actual power production matches the estimated power production

Let:

- p_{max} be the power consumption when operating at maximum duty cycle dc_{max}
- p_{min} be the power consumption when operating at minimum duty cycle dc_{min}

Considers two sets:

$S = \{i \in [1, k]: \tilde{p}_s(i) \geq p_{max}\}$ set of slots with overproduction – «sun slots»

$D = \{i \in [1, k]: \tilde{p}_s(i) < p_{max}\}$ set of slots with underproduction – «dark slots»

Kansal's algorithm

a first solution (possibly non optimum and not even admissible) would be:

$$dc_i = dc_{max} \;\; \forall i \in S \text{ (sun slots)}$$
$$dc_i = dc_{min} \;\; \forall i \in D \text{ (dark slots)}$$

- i.e. it assigns the maximum duty cycle in the sun slots and the minimum duty cycle in the dark slots.
- Then, in order to optimize the overall utility we have two options:
  - with this solution there is a surplus of power production at the end of the day
  - with this solution there is an underproduction of power at the end of the day

Kansal's algorithm
case 1 – overproduction

Case 1: there is a surplus p_r of power production at the end of the day:

// p_r is the surplus of energy: $p_r = B(k+1) - B_1 > 0$

while $p_r > p_{max} - p_{min}$ { *// p_r can power up a slot to the max utility*

  let $i \in D$ be a slot with $dc_i = dc_{min}$ *// let i be the minimum index*

  $dc_i = dc_{max}$;

  $p_r = p_r - (p_{max} - p_{min})$;

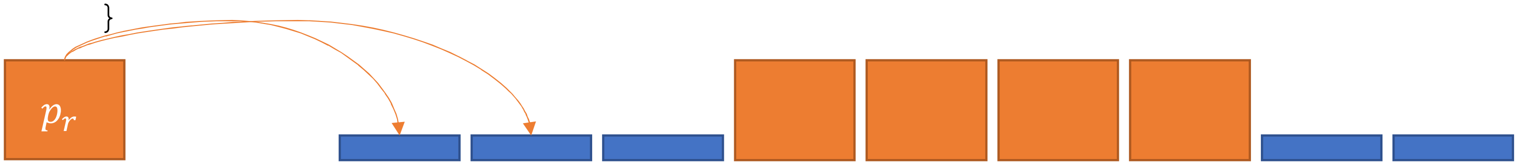
}

if $p_r > 0$ { *// p_r is insufficient to maximize the utility of another slot*

  let $i \in D$ be a slot with $dc_i = dc_{min}$ *// let i be the minimum index*

  $dc_i =$ DutyCycle($p_r + p_{min}$); *//increases the DC of the slot as possible*

}



Kansal's algorithm
case 2 – underproduction

if p < p

Case 2: there is an underproduction p_u of power at the end of the day :

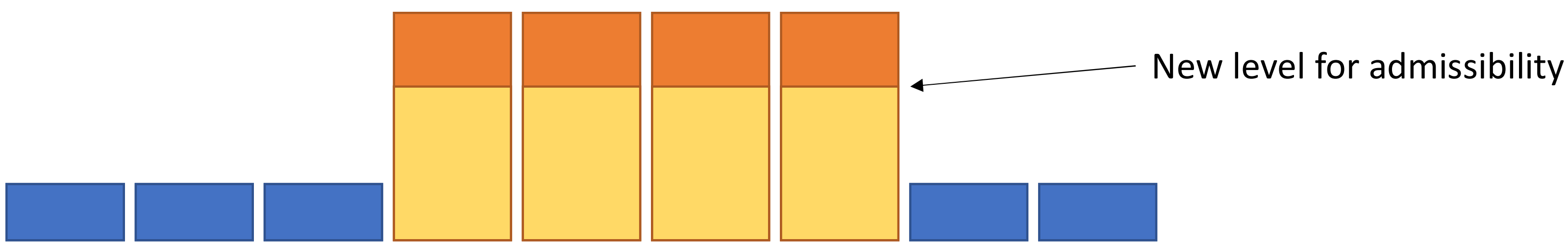
cus $p_u < 0$

cant

sustract

```
// p_u is negative, it's the underproduction: p_u = B(k+1) − B_1 < 0
if p_max + p_u/|S| < p_min
      return(-1) // no admissible solution, too much underproduction

else for each i ∈ S    //i.e for the slots with dc_i = dc_max
  //decreases the DC of the sun slot of the same quantity
  dc_i = DutyCycle(p_max + p_u/|S|);
```



Kansal’s algorithm – dynamic adaptation

- The Kansal’s algorithm assigns a duty cycle to each slot based on the expected power production
- However, it may happen during the day that the actual production differs significantly from the expected one
- The algorithm can thus make a dynamic adaptation of the duty cycles of the remaining slots in the day in order to:
  - Take advantage of an overproduction or
  - Make sure the system remains energy neutral in case of underproduction
- The algorithm for dynamic adaptation is very simular to the basic algorithm

Kansal’s algorithm - highlights

- The Kansal’s algorithm reaches the optimum
- It is simple and does not requires linear programming solvers
- It does not even have strong memory requirements
- It can be embedded in low-power devices

Kansal’s algorithm: drawbacks

- It is based on a linear scaling of the duty cycle
  - hence the sampling activity of a sensor will, in general, have irregular sampling frequencies, which may not be the best thing for some applications
- It uses only duty cycle adaptation to adapt the load to the power production
  - Hence it is not suitable to model more complex behaviours…
  - … like choosing between alternative transducers to achieve sampling tasks
  - … or using different processing algorithms

Exercise

The following table reports, for each slot in a day, the expected energy production of an energy-harvesting IoT device.

Assuming that:

- $dc_{max} = 90\%; p_{max} = 5mA\ ; u(dc_{max}) = 100;$
- $dc_{mi} \ \ = 50\%; p_{min} = 1mA; u(dc_{min}) = 10;$

Complete the table by assigning a duty cycle, an utility, and a power consumption to each slot, according to the Kansal's algorithm

Slot	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**
$\tilde{p}_s(i)$	0	0	0	2	6	11	10	7	3	0	0	0
dc_i												
$u(\cdot)$												
$p(\cdot)$												

Solution – 1/3

- $dc_{max} = 90\%; p_{max} = 5mA\ ; u(dc_{max}) = 100;$
- $dc_{min} = 50\%; p_{min} = 1mA; u(dc_{min}) = 10;$

Solution – 2/3

- $dc_{max} = 90\%; p_{max} = 5mA\ ; u(dc_{max}) = 100;$
- $dc_{min} = 50\%; p_{min} = 1mA; u(dc_{min}) = 10;$

Solution – 3/3

- $dc_{max} = 90\%; p_{max} = 5mA\ ; u(dc_{max}) = 100;$
- $dc_{min} = 50\%; p_{min} = 1mA; u(dc_{min}) = 10;$

Exercise

The following table reports, for each slot in a day, the expected energy production of an energy-harvesting IoT device.

Assuming that: → $M_{max} = 100$

$U_{min} = 10$

- $dc_{max} = 90\%$; $p_{max} = 5mA$; $u(dc_{max}) = 100$;
- $dc_{min} = 50\%$; $\boldsymbol{p_{min} = 2mA}$; $u(dc_{min}) = 10$;

Complete the table by assigning a duty cycle, an utility, and a power consumption to each slot, according to the Kansal's algorithm

Slot	1	2	3	4	5	6	7	8	9	10	11	12
$\tilde{p}_s(i)$	0	0	1	3	5	8	4	3	2	0	0	0
dc_i	50	50	90	90	90	90	90	90	90	50	50	50
$u(\cdot)$	10	10	100	100	100	100	100	100	100	10	10	φ
$p(\cdot)$	2	2	5	5	5	5	5	5	5	2	2	2

$$\rho = \frac{P_{max} - P_{min}}{dc_{max} - dc_{min}} = \frac{5mA - 2mA}{90 - 10} = \frac{3mA}{80} = 0{,}0375\,mA$$

$$p(dc) = \rho \cdot dc + \sigma$$

$$\sigma = P_{min} - \rho \cdot dc_{min}$$

$\sigma = 2mA - 0{,}0375mA \cdot 50\% =$

$2mA - 0{,}1875mA =$

$= 1{,}8125mA$

Solution – 1/3

- $dc_{max} = 90\%; p_{max} = 5mA\,; u(dc_{max}) = 100;$
- $dc_{min} = 50\%; p_{min} = 2mA; u(dc_{min}) = 10;$

energy production of an energy-harvesting IoT device.

Assuming that: $\rightarrow u_{max} = 100$

$u_{min} = 10$

- $dc_{max} = 90\%$; $p_{max} = 5mA$; $u(dc_{max}) = 100$;
- $dc_{min} = 50\%$; $\boldsymbol{p_{min}} = \boldsymbol{2mA}$; $u(dc_{min}) = 10$;

Complete the table by assigning a duty cycle, an utility, and a power consumption to each slot, according to the Kansal's algorithm

Slot	1	2	3	4	5	6	7	8	9	10	11	12
$\tilde{p}_s(i)$	0	0	1	3	5	8	4	3	2	0	0	0
dc_i	50	50	90	90	90	90	90	90	90	50	50	50
$u(\cdot)$	10	10	100	100	100	100	100	100	100	10	10	10
$p(\cdot)$	2	2	5	5	5	5	5	5	5	2	2	2

$$\rho = \frac{P_{max} - P_{min}}{dc_{max} - dc_{min}} = \frac{5mA - 2mA}{90 - 10} = \frac{3mA}{80} = 0{,}0375\,mA$$

$$p(dc) = \rho \cdot dc + \sigma$$

$$\sigma = P_{min} - \rho \cdot dc_{min}$$

First use

last

order of u

1) $\xrightarrow{Ams}$ Go map

2) reduce slots

5 6

55 55

3 3

Solution – 1/3

- $dc_{max} = 90\%; p_{max} = 5mA\ ; u(dc_{max}) = 100;$
- $dc_{min} = 50\%; p_{min} = 2mA; u(dc_{min}) = 10;$

Task-based model for energy neutrality

• TASK-BASED MODEL TO ENERGY NEUTRALITY: models behaviour of the device according to ≠ tasks (no in terms of single DC, like Kansal)

A task-based model of IoT apps

MOTIVATION

- Usually, an application for an IoT device implements 4 ~~steps~~ TASKS in a cycle: => related to ≠ DCs
  - Sensing
  - Storing
  - Processing
  - Transmitting
- All these phases may have different alternative implementations

old
• exercise: computing Heart rate
1) in device itself and transmit
or
2) transmit everything and computing elsewhere (server)
or
3) collect several data, transmit only once
or
4) transmit in real time every single data

– in terms of
=> • Functionality of application => change nothing
• NON-FUNCTIONAL behaviour of app => changes!
(real-time, response time...)

=> use same concept 4 Energy neutrality
• considering that the tasks are not all together (can have a DC: 1 sensing, process data locally, transmit and compute elsewhere or a mix)

=> can consider devices as having options to run ≠ alternative implementation of app
=> each possible implementation corresponds to ≠ energy profile
=> TASKS
=> can take this info to make optimizations and make device energy neutral
=> scheduler decides which task execute at a given time frame

=> Scheduling task must be notified to server (this prob was also if change DC)
=> change rate to transmit
=> server waits data at ≠ time frame
=> SAME PROBLEM

A task-based model of IoT apps

Example – an application can:
• CAN EMBED INTO A DEVICE

- choose among different transducers to monitor some envi. process
  - Each with a different energy consumption and performance (resolution) for each ≠ transducer
- OR disable data processing local and only store data when the battery level is too low => energy COST - UTILITY no longer linear
- scale up/down the sampling frequency
- use more transducers to increase its utility level
- communicates more/less frequently and reliably

=> what should be do must be in fun of the device

• if you want ↑ UTILITY => use transducer that consumes more
• if you want to save energy => use transducer that consume less => UTILITY ↓

=> RELATION BETWEEN UTILITY - ENERGY COST : NO-LONGER LINEAR
(out of knapsack approach)

COST ⟺ utility
↳ no longer linear => no continuous values
=> discrete possibility to change cost / utility

A task-based model of IoT apps

Different implementations may imply a different behaviour of the server (either in the gateway or in the cloud)

Example 1:

- If the application process all data and transmits only the results…
- …then the server may just store the received data.

Example 2:

- If the application just transmits all the sensed data…
- … then the server has to process and then store the data.

The system functionality does not change, but the implementation on the IoT devices changes a lot…

We call *task* an implementation of the application in the IoT device.

- An IoT device may have several alternative tasks to implement an application.
- Each task characterized by:
  - A power consumption (cost) per unit of time
  - A utility gained when it is executed
- Now tasks even in Aduino…
  - https://www.arduino.cc/en/Tutorial/MultipleBlinks/
  - https://www.arduino.cc/en/Reference/Scheduler

Load modulation:

- achieved by scheduling the appropriate tasks in execution (over time)
- … and of course, the server needs to know which task is running

- every task is stored in firmware, takes space
  → cannot have many of them

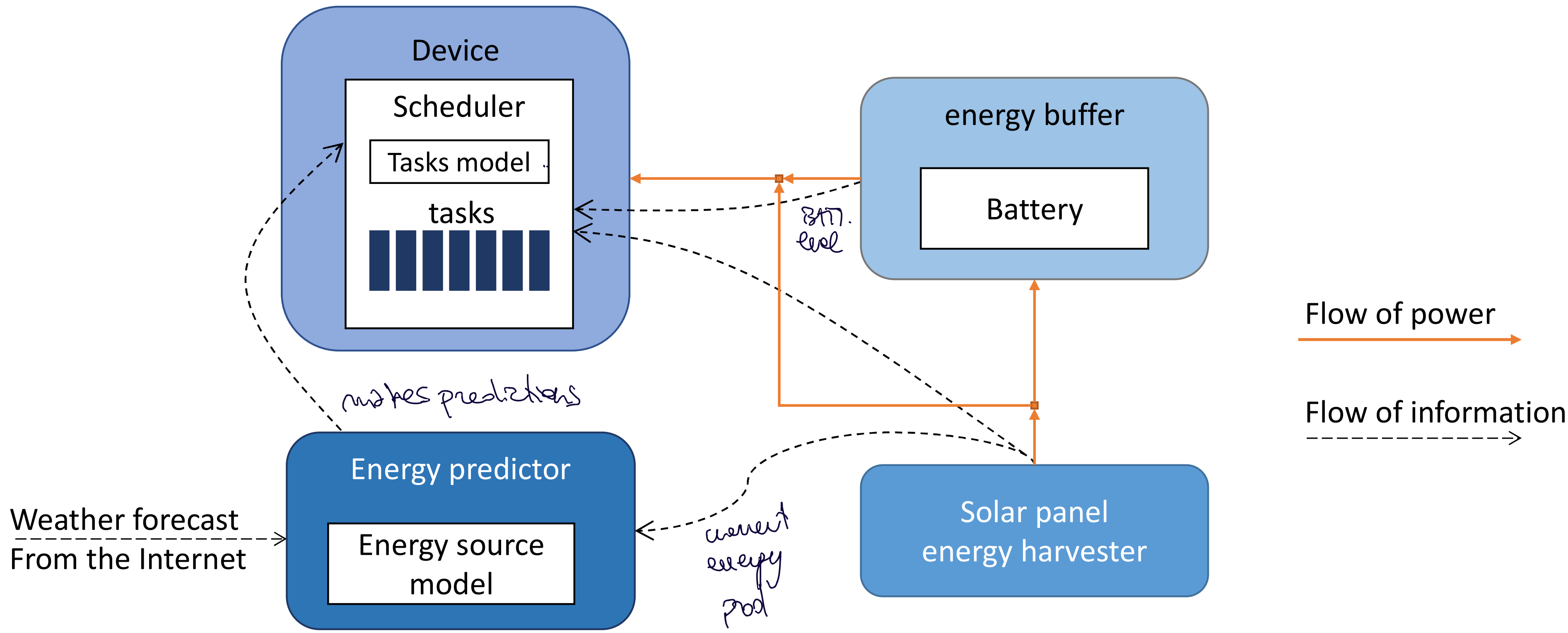
Modulating the load with tasks

So we are ... modulating the load, ramping it up or down

- each task is characterize by a power consumption per unit of time, and utility per unit of time (when is executed)

• CHANGE BEHAVIOUR IN SW PART: most on scheduler: task model = ∀ task tells its utility and cost
=> here, task model is discrete, task by task, 2 infos: utility and cost

(Kansal was: utility <=> DC
DC <=> power
with 2 linear scales)



Modulating the load with tasks

• as Kansal => time is slotted → also here, switch tasks, inform server => don't want change too frequently

• was cd. use time slots to cut change sampling rate continuously => no effective way to analyze data

=> here no DC to a slot but => task to a slot

Task-based model

- Slotted time, k slots
  - e.g. 24 slots a day (slots of 1 hour)
- A scheduler assigns one task per slot
  - $x_{i,j} = 1$ iff task j assigned to slot i
  - Scheduler runs once per day (e.g. at midnight)

Bin MAT

Tasks and their properties

(X)

Alternative tasks impl.	Energy cost	utility
T_0	c_0	u_0
T_1	c_1	u_1
T_2	c_2	u_2
…	…	…

n Tasks

Initial battery level $B(1)$

Initial battery level slot 3: $B(3)$

final battery level slot 3: $B(4) = B(3) - c_2 + \ldots\ldots$

final battery level $B(k+1)$

Task	T_1	T_2	T_2	T_0	T_0	T_0	T_1	T_1	T_1
Slot #	1	2	3	4	5	6	7	8	9

15 min / 30 min depends

(X) ∀ task: energy cost & utility, no linear relationship, just a straight association between cost-utility

• Binary matrix allows to the scheduler to assign a task to a slot (not in practice, abstraction)

↳ tells ∀ slot which is the task assigned, has 1,1 in a column, ...

• In addition: • initial battery level B(1), final B(k+1) • at end of slot 3, Battery level is B(4) = B(3) + what produced

in slot 3
- what we have consumed,
- taking into account charge efficiency η

Task-based model

need

- Weather forecast to estimate the expected energy production in slot i : $p_s(i)$
- Energy consumption in slot i : $p_c(i)$

  next term
  - it is the power consumption of the task assigned to the slot (see $x_{i,j}$...)
- $p_s^+(i) = [p_s(i) - p_c(i)]^+$

  at any slot we can consider expected energy production $P_s(i)$ computed by the prediction
  expected consumption in slot i, depends on task we're running on the slot
  - Expected power produced and not consumed in slot i
  - ... that thus recharges the battery
- $p_c^-(i) = [p_c(i) - p_s(i)]^+$
  - Expected power consumed from the battery in slot i
- η : charging efficiency of the battery

  ↳ models charge efficiency

$\Downarrow$ 2 cases:

1) Produce > consume => part of energy goes to battery $P_s^+(i)$
2) or cons > produce

$\Downarrow$

Take out energy from battery, called $P_c^-(i)$

NB: if we are putting => taking out is zero
if taking out => putting is zero

battery level at begin of the slot + put into battery − taken out

Cons: cons but

Task-based model

The battery level at the end of slot i, is thus:

rewrite like ground

$$B(i+1) = \boxed{\min}\{B_{max}, B(i) + \boxed{\eta \cdot p_s^+(i)} - \boxed{p_c^-(i)}\}$$

END

this

kelp p

eq

send slot

The battery level cannot be above B_{max}

Battery charge in the slot

Power of the battery consumed in the slot

Tasks and their properties

Alternative tasks impl.	Energy cost	utility
T_0	c_0	u_0
T_1	c_1	u_1
T_2	c_2	u_2
…	…	…

Initial battery level $B(1)$

Initial battery level slot 3: $B(3)$

final battery level slot 3: $B(4)$

final battery level $B(k+1)$

Task	T_1	T_2	T_2	T_0	T_0	T_0	T_1	T_1	T_1
Slot #	1	2	3	4	5	6	7	8	9

WE are making an ASSUMPTION:

1) - during a slot, the CONSUMPTION IS CONSTANT, works if slot reasonably short

2) May possible, level of battery exceeds B_{max} => which is not possible, if put more than you can => throw away energy => that's why the minimum (B_{max}, other)

Optimization problem

- The optimization problem thus becomes:

$$\max \sum_{i=1}^{k} \sum_{j=0}^{n} x_{i,j} \cdot u_j$$

Utility of the tasks assigned to the slots

- Given:

$$\sum_{j=1}^{n} x_{i,j} = 1 \;\; \forall i \in [1,k]$$

Exactly one task per slot

$$B(1) \leq B(k+1)$$

Energy neutrality

$$B_{min} \leq B(i) \;\forall i \in [1,k]$$

In a slot the battery charge cannot be below B_{min}, otherwise the device will stop working

$$B(i+1) = \min\{B_{max}, B(i) + \eta \cdot p_s^{+}(i) - p_c^{-}(i)\} \;\; \forall i \in [1,k]$$

The battery charge at the end of slot i depends on the initial charge, and the production and consumption in the slot

- ∀ slot, pick up a task
- task tells to device what do in that slot
  (sample at given frequency, processing, store all in flash and transmit every 10 samples | another task will say something different)
- then, will be the ESTIMATION of BATTERY LEVEL at any slot depending on task assigned to that slot and depending on EXPECTED POWER PRODUCTION for that slot

- given this, can formalize the problem as INTEGER LINEAR PROGRAMMING PROBLEM for which I want to MAXIMIZE THE UTILITY FOR ALL THE SLOTS

1) $$\max \boxed{\sum_{i=1}^{k}\sum_{j=0}^{n} x_{i,j}\cdot u_j}$$

- $x_{i,j}$ tells: ∀ slot i, what is task assigned to that slot
  es $x_{i,j}=1$ in slot i, will run task j
  (5, 4) (5) (4)
- u_j: utility of that task, task $j=4$
- Summing all, considering that you can have only 1 task per slot
  ⇒ this gives the sum of the UTILITIES ∀ slot.

- Constraints: tells can assign only 1 task per slot:

2) $$\boxed{\sum_{j=1}^{n} x_{i,j} = 1 \ \forall i \in [1,k]}$$

- if sum for all the possible tasks, for 1 given slot i
  ⇒ the sum will be 1
  ⇒ just one of the n-tasks will be assigned to slot i

3) $$\boxed{B(1) \le B(k+1)}$$

⇒ • solution should be admissible (end of battery level $B(k+1) \ge B(1)$ → battery level we have started)
⇒ SOLUTION MUST BE ENERGY NEUTRAL

4) $$\boxed{B_{min} \le B(i) \ \forall i \in [1,k]}$$

⇒ can't go below B_{min} in ANY SLOT

5) $$\boxed{B(i+1) = \min\{B_{max}, B(i) + \eta \cdot p_s^+(i) - p_c^-(i)\} \ \ \forall i \in [1,k]}$$

- tells how battery level at end of each slot is related to battery level at the beginning of the slot

⇒ min makes the problem not exactly linear (min isn't a linear behaviour)
⇒ but if BATTERY LARGE ENOUGH and OPERATING IN INTERMEDIATE RANGE OF THE BATTERY, most likely will not reach the maximum and not go below minimum
⇒ in practice operating in range of the battery makes the minimum not necessary
⇒ can have minimum to make constraints linear, but problem still complex

⇒ dealing with an INTEGER-LINEAR PROGRAMMING PROBLEM ⇒ solution NP hard (*)
⇒ if want to reach the optimum ⇒ exponential algorithm
(maybe exist a polynomial, but don't know)

KNAPSACK PROBLEM like

- knapsack can have pseudo-polinomial solutions: complexity of the problem is based to the size of the problem, sort n, n is size, complexity is $O(n \log n)$ in term of items to sort
- PARAMETER OF THE PROBLEM: if you measure the size of the problem to another parameter, which grows exponentially, not linearly => from POV of that param, solution is polinomial

=> take NP-hard problem, knapsack with n-elements, analize problem according to another param which is $\log_2 n$ => solution will be polinomial to respect to that parameter (but problem remains NP-hard)

A task-based model

optimization problem

- … the problem is NP-Hard
  - It can be reduced to a version of knapsack
- However, there exists a pseudo-polynomial solution based on dynamic programming
  - under some realistic restrictions it can run even in low-power IoT platforms

→ to make this problem tractable even in Arduino

• EQUATIONS that are the keys to find a dynamic programming solution for the problem in pseudo-polinomial time

• recursive equations taking as parameter: - slot you want to optimize
- battery level at beginning of that slot

• you reach last slot (k), and have your residual battery level (b)
• the max utility you can achieve is: - I've already filled the previous slot, have just to decide for the last one
- in the last one have to decide among n-tasks
- optimal solution will be: the task corresponding to the max utility for which solution is admissible

A task-based model dynamic programming

1) Se sono riuscito ad ottimizzare tutti gli slot eccetto l'ultimo, nell'ultimo => easy metto il task con utilità massima per cui la soluzione è AMMISSIBILE (BASE of induction)

- System state: (i, b)
  - previous
  - i is the slot up to which the system is optimized
  - b is the corresponding battery level at the beginning of that slot
- The optimal utility $opt(i, b)$ can be found using a backward recursive rule: last slot with residual battery b

$$1)\ opt(k, b) = \max_{j=1,\dots,n} \{u_j : b + \eta \cdot p_S^+(k) - p_S^-(k) \geq B(1)\}$$

n-task

$$2)\ opt(i, b) = \max_{j=1,\dots,n} \{u_j + opt(i+1, B^j(i+1)) : B^j(i+1) \geq B_{min}\}$$

- Where in opt(k,b) results from assuming task j is assigned to slot k
- Where $B^j(i+1)$is the residual battery at the end of slot i if task T_j is assigned to that slot

PASSO RICORSIVO:
2) - sono in uno slot i
- all'inizio dello slot ho un livello di batteria b
- Per trovare la soluzione ottima: metto in questo slot il task migliore,
+ l'ottimizzazione dei rimanenti slot (recursive call)
con un livello rimanente di batteria che dipende dal task che ho messo nello slot precedente

POLINOMIAL CUT:
• DEVO: inizio con lo SLOT 1 ed ottimizzare, e devo chiedermi: QUAL'È l'ottimizzazione con un INIZIALE livello di batteria che non conosco? => DEVO RISOLVERE IL PROBLEMA PER TUTTI I BATTERY LEVEL

• EQUATIONS that are the keys to find a dynamic programming solution for the problem in pseudo-polynomial time

• recursive equations taking as parameter: - slot you want to optimize
- battery level at beginning of that slot

• you reach last slot (k), and here you've residual battery level (b)

• the max utility you can achieve is: - I've already filled the previous slot, have just to decide for the last one
- in the last one have to decide among n-tasks
- optimal solution will be: the task corresponding to the max utility for which solution is admissible

• System state: (i, b)

but I'm ADC → discrete

• (battery level) MEANS FUL

• in order to solve this problem, if I'm already at the last slot, have to maximize the utility among all possible tasks by FORCING the admissibility for the last thing

• is the initial battery level B + what produce − what you consume should be above B(1)

- if we are able to optimize all of the slots up to the last one, this equation will tell how to find the optimum just choosing the right task for the last one (Base of induction)

$$opt(k,b) = \max_{j=1,\dots,n} \{u_j : b + \eta \cdot p_s^+(k) - p_s^-(k) \geq B(1)\}$$

$$opt(i,b) = \max_{j=1,\dots,n} \{u_j + opt(i+1, B^j(i+1)) : B^j(i+1) \geq B_{min}\}$$

• Recursive step: say I'm in an arbitrary slot, and begin the slot with battery level B, how find optimum solution?

⇒ if I'm in slot (i) and battery level is B the optimum is given by: the best task you can put in that slot + optimization of the remaining one (RECURSIVE CALL) with remaining battery depending on the task we pick-up for our slot.

⇒ if I'm in slot (i) and initial battery level is B, in slot (i) i decide to test task 1, try task 1 with utility 1 in slot i, if I use task 1, residual battery level will be: B + what expect to produce − cost of task 1 and optimum of the utility will be given by: u_1 + recursive call for optimization of remaining tasks

• with this rule can write a pseudo-polynomial algorithm : cuz if I'm beginning the optimization, have to start with slot 1, and for slot 1 have to ask to myself: what's the optimization of slot 1 with an initial battery level which I don't know?

=> to solve this, have to solve the problem for all possible battery level and executing this for all the battery levels, this equation will be the optimum (but have to solve to all battery level)

• the THING IS: the battery levels are ∞ (continuous value) => so will be ∞ time but battery level are measured by an ADC => DISCRETE BATTERY LEVEL

(Arduino ADC in 10 bits, battery levels are at most 1'000, considering that some of those battery levels corresponds to B level below the minimum, battery levels meaningful are 300/500

A task-based model

dynamic programming

- The time complexity is $O(k \cdot Battery\ levels)$
  - we can disregard the number of tasks that is a small number
- The memory complexity (with some optimizations) is $O(Battery\ levels)$
- Hence the complexity depends on the number of slots and on the battery levels…
- … but the battery level is a **real number** ranging from B_{min} to B_{max}… is it?

=> # slots

& => # of task = 24 in Arduino

=> Complexity will be of same thing (300)

TRACTABLE

Just exploiting the limitation of the HW

NB: reducing battery levels, problem becomes tractable but Battery in reality have continuous values, in discrete level cuz just can't measure it

=> APPROXIMATION

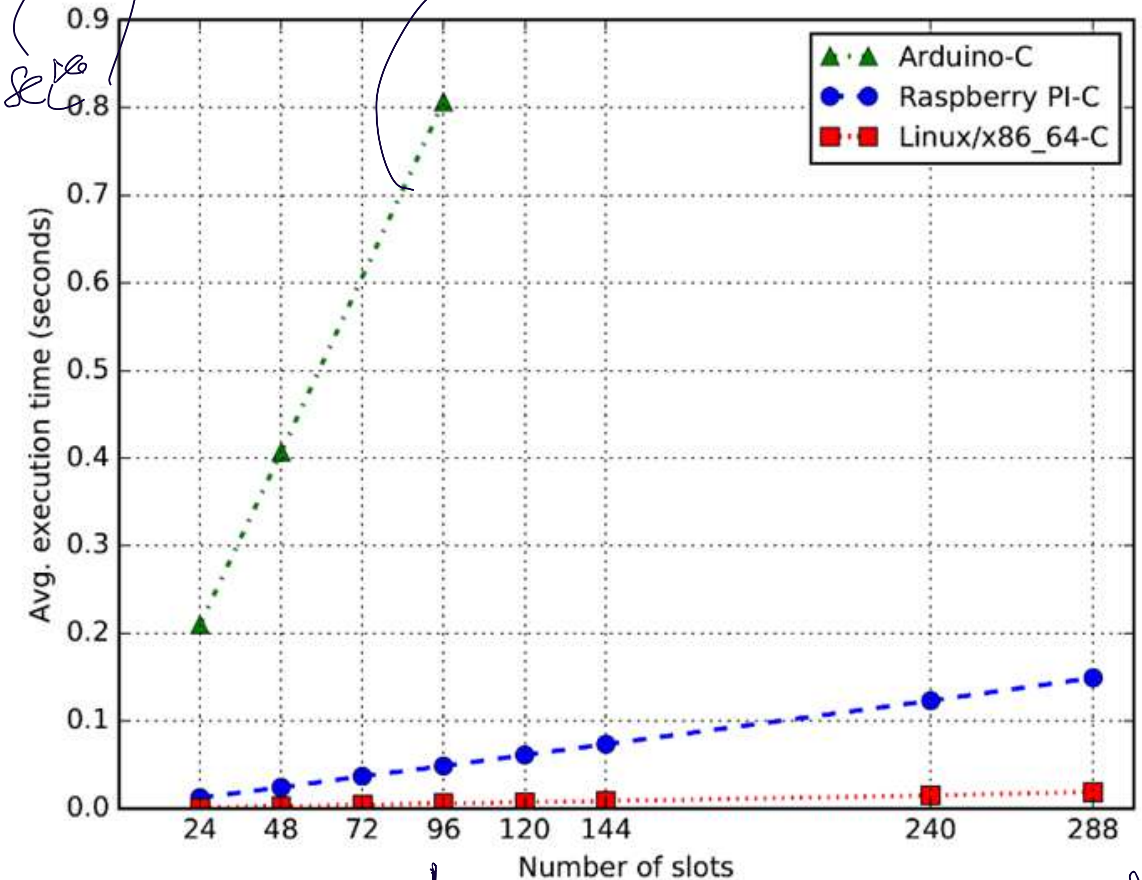
=> discretize battery level (continuous) cuz of ADC

→ 0,8 sec to implement the optimization of 96 slots ⇒ affordable

→ to implement

→ optimize with 96 slots

sec

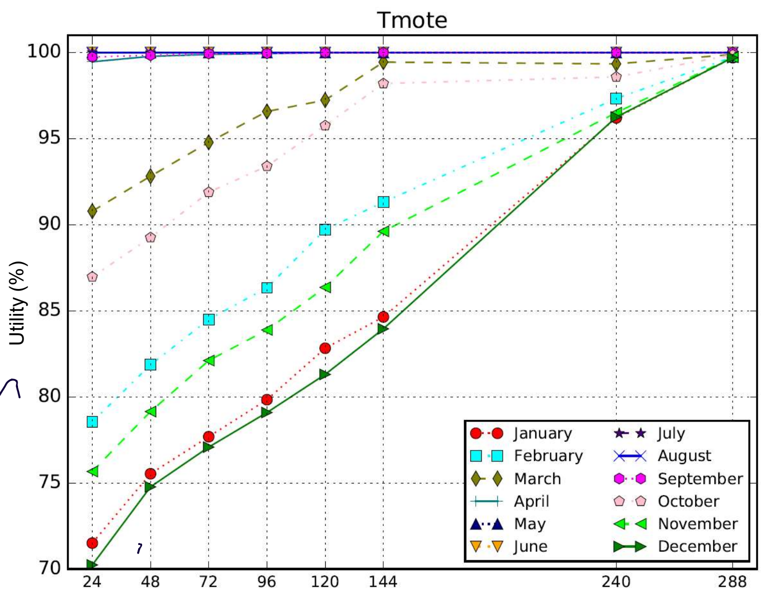


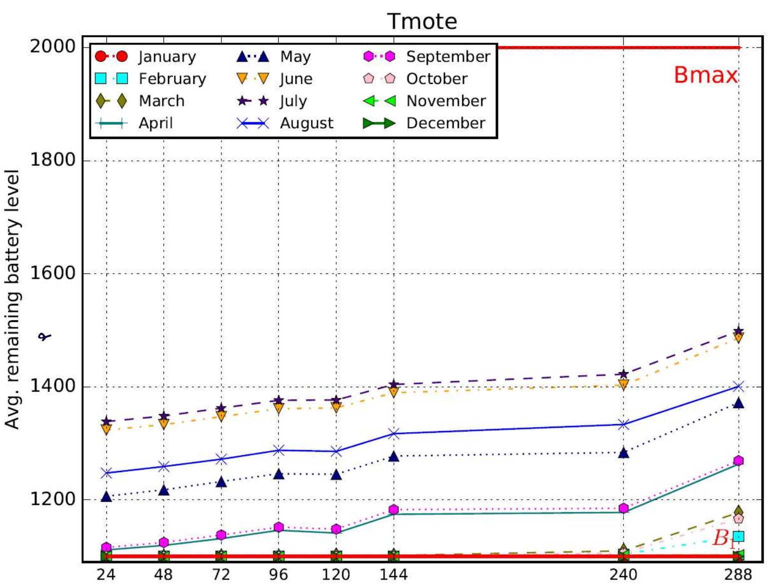
Execution times ⇒ this

- the complexity remains $O(k \cdot Battery\ levels)$
- … but we can choose the number of slots and the battery level to keep feasibility for the target platform.
- Execution time of C code:
  - Platforms: Arduino uno, Raspberry PI, PC Linux
  - Battery max charge: 2000 mAh
  - Operative range: 4.2 V – 3.4 V
  - ADC: 10 bits
  - Battery levels: from 828 to 1024

15 minutes slots, after that, out of memory

- TO ~~REACH~~ perform OPTIMUM, you don't have to run many, many tasks
  ⇒ if have length of tasks, more the opt ⇒ reasonable one with 3 tasks





acc. to period

=> achieve # slots ...

Simulations

slots => granular decision

=> match real fuel

batt, pred =>

Slots

- utility ... # of slots in # months
- utility ↑ if you take larger # slots or with bigger # slots you take ...

- simulated applications with different duty cycles, from 2% to 46% (and corresponding utilities)

  granular decisions, so you're able to catch better the real behaviour of battery and pred

- over Arduino uno, Tmote and Raspberry PI platforms
- battery of 2000 mAh and solar panel KL-SUN3W
- number of slots from 24 to 288

  . few slots force you to approximate ⇓ results in "worst optimization"

- The figures report the results for Tmote:
  - Utility
  - Average remaining battery level

which is better in summer cuz is quicker larger production to catch even with larger slots

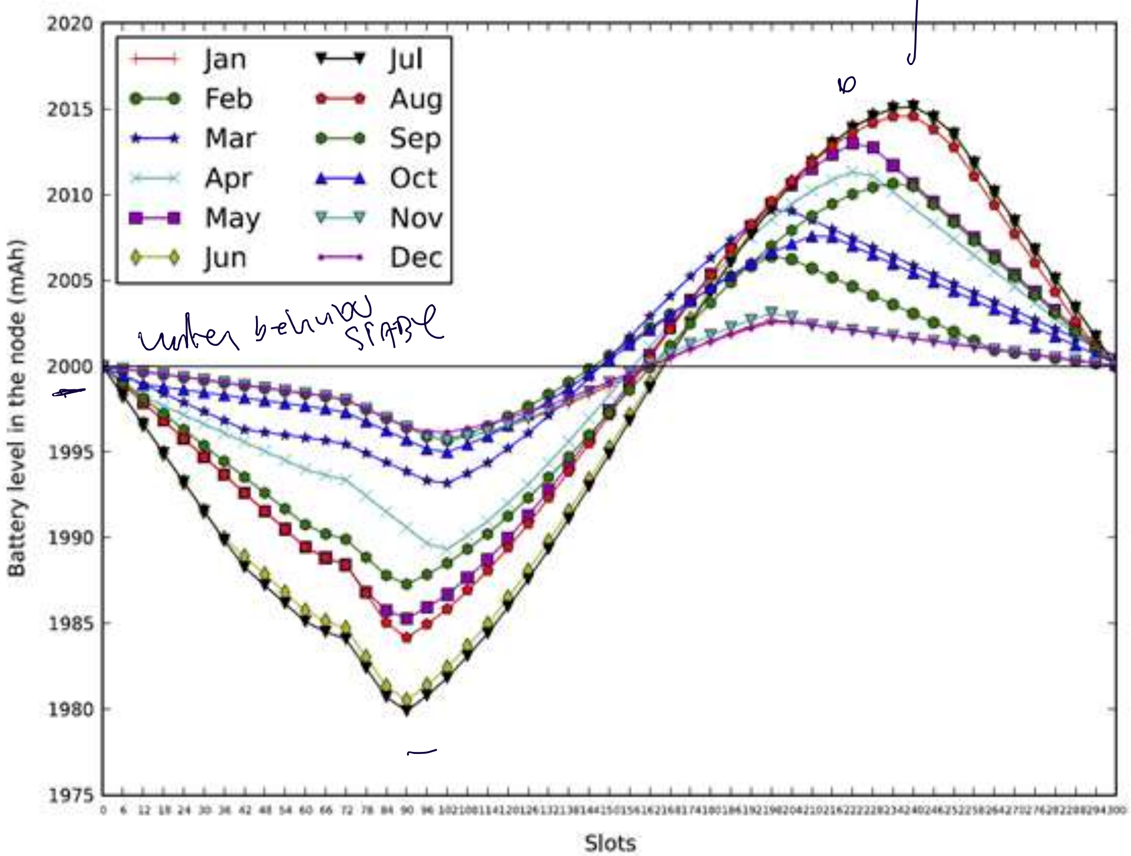
- quality of opt depends on # of slots

  can make more refined for solution

in summer you can push out more production, so less on the battery

Simulations

opt dev. in ≠ month



Battery level along the day, in different months

References

- Aman Kansal, Jason Hsu, Sadaf Zahedi, "Power management in energy harvesting sensor networks", *ACM Transactions on Embedded Computing Systems*, 6(4), September 2007. https://doi.org/10.1145/1274858.1274870
- A. Caruso, S. Chessa, S. Escolar, X. del Toro, J. C. Carlos López, “A Dynamic Programming Algorithm for High-Level Task Scheduling in Energy Harvesting IoT”, *IEEE Internet of Things Journal*, 5(3):2234-2248 (2018)

Summary

Energy harvesting IoT architectures → Characteristics of the energy sources → Characteristics of batteries and measurement of the charge

→ Energy neutrality → Kansal's model for energy neutrality → A task-based model for the utility and modulation of the load

→ A dynamic programming scheduler to achieve energy neutrality

A task - based model dev

Exercise E-2

The table reports, for each slot in a day, the expected energy production of an energy-harvesting IoT device.
Assuming that:

- $dc_{max} = 90\%; p_{max} = 5mA\ ; u(dc_{max}) = 100;$
- $dc_{min} = 50\%; p_{min} = 2mA; u(dc_{min}) = 10;$

Complete the table by assigning a duty cycle, an utility, and a power consumption to each slot, according to the Kansal's algorithm

Slot	1	2	3	4	5	6	7	8	9	10	11	12
$\tilde{p}_s(i)$	0	0	0	3	5	7	4	3	2	0	0	0
dc_i												
$u(\cdot)$												
$p(\cdot)$												

Exercise E-2

The table reports, for each slot in a day, the expected energy production of an energy-harvesting IoT device.
Assuming that:

- $dc_{max} = 90\%; p_{max} = 5mA\ ; u(dc_{max}) = 100;$
- $dc_{min} = 50\%; p_{min} = 2mA; u(dc_{min}) = 10;$

Complete the table by assigning a duty cycle, an utility, and a power consumption to each slot, according to the Kansal's algorithm

Slot	1	2	3	4	5	6	7	8	9	10	11	12
$\tilde{p}_s(i)$	0	0	0	3	5	7	4	3	2	0	0	0
dc_i												
$u(\cdot)$												
$p(\cdot)$												

ES. 1: Following table report expected energy production, for each day

. Assuming that:

maximum consumption

- $dc_{max} = 90\%$ $\quad P_{max} = 5\,mA$ $\quad u(dc_{max}) = 100$
- $dc_{min} = 50\%$ $\quad P_{min} = 1\,mA$ $\quad u(dc_{min}) = 10$

, compute DC, utility, power consumption to each slot (Kansal's Algo)

SLOT	1	2	3	4	5	6	7	8	9	10	11	12
$\tilde{P}_s(i)$	0	0	0	2	6	11	10	7	3	0	0	0
$dc(i)$	50	50	50	50	90	90	90	90	50	50	50	50
$u(\cdot)$	10	10	10	10	100	100	100	100	10	10	10	10
$p(\cdot)$	1	1	1	1	5	5	5	5	1	1	1	1

1) Assegno S max dc / D min

1) IDENTIFICO SUN SLOTS & DARK SLOTS & ASSEGNO MAX DC & MIN DC

SUN SLOTS ⇒ expected production > maximum consumption 5 mA

DARK SLOTS ⇒ expected production < maximum consumption 5 mA

2) Adesso verifico se sono in UNDER/OVER PRODUCTION ⇒ Se nello slot 1 ho DC = 50

⇒ consumption 1 mA

se in 5 ⇒ consumption 5 mA

TOTAL production & TOTAL CONSUMPTION CON L'ASSEGNAMENTO INIZIALE:

⇓ 39 mA

⇓ (8 D slots · Pmin: 8 · 1mA = 8mA) + (4 Sunslots · Pmax: 4 · 5mA = 20mA)

⇒ 8 mA + 20 mA = 28 mA

Prod − Comp: = 39 mA − 28 mA = 11 mA ⇒ AMMISSIBLE BUT, NOT OPTIMUM

⇒ OVER PRODUCTION ✓

Δ uso per tirare su i Dark slot

CASO 1 : overprod

⇒ aumento Dark slots up to max utility

P_r = 11 mA

SLOT	1	2	3	4	5	6	7	8	9	10	11	12
$\tilde{P}_s(i)$	0	0	0	2	6	11	10	7	3	0	0	0
dc(i)	90	90	50	50	90	90	90	90	50	50	50	50
U(·)	100	100	10	10	100	100	100	100	10	10	10	10
P(·)	5	5	1	1	5	5	5	5	1	1	1	1

⇓ 3

5mA − 1mA = 4mA da tirare su 1 Dark Slot

⇒ 1/2 up

P2 = 3 ⇒ non posso andare al max utility nel 3°

più forma 3 ⇐

Adesso lo sbt 3 lo porto a (4) => devo trovare il DC & utility correlato

$$p(DC) = \frac{P_{max} - P_{min}}{dc_{max} - dc_{min}} \cdot dC - P_{max} - P_{min}$$

$$u(DC) = 2{,}75 \cdot dC - 102{,}5$$

SbT	1	2	3	4	5	6	7	8	9	10	11	12
$\tilde{P}_s(i)$	0	0	0	2	6	11	10	7	3	0	0	0
$dc(i)$	90	70	80	50	90	90	90	90	50	50	50	50
$u(\cdot)$	100	100	77,5	10	100	100	100	100	10	10	10	10
$p(\cdot)$	5	5	4	1	5	5	5	5	1	1	1	1

4

3

SGT	1	2	3	4	5	6	7	8	9	10	11	12
$\tilde{P}_s(i)$												
$dc(i)$												
$u(\cdot)$												
$p(\cdot)$	1											