RETI DI SENSORI

OBIETTIVI DEI PROTOCOLLI MAC

I protocolli MAC cercano il più possibile di ottimizzare i seguenti obiettivi, con lo scopo di migliorare al massimo il trasferimento:

- **Collision Avoidance** (come abbiamo già visto nelle reti Wireless)
- Latenza (alta)
- Largezza di banda
- **Scalabilità** (adattare le dimensioni a seconda delle situazioni)
- **Efficienza Energetica (consumo)**

CONSUMO ENERGETICO

A proposito dell'ultimo aspetto, si scopre facilmente che le sorgenti di *maggiore spreco di energia* sono:

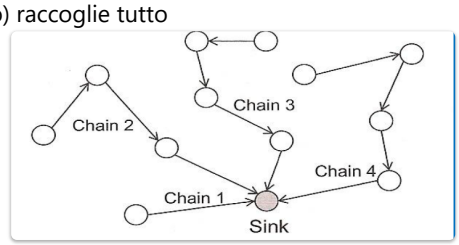
- Il trasmettitore
- Il ricevitore
  I protocolli MAC si impegnano a ridurre quanto possibile questi consumi
- Per questa ragione sono state sviluppate delle tecniche, tra cui le seguenti (dette di $net\ processing$)

DATA AGGREGATION

- È un processo che combina informazioni multiple in un singolo messaggio che sarà trasmesso e che quindi rappresenta un "riassunto" sintetico delle informazioni:

  $$m_{\mathrm{A}} = aggregate(m_0, m_1, \ldots, m_n)$$

  In questo modo basta un unico messaggio, e quindi il tutto sarà più corto
  Si cerca di aggregare tutte le informazioni prima del routing, in questo modo l'instradamento è più veloce e meno pesante per i dispositivi di rete
- Un dispositivo detto $sink$ (logica tipo lavandino) raccoglie tutto



> *Rappresentazione a catena ↑ ; esiste anche quella ad albero, a griglia*

DATA FUSION

- È una tecnica che consiste nell'annotare sulle informazioni ricevute alcune informazioni aggiuntive (metadati):

  $$m_{\mathrm{F}} = fusion(m_0, m_1, \ldots, m_n)$$

  Ad esempio: aggiungere un time stamp o la localizzazione (pensiamo a una foto - ha un orario di scatto e magari anche il luogo o il nome della macchina fotografica)

CLUSTERING

Permette di suddividere un insieme di dispositivi secondo un certo criterio

- Particolarmente utile per reti WSN (Wireless con sensori) perché migliorano l'inoltro dei messaggi in rete, limitando collisioni

LEACH

È una particolare tecnica utile per massimizzare il $life\ time$ dei dispositivi in rete, permettendo di distribuire bene la potenza relativa a un dispositivo (sensore) → consumo equo

😃: implementare nei nodi scelti tutte le funzioni di $net\ processing$, come ad esempio la $data\ aggregation$

- In particolare, allo scopo proprio di implementare la data aggregation, sarà necessario scegliere un *clusterhead*, ovvero un nodo di riferimento a cui arrivano tutte le informazioni dei client vicini - è una interfaccia verso il nodo $sink$
  - Naturalmente si sceglierà un dispositivo che permette quante più connessioni (radio) possibile - eventualmente un nodo che nel suo range radio include tutti gli altri dispositivi
  - Tale nodo esegue talvolta una rielaborazione delle informazioni (data aggregation) e poi sarà in grado di trasferire tutto verso un nodo

centrale della rete
- Ogni tot tempo si cambia clusterhead (rotazione di ruolo) per non sovraccaricare troppo
😃: solo il capo cluster accede alla rete (meno congestione) e se riesce a eseguire anche una compressione delle informazioni che gli arrivano, c'è anche meno pressione sulla rete
😃: si evita che tutti dialoghino con il nodo centrale consumando energia

Il processo di Leach è il seguente:
Ipotesi: esiste un unico sink; tutti i nodi possono comunicare verso il sink (che sia in modo aggregata o individuale)

Fasi

1. Si definisce la probabilità p che ciascun nodo ha di essere elettro $clusterhead$
2. Si esegue una "elezione cooperativa": ogni sensore tira a sorte un numero casuale $x \in [0;1]$ con probabilità uniforme
3. Si definisce un valore di soglia valido per ogni sensore con la seguente regola: se il nodo appartiene all'insieme G che rappresenta l'insieme dei nodi che ancora non hanno avuto "l'onore" di essere stati eletti clusterhead allora ha un certo valore (vedi dopo); altrimenti la soglia vale zero (una volta che è stato rieletto non può essere rieletto). In formule:

$$T(n) = \begin{cases} \frac{p}{1-p\, r \mod \left(\frac{1}{p}\right)} & \text{se il nodo } \in G \\ 0 & \text{altrimenti} \end{cases}$$

   - dove r è il numero di round di elezione
   - $\frac{1}{p}$ indica il numero di nodi che negli ultimi $\frac{1}{p}$ round non sono stati eletti. Quindi in generale meno sensori abbiamo e più è alta la probabilità di scelta di un certo nodo
4. si confrontano quindi i vari $T(n)$ e si candidano ufficialmente tutti i nodi che hanno $x < T(n)$
5. tra tutti i candidati se ne sceglie uno (magari quelli con più elementi aggregati)

😡: elegge il clusterhead semplicemente tenendo conto della frequenza di elezione (round), senza preoccuparsi che chi viene eletto poi ha effettivamente le capacità di svolgere quel ruolo (esempio: sufficiente potenza). 😃 È stato proposta la tecnica HEED per risolvere questa "falla"

HEED

Tecnica evolutiva (migliorata) del Leach

- Si considera con attenzione il consumo in termini di energia di ciascun nodo

Tre Fasi

- Inizializzazione
- Selezione del clusterhead
- Setup delle reti di clustering

$Nota$: per ogni nodo si definisce la probabilità che esso possa diventare $clusterhead$ con la seguente regola:

$$CH_{\text{prob}} = C_{\text{prob}} \cdot \underbrace{\frac{E_{\text{res}}}{E_{\max}}}_{\text{novità}}$$

- C_{prob} è la probabilità che un sensore può essere eletto
- $\frac{E_{\text{res}}}{E_{\max}}$ è il rapporto tra le energie residue di un nodo (numeratore) con il valore di massimo di carica che il sensore ha (in generale è uguale per tutti i nodi).

ALGORITMI DI ROUTING PER RETI DI SENSORI

FLOODING

Tecnica più basilare - introdotta perché (soprattutto in passato con IPv4) non tutti i sensori avevano il proprio indirizzo IP pubblico (quindi non si possono indirizzare in modo univoco ma si devono utilizzare modalità diverse)

- Idea: un nodo un messaggio che deve trasferire. Questo lo fa ripetendo il messaggio su tutti i suoi vicini salvo quello da cui lo ha ricevuto
  - 😡: Ripetuta su tutti i nodi della rete tuttavia, un certo nodo può ricevere lo stesso messaggio da più parti → causa inondazione della rete (*congestione*) → alto consumo

Esistono diverse varianti:

- Quella più semplice, consiste nel *etichettare un messaggio* in modo tale che esso sarà poi riconosciuto in rete, evitando troppe inutili ripetizioni
  - La $label$ che si attacca è il TTL (Time To Live): si associa al messaggio un numero massimo consentito di ripetizioni sulla rete. Ogni volta che un messaggio arriva a un nodo, il contatore TTL viene diminuito di 1. Quando $\mathrm{TTL} = 0$, il nodo che legge questo dato non lo inoltra più
  - Non si elimina, ma comunque si limitano le congestioni

GOSSIPING

Altra tecnica: include un criterio *statistico* nella scelta se ripetere o meno un messaggio

- Ogni volta che un messaggio (pacchetto) arriva a un nodo:
  - Con probabilità p si decide di ripeterlo in tutte le porte tranne quella da cui è arrivato
  - Con probabilità $1 - p$ si decide di non ripeterlo

> *Possiamo anche unire le varie tecniche*

SEMANTIC ROUTING

Fin ora abbiamo considerato il $routing$ (instradamento) dei messaggi senza conoscerne il significato di questi ultimi.

- Le reti di sensori si prestano bene ad applicare un *nuovo paradigma: $semantic\ routing$*, che consiste nel considerare (anche) il significato e l'importanza che la conoscenza di un messaggio porta al processo
- Tecniche più avanzate: il routing non consiste solo nell'instradamento ma anche nella conoscenza in dettaglio delle informazioni che circolano

DIRECT DIFFUSION

- Prima tecnica (di due) che vediamo
  Esiste un nodo centrale (un $sink$) che ha il privilegio di poter interrogare i propri $client$ (sensori) per chiedergli delle informazioni. Le fasi (3) sono le seguenti (stile handshaking)

Interest & Gradient:

- Parte con una richiesta di interesse (il $sink$) invia una richiesta d'informazioni ai client specificando alcuni attributi che si vuole (esempio: una certa qualità/tolleranza etc...) - viene mandata in floating (tipo broadcast) con un certo TTL
- Chi è in grado di poter soddisfare quanto richiesto risponde con un $gradient$: dà una risposta positiva e specifica anche il valore aggiunto che può dare (esempio se era stato chiesto: qualità 100, il nodo risponde con: non solo ti posso dare 100, ma ti posso dare anche 130) - contrattazione
  Reinforcement:
- Il $sink$ esamina tutte le offerte di contrattazione ricevute e ne sceglie una (in base a un certo criterio) e avverte tutti che ha scelto quel determinato sensore inviando un messaggio visibile a tutti ma che solo chi ha formulato quella offerta capisce di essere stato scelto
  Data Propagation:
- Il nodo scelto trasferisce i dati

SPIN

$Sensor\ Protocol\ for\ Information\ via\ Negotiation$

- Altra tecnica che usa tre tipi di messaggi: $\mathrm{ADV},\ \mathrm{REQ},\ \mathrm{DATA}$

Il $sink$, o più in generale un sensore annuncia una informazione che ritiene utile per la comunità

- Cioè si propone come fornitore di questa informazione
  - Attraverso un messaggio detto ADV (Advertisement) $\rightarrow$ è un $preview$ di quello che potrebbe essere servito
    - Viene inviato a tutti i nodi entro il raggio radio
    - Non si invia tutto perché magari non interessa poi
- I nodi destinatari esaminano il contenuto e se gli è di interesse rispondono con un messaggio di REQ, cioè di esplicito invio del contenuto ($request$)
- Il $sink$ (o comunque il nodo sorgente) invia a chi ha fatto la REQ il contenuto, con un messaggio di DATA

> *Si è risparmiato sull'invio: solo ciò che è di interesse viene posto in rete*

😃: l'informazione si propaga bene nella rete - un destinatario può diventare sorgente per altri dispositivi
😡: problema intrinseco - se una informazione particolarmente utile viene trasmessa ma non trattenuta da nessun dispositivo del "primo" raggio radio, chi ne è fuori che magari avrebbe interesse non la vedrà mai.

Esempio: $A \rightarrow B \rightarrow C$ - magari C ha interesse dell'informaizione in A, ma essendo fuori range, se B non la cattura "facendo da ponte", C non riceverà mai l'informazione

$Nota$: questa era la F19/F20 - da qui si passa a IoT