Azzolini Riccardo 2020-11-26

Protocolli di routing per wireless sensor network

1 Routing nelle wireless sensor network

I protocolli/algoritmi di routing tradizionali non sono adeguati all'uso nelle wireless sensor network, perché:

- le WSN sono molto dense, hanno un elevato numero di nodi;
- ci sono frequenti cambiamenti topologici;
- i nodi hanno capacità computazionale, memoria e risorse energetiche limitate;
- i nodi non hanno un identificativo globale (come un indirizzo IP);
- è necessaria l'integrazione con il compito di acquisizione dei dati che ciascun sensore svolge come sua funzione primaria.

Sono allora stati definiti numerosi protocolli di routing appositi per le WSN. Essi si suddividono in tre categorie:

- protocolli **data centric** (*flooding*, *gossiping*, *SPIN*, *directed diffusion* e altri), che si focalizzano sui dati scambiati tra i nodi piuttosto che sull'identità dei nodi stessi;
- protocolli **gerarchici** (come ad esempio il *LEACH*);
- protocolli **location based**, che (come indica il nome) si basano sulla posizione fisica dei sensor nodes.

Il protocollo di routing ideale dovrebbe:

- selezionare sempre i percorsi più corti per ogni dato trasmesso;
- evitare che lo stesso dato venga inutilmente inviato più volte a uno stesso nodo;
- minimizzare il consumo di energia;
- sfruttare una conoscenza globale della topologia della rete.

2 Flooding

Il più semplice protocollo di routing è il **flooding**: ogni dato ricevuto da un nodo viene inoltrato in broadcast a tutti i nodi vicini.

A parte la semplicità, il vantaggio di quest'approccio è che, siccome ciascun dato passa sempre da tutti i possibili percorsi, è garantito che passi anche da quello più breve verso la sink, quindi si ha il ritardo minimo possibile. D'altro canto, ci sono numerosi problemi:

- **Implosion**: si ha un elevato overhead, un'"invasione" della rete, che rischia addirittura di mandarla in congestione.
- **Data overlap**: ogni nodo può ricevere più volte lo stesso dato, perché questo viene trasmesso da diversi suoi vicini. Tali trasmissioni ridondanti non fanno altro che sprecare energia.
- **Resource blindness**: viene fatto un uso "cieco" delle risorse disponibili, senza cercare di sfruttarle in modo ottimale. Ciò comporta un uso inefficiente della potenza/energia disponibile.

3 Gossiping

Il **gossiping** prevede che ogni nodo inoltri i dati ricevuti solo a un suo vicino scelto casualmente, invece di inviarli in broadcast a tutti. Così facendo, si risparmia energia e si evitano implosion e overlap, ma non è più garantito che i dati giungano alla sink lungo il percorso più breve, quindi si hanno ritardi maggiori rispetto al flooding.

4 SPIN

Con **SPIN**, **Sensor Protocol for Information via Negotiation**, i sensor nodes si scambiano informazioni sui dati di cui sono già in possesso e su quelli che desiderano avere, in modo da evitare trasmissioni superflue e diminuire quindi i consumi di energia.

SPIN usa tre tipi di messaggi, secondo un protocollo chiamato **3-stage handshake**:

1. quando un sensore ha nuovi dati da trasmettere, invia in broadcast un pacchetto di **advertisement** (**ADV**) per informare gli altri nodi;
2. in risposta, i nodi interessati ai dati inviano un pacchetto di **request** (**REQ**);
3. infine, il sensore trasmette un pacchetto di tipo **DATA**, contenente i dati veri e propri, ai soli nodi che li hanno richiesti.

Questo meccanismo è semplice, ed evita implosion e overlap senza introdurre ritardi eccessivi, ma introduce un overhead (dovuto ai messaggi di advertisement e request) che comporta un certo consumo di potenza.

Esiste anche una versione migliorata di SPIN, chiamata SPIN-2, che si comporta sostanzialmente come SPIN-1 (la versione originale), ma in più introduce un **low-energy threshold**: quando l'energia a disposizione di un nodo scende sotto una determinata soglia (stabilita in modo empirico, mediante analisi del comportamento della rete), esso passa a uno stato **DORMANT**, nel quale riduce la propria partecipazione al protocollo per conservare l'energia propria e degli altri nodi. In particolare, un nodo in questo stato partecipa a un 3-stage handshake solo se ritiene di avere energia sufficiente per poter completare tutte le fasi rimanenti. Altrimenti, potrebbe ad esempio capitare che un nodo richieda un dato ma, dopo averlo ricevuto, non abbia più energia sufficiente a inoltrarlo: in tal caso, l'energia consumata per trasmettere il dato a questo nodo risulterebbe sprecata.

Con SPIN-1 si ha un risparmio di energia del 70 % rispetto al flooding, e con SPIN-2 si risparmia un ulteriore 10 % rispetto a SPIN-1.

5 Directed diffusion

L'algoritmo di routing **direct diffusion** è sostanzialmente il "duale" dello SPIN: la sink dichiara in broadcast il proprio interesse a determinati tipi di dati, mediante un meccanismo di **interest propagation**, e rispondono solo i nodi che sono in possesso di tali dati.

L'identificazione dei dati da diffondere avviene tramite un **naming scheme**: i dati generati dai sensori sono rappresentati sotto forma di coppie attributo-valore. Per ottenere i dati, la sink interroga i sensori mediante delle query, che prendono il nome di **interest**. Ciascuna query è una lista di coppie attributo-valore che descrive i criteri di ricerca dei dati.

Ad esempio, in un'applicazione di tracciamento di animali, la sink potrebbe inviare la query

$\mathit{Tipo} = \text{quadrupede}$	(animali di cui rilevare la posizione)
$\mathit{Intervallo} = 20\,\text{ms}$	(invia eventi ogni 20 ms)
$\mathit{Durata} = 10\,\text{s}$	(per i prossimi 10 s)
$\mathit{Regione} = [-100, 100, 200, 400]$	(da sensori all'interno di questo rettangolo)

Se un sensore che rilevasse un animale che soddisfi i criteri specificati nella query, potrebbe generare in risposta i seguenti dati,

$Tipo = \text{quadrupede}$	(tipo di animale rilevato)
$Istanza = \text{elefante}$	(istanza del tipo di animale)
$Posizione = (125, 220)$	(posizione del nodo)
$Intensità = 0.6$	(misura dell'intensità del segnale)
$Timestamp = 01{:}20{:}40$	(tempo di generazione dell'evento)

e inviarli alla sink (sapendo che è interessata).

I vantaggi di questo algoritmo di routing sono che:

- essendo data centric, non necessita di uno schema di indirizzamento;
- ogni nodo può fare aggregazione e caching;
- si ha uno sfruttamento efficiente dell'energia, poiché i dati vengono inviati solo su richiesta (on demand);
- non è necessario conoscere la topologia globale della rete.

Ci sono però anche degli svantaggi:

- il modello di consegna dei dati basato sulle query non è una buona scelta per applicazioni che necessitano di un flusso di dati continuo (come ad esempio il monitoraggio ambientale);
- i naming scheme sono dipendenti dall'applicazione;
- il processo di matching tra dati e interrogazioni introduce un certo overhead.

6 LEACH

LEACH, **Low Energy Adaptive Clustering Hierarchy**, è un protocollo di routing gerarchico basato su *clustering*, che minimizza la dissipazione di energia. L'idea è di raggruppare i sensor nodes in dei **cluster**, eleggendo casualmente alcuni sensori come **cluster head**, che fungono da router (relay), e associando a essi gli altri sensori in base alla forza dei segnali ricevuti.

LEACH funziona in due fasi: la **set-up phase** e la **steady state phase**.

1. Nella set-up phase, ogni sensor node ha una certa probabilità di eleggere se stesso come cluster head. In particolare, esso estrae un numero casuale tra 0 e 1; se questo è minore di una determinata soglia (fissata nella progettazione della rete), il nodo diventa un cluster head.

Ogni nodo che diventa cluster head invia in broadcast un messaggio di **advertisement** per informare gli altri nodi. Questi ultimi scelgono il cluster head a cui associarsi in base alla forza dei segnali di advertisement ricevuti (se il segnale ricevuto è più forte, significa che serve meno energia per la comunicazione), e inviano a esso una richiesta di associazione, formando così un cluster. Infine, ciascun cluster head assegna gli intervalli di tempo entro cui i vari nodi del proprio cluster possono trasmettere.

2. Nella steady state phase, i sensori rilevano i dati (sense) e li trasmettono ai cluster head. Questi ultimi aggregano i dati provenienti dai nodi nei loro cluster e li inoltrano verso la sink.

Dato il loro ruolo centrale nella comunicazione, i cluster head consumano più energia dei nodi "normali", quindi prima o poi si scaricherebbero, compromettendo il funzionamento della rete. Per evitare ciò, LEACH implementa un *clustering dinamico*: dopo un certo periodo trascorso nello steady state, la rete ritorna nella set-up phase per selezionare nuovi cluster head. I nodi che erano head nel "turno" precedente non partecipano a questa selezione, per evitare di consumare troppa energia.

Stabilire il numero ottimale di cluster non è immediato:

- se ci fossero troppi pochi cluster, allora ciascuno di questi sarebbe piuttosto grande, dunque i nodi sarebbero in media lontani dai cluster head, e consumerebbero più energia per comunicare;
- se invece ci fossero troppi cluster, esisterebbero molti nodi (cluster head) che inviano dati alla sink, e così si perderebbe in gran parte il vantaggio di un'organizzazione gerarchica (nella quale la maggior parte dei nodi devono comunicare solo a breve distanza, all'interno del loro cluster).

I vantaggi del protocollo LEACH sono i seguenti:

- si ha un risparmio energetico rispetto alle comunicazioni dirette;
- siccome i nodi possono morire casualmente, il clustering dinamico incrementa la vita del sistema;
- è un protocollo completamente distribuito, che non richiede una conoscenza globale della rete;
- adotta un *single hop routing*, cioè ogni nodo può trasmettere direttamente al cluster head.

Uno svantaggio, invece, è che questo protocollo non è applicabile a reti installate in ampie regioni.