Data centric and content based networking :

→ Here discussion about networking that took place directly based on content is discussed rather than routing through routing Protocols that use a direct identifier of nodes (either node id or position).

→ The content can be collected from n/wk, Processed in the n/wk, and stored in the n/wk. Therefore, this chapter focus on Content based networking and data aggregation mechanisms.

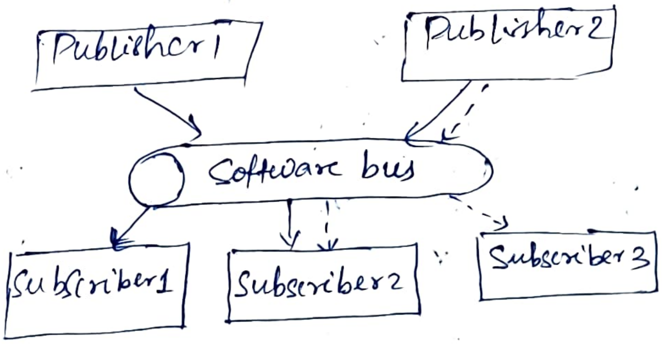
Network interaction Paradigm :

→ Standard network uses client/server, Peer-to-peer Commu^{n}.

→ for WSN it uses decoupling in space (neither sender nor receiver need to know their partner, i.e address of sender/receiver unknown). WSN uses decoupling in time, i.e answer not necessarily directed triggered by question (asynchronous comm^{n}.)

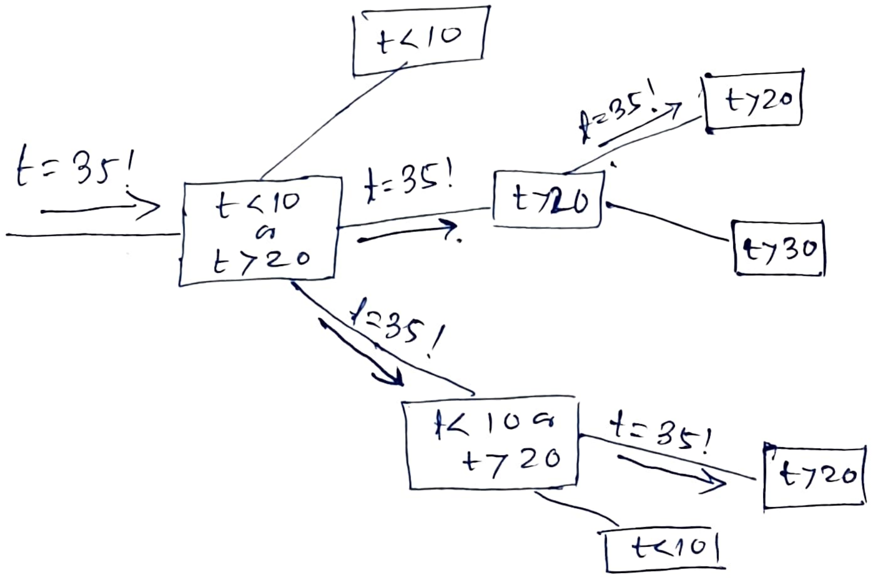
Interaction paradigm : Publish/subscribe :

→ idea : Entities can publish data under certain names.
- entities can subscribe to updates of such named data.
- conceptually implemented by a software bus. Software bus stores subscriptions, Published data names used as filters, subscribes notified when values of named data changes.



- Variations :
- Topic-based P/S - Inflexible
- Content-based P/S - Use general Predictions over named data.

Publish/subscribe implementation options
- Central server (Mostly not applicable)
- TOpic-based P/S : Group comm^{n}. Protocols.
- Needs Content-based routing / frowarding for efficient networking.



Data centric routeing:

One-shot interactions with big data sets;

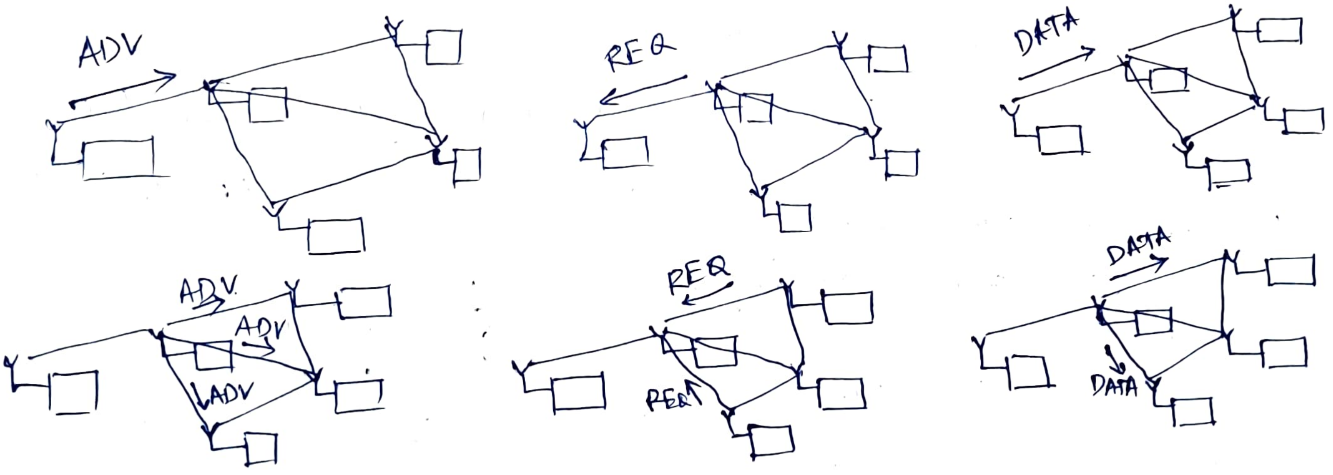
Scenario: large amount of data to be communicated, eg: video Picture

Idea is to exchange characterization with neighbor, to ask whether it is (interested in data.)

-> It only transmit data when explicitely requested

-> Nodes should also know the interests of further away nodes.

Sensor Protocol for information Via Negotiation (SPIN)



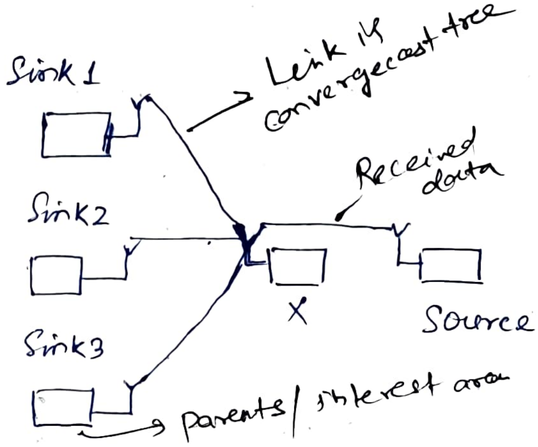
Repeated interactions:

Interesting Part is subscribe once, event happen multiple times. The question which node can provide data, if multiple nodes might ask for data, then how to map into routing Problem. The idea is to put enough information into network so that publications & subscriptions can be mapped onto each other. But unique identifiers are avoided (as content based) as it might not be available, might require too big state size in intermediate nodes.

* Directed diffusion is one option for implementatn

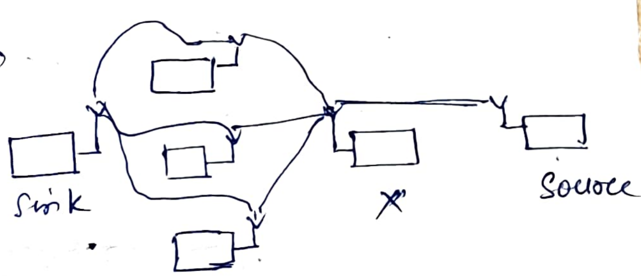
→ It try to relay ~~on~~ on local interactions for implementatn

Directed diffusion - Two-phase pull



Phase1: Nodes distribute interests in certain kinds of named data (specified ~~as~~ as attribute value pairs). The interests are flooded in the n/work.

→ Remembering from where interests came a convergence tree is set up. As shown from both figures node x cannot distinguish in absense of unique identifiers, hence it set up either only one or three convergecast trees.



Directed diffusion - Gradients in two phase pull:

Option 1: Node x forwarding received data to all "parents" in a "Convergecast tree". This is not attractive as many needless packet repetitions occur over multiple routes. (If sink n doesnot intend a particular data sent from X).

Option2: Node x only forwards to one parent. It is not acceptable as data sinks might miss events. (If all sinks desires a particular data from X)

Option 3: only provisionally send data to all parents, but ask data sinks to help in selecting which paths are redundant, which are required. (Acknowledgement from sink)

→ The information from where an interest came is called 'gradient.

→ Forward all published data along all existing gradients.

Gradient reinforcement:

→ Gradient ~~can not~~ represent not only a link in a tree but a quantified 'strength' of relationship.

→ Initialized to low values

→ Strength repoesents rate with which data is to be sent.

The intermediate nodes forward on all gradients and use a data cache to supress needless duplicates.

Second phase:

Nodes that contribute new data (not found in cache) is encouraged to send more data.

→ If sending rate is increased, the gradient is reinforced. Gradient reinforcement start from the sink. If requested rate is higher than available rate, gradient reinforcement propagates towards original data sources.

→ Gradient reinforcement adats to changes in data sources, topology, sinks.

Some extensions to directed diffusion are:

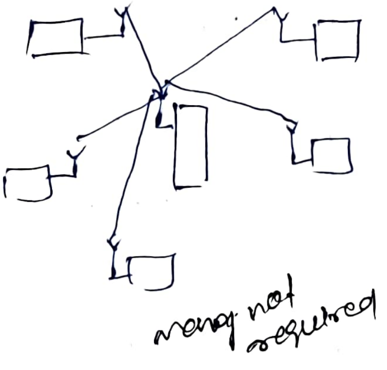
Geographic scoping

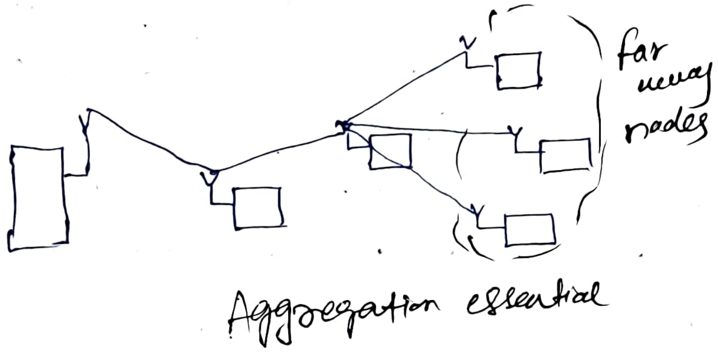
Push diffusion - few senders, many receivers. Here interests are not flooded rather relatively few data are flooded. Finally, the interested nodes will start reinforcing the gradients.

Pull diffusion : Many senders, few receivers, it still flood interest messages but directly set up a real tree.

Data aggregation:

To transmit data, packets need to combine their data into fewer packets, i.e. aggregation is needed. Depending on n/wok aggregation can be usefull or pointless.





Aggregation essential

Meet

Metrices for data aggregation

Accuracy: The difference betn values the sink obtains from aggregated packets and from the actual value (obtained in case no aggregation).

Completeness: Percentage of all readings included in computing the final aggregate at the sink.

Latency:

Message overhead:

How to express aggregation request

One option is database abstraction of WSN. Aggregation is requested by appropriate SQL clauses.

```
SELECT {agg(expr), attributes} FROM sensors
WHERE {selection predicates}
GROUP BY {attributes}
HAVING {havingPredicates}
EPOCH DURATION
```

agg(expr) - Actual aggregation functn. eg. AVG(temperature)

WHERE : filter on value before entering aggregation Process. Usually evaluated locally on an observing node.

GROUPBY : Partition into subsets, filtered by HAVING

- GROUP BY floor HAVING floor > 5

Aggregation operations:

→ **Duplicate sensitive**: ex: Median, sum, histograms : (These operations have duplicate values)

" insensitive : Maximum or Minimum.

→ **Summary or examplary**, → **composible**

→ for f aggregation function, there exist g such that Both f & g are aggregation functions.

$f(W) = g(f(W_1), f(W_2))$

→ **Behavior of Partial state records** (PSR)

Partial state records represent intermediate results (i.e compute the average, sum and number of Previously aggregated values

(i) Distributive - end results directly as PSR. Ex: MIN

(ii) Algeboric - PSR has constant size, end result easily derived.

(iii) Content-sensitive - size and structure depend on measured values ex: (histogram)

(iv) Holistic - All data need to be included, eg: Mean

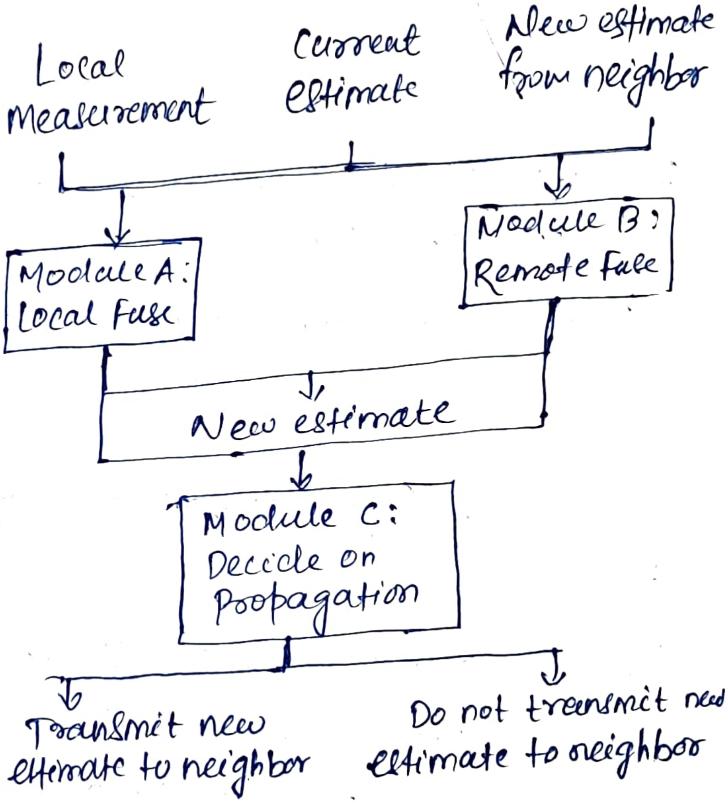
- Monotonic:

Broadcasting an aggregated value. (Gossiping + aggregation) (New estimate for aggregation is done when there is new informatn obtained from gussiping.

→ Here goal is to distribute aggregate of all node's measurements to all nodes. Therefore setting |V| (= no. of nodes)

Convergecast trees not useful.

→ The idea is to use (gossiping combined with aggregation.) (when new informatn. is obtained (due to gossiping) locally or from neighbor, compute the new estimate by aggregation.)


Local Measurement
Current estimate
New estimate from neighbor
Module A: Local Fuse
Module B: Remote Fuse
New estimate
Module C: Decide on Propagation
Transmit new estimate to neighbor
Do not transmit new estimate to neighbor

→ ~~A decision is made about new information to whether gossip the new estimate~~

→ Decision is made whether to gossip the new ~~information~~ estimate, detect whether a change is significant

Data storage: (Data centric storage)

Sometimes, data need to be stored for later retrieval which is a problem. The question is where on which node to put a data. The idea is ~~to~~ let the name of data describe which node is in charge.

→ Data name is hashed to a geographic position.
→ Node closest to this position is in charge of holding data.
→ Peer-to-peer networking / distributed hash tables
→ Geographic Hash Tables (GHT).
→ Use geographic routing to store/retrieve data at this location/node.