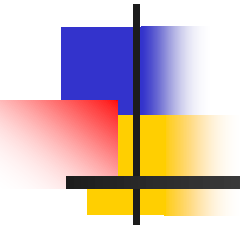
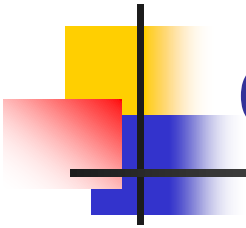


Opportunistic Networks

Barbara Guidi

8 Maggio 2012



Outline

- Opportunistic Networks
  - Ubiquitous Computing
  - MANETs
  - Delay Tolerant Networks (DTNs)
  - Caratteristiche principali
- Tecniche di Routing opportunistico
  - Senza infrastruttura
  - Con infrastruttura
- Data dissemination
  - PodNet
  - ContentPlace
- Haggle

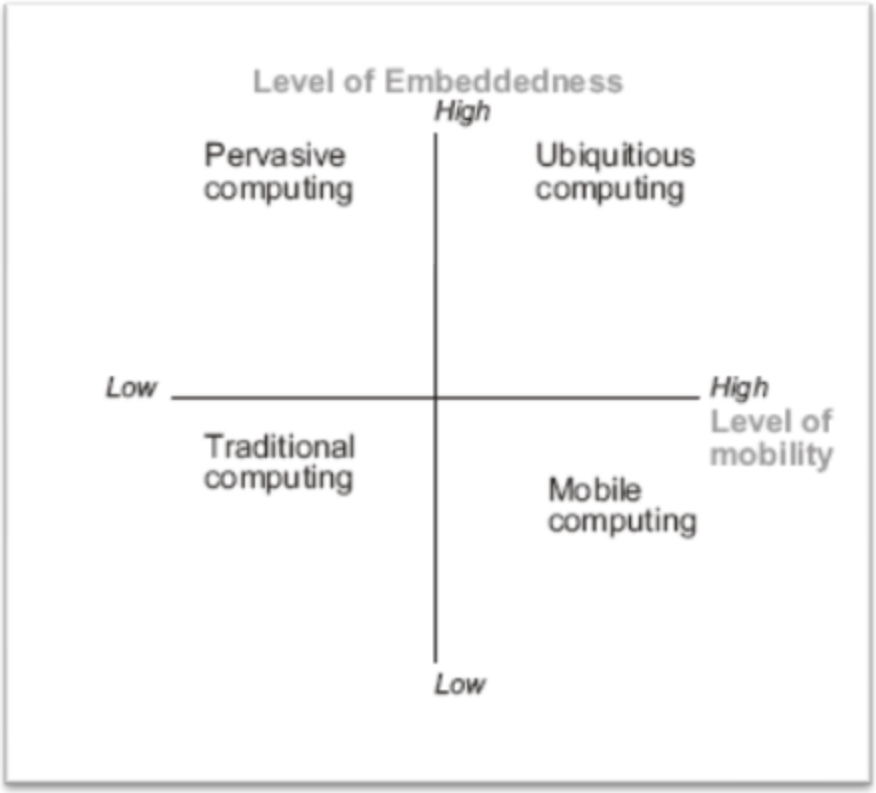
Opportunistic Networks, perché?

- Nel 1991, Mark Weiser, formulò la sua idea su di una nuova area della computer science, ubiquitous computing (UC)
- La sua visione prevedeva che le persone potessero muoversi e interagire con i computer

"continually interacting with hundreds of nearby wirelessly interconnected computers"

- Un concetto predominante nell'UC è la natura e la facilità di interazione degli utenti con l'ambiente intelligente
- Ubiquitous computing, pervasive computing e calm technology, l'ambiente deve essere progettato in modo che contenga già tutto quello di cui si ha bisogno

Opportunistic Networks, perché?



Con l'incremento delle tecnologie di comunicazione a basso raggio all'interno dei dispositivi mobili, nasce l'esigenza di nuove tipologie di comunicazione ed interazioni spontanee tra i vari dispositivi (Opportunistic Networks)

rappresentano il tentativo di colmare il divario tra l'uomo e il comportamento della rete adottando un approccio user-centric e sfruttando la mobilità come un'opportunità

Opportunistic Network (OppNets)

Reti wireless self-organized che utilizzano in maniera opportunistica tutte le possibilità offerte dai dispositivi distribuiti in modo pervasivo nell'ambiente o trasportati dagli utenti

- nate circa 7 anni fa come sviluppo dell'Ubiquitous computing e del pervasive computing
- Dimensione e struttura della rete non nota a priori
- Rete di nodi connessi attraverso wireless links
  - *Mobile nodes*, tecnologie di comunicazione wireless che coprono soltanto un'area di basso raggio (circa 100-300 metri)
  - *Fixed nodes*, device posizionati in locazioni specifiche della rete
- dispositivi con risorse limitate (batteria, memoria, …) con capacità di connessione wireless (PDAs o cellulari). In molti casi si parla di Bluetooth (Bluetooth SIG Inc., 2003-2005) o WiFi 802.11b (IEEE, 1999)

Opportunistic Networks (OppNets)

- Data sharing protocol:
  - *Node discovery*, un nodo può individuare qualsiasi nodo entro il proprio raggio di comunicazione
  - *One-hop message exchange*, un nodo invia e riceve messaggi arbitrariamente da nodi nel suo raggio di comunicazione
- Connettività ad intermittenza, trasferimento dati senza conoscenza del percorso da compiere (potrebbe non esistere un end-to-end path)
- si sfrutta la mobilità dei nodi all'interno della rete per poter realizzare la propagazione dei dati
- concetti molto vicini alle MANETs
- costituiscono un sottoinsieme delle Delay-Tolerant Network (DTNs)

MANETs: Richiami

Self-organized autonomous system, costituito da dispositivi mobili connessi ad hoc mediante collegamenti di tipo wireless

- infrastructured-less
- i nodi sono disponibili a propagare le informazioni
- La topologia della rete è dinamica e cambia frequentemente
- end-to-end path tra sorgente e destinazione
- Routing:
  - Ogni nodo ha una Routing Table (distanza in numero hops, indirizzo IP della destinazione, numero di sequenza della destinazione, …)
  - calcolo della distanza:
    - Distance vector, numero di hop che separa un nodo dalla sorgente
    - Link-State, topologia della rete. Una entry corrisponde ad un link noto all'interno della rete
  - Protocolli proactive e reactive

MANETs: Richiami

- Proactive Routing, ogni nodo periodicamente o dopo un cambiamento nella topologia trasmette la propria Routing Table ai vicini
  - Destination-Sequenced Distance Vector (DSDV)
    - Basato sul Distance Vector
    - Aggiornamento della tabella di routing "full dump" o con aggiornamento incrementale.
  - Optimized Link State Routing (OLSR)
    - ogni nodo collabora con i vicini per costruire una tabella di routing contenente i cammini verso tutti i nodi esistenti
    - i nodi si scambiano periodicamente dei messaggi per mantenere aggiornati i propri dati topologici (*Broadcast completo o Broadcast MPR)*
- riducono la latenza ma aumentano l'overhead dei dati

MANETs: Richiami

- Reactive Routing, i percorsi vengono costruiti on demand
  - Ad-Hoc On-Demand Distance Vector (AODV)
    - protocollo di tipo Distance vector
    - sviluppato sulla base del protocollo DSDV
    - una route viene costruita solo quando un nodo sorgente lo richiede e mediante un ciclo Route-Request/Route-Reply
    - Mantiene un destination sequence number per ogni entrata nella tabella di routing. Creato dal nodo destinazione e spedito al nodo sorgente tramite un route reply o un route request
    - Se un nodo deve scegliere tra due differenti cammini selezionerà quello con il numero di sequenza maggiore
- hanno basso overhead, ma alto ritardo nel calcolo delle rotte

Dalle MANETs alle OppNets

- MANETs:
  - Originariamente impiegate in ambienti militari
  - Applicazioni:
    - *precision agricolture*, si fanno uso di reti di sensori ad hoc per collezionare informazioni circa le condizioni del terreno e la variabilità spaziale e temporale nel raccolto e nella qualità
    - *Wildfire tracking*, utilizzato nella biologia
    - *health care*, le Personal Area Networks (PANs) sono utilizzate per applicazioni utili al miglioramente della salute delle persone
- MANETs vs. OppNets
  - Caratteristiche comuni:
    - infrastructured-less
    - self-organizing networks
    - device con risorse limitate

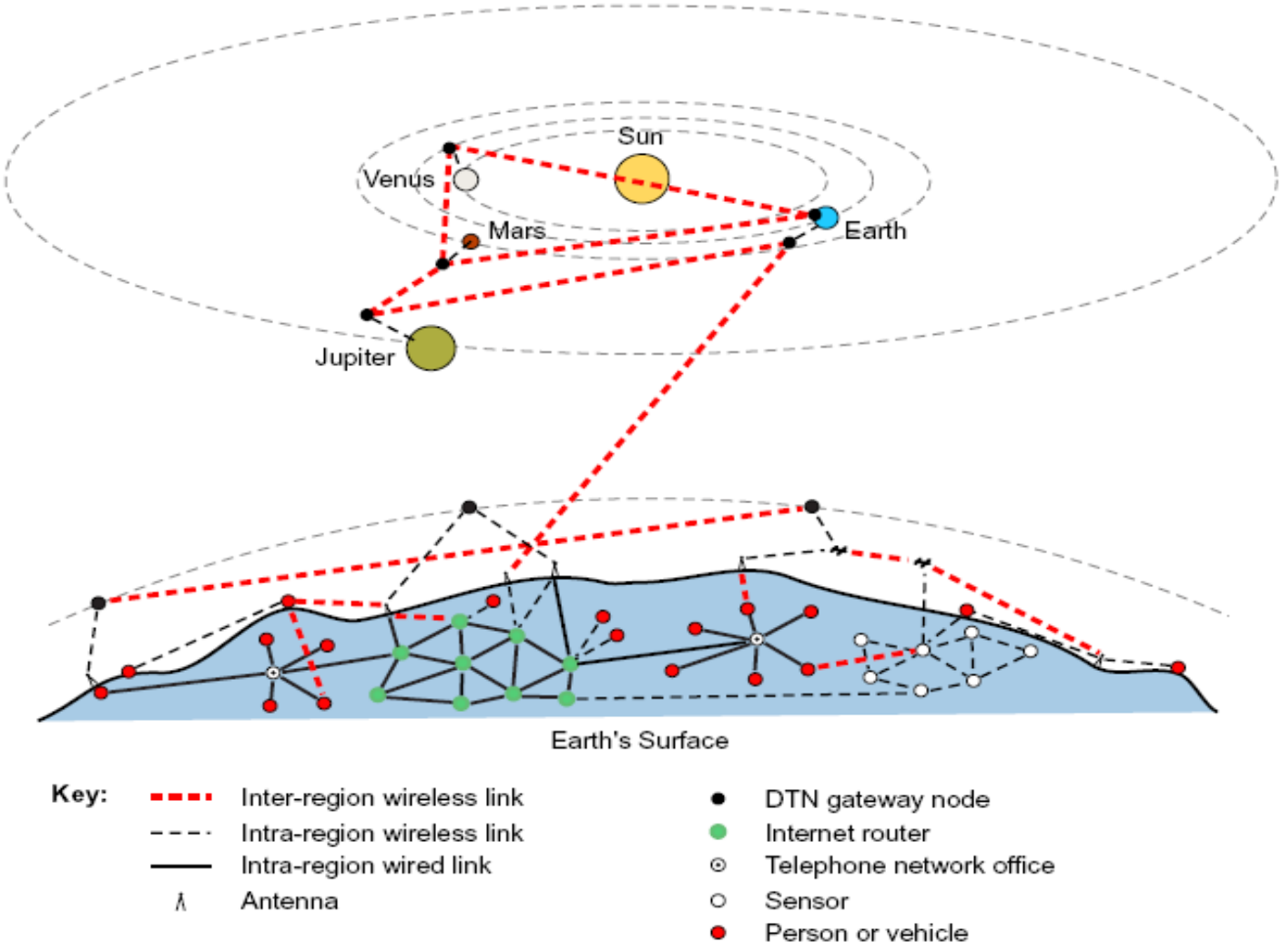
Dalle MANETs alle OppNets (cont.)

- MANETs vs. OppNets
  - Differenze:
    - Nelle MANETs si assume che tutti i nodi contribuiscano alla comunicazione e non sempre vero in uno scenario reale
    - nelle MANETs si presuppone che la rete sia connessa, nelle OppNets la rete può essere partizionata
    - Nelle OppNets non è certa l'esistenza di un end-to-end path. (***Store-Carry-and-Forward routing)***
    - le MANETs tollerano la mobilità dei nodi entro un certo limite, le OppNets sfruttano la mobilità
    - Network layer vs. Application layer. Nelle OppNets non ci sono vincoli sullo standard di comunicazione utilizzato

Delay Tolerant Network (DTN)

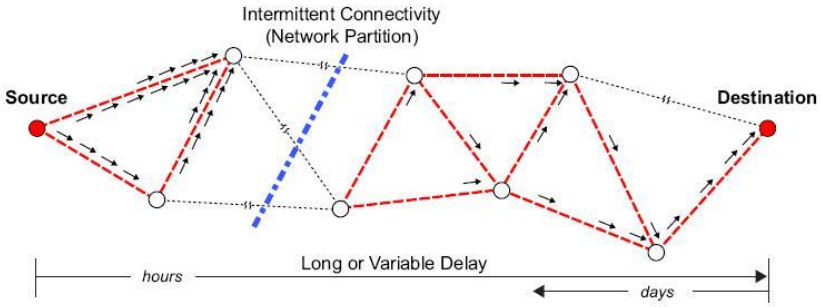
- Architettura che si propone come obiettivo quello di far comunicare tra loro reti indipendenti (regions networks), mutuamente incompatibili e non appartenenti ad Internet
- Un nodo appartente ad una region network può trasferire messaggi all'interno della propria rete, ma non può farlo all'esterno
- Una region network ha caratteristiche di comunicazione relativamente omogenee
- Un esempio di reti wireless al di fuori di internet sono ad esempio:
  - terrestrial civilian networks
  - military battlefield networks
  - interplanetary network

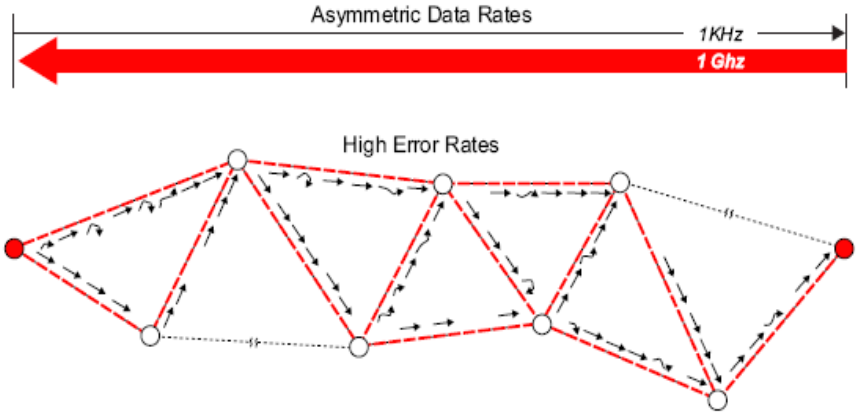
Delay Tolerant Network (DTN) (cont.)



Delay Tolerant Network (DTN) (cont.)

- L'architettura DTN definisce una serie di meccanismi che consentono di effettuare le seguenti operazioni:
  - minimizzare gli effetti finali del ritardo di trasmissione e di interruzione della comunicazione, sia che avvengano all'interno di una rete o nello scambio tra reti diverse
  - minimizzare gli effetti finali di errori, distorsioni e corruzione del segnale
  - aggirare le incompatibilità tra le diverse reti tramite un meccanismo opportuno di traduzione dell'informazione

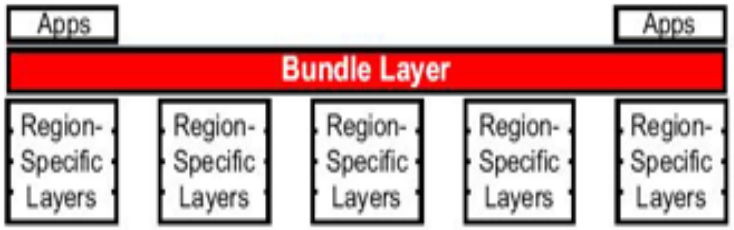




Delay Tolerant Network (DTN) (cont.)

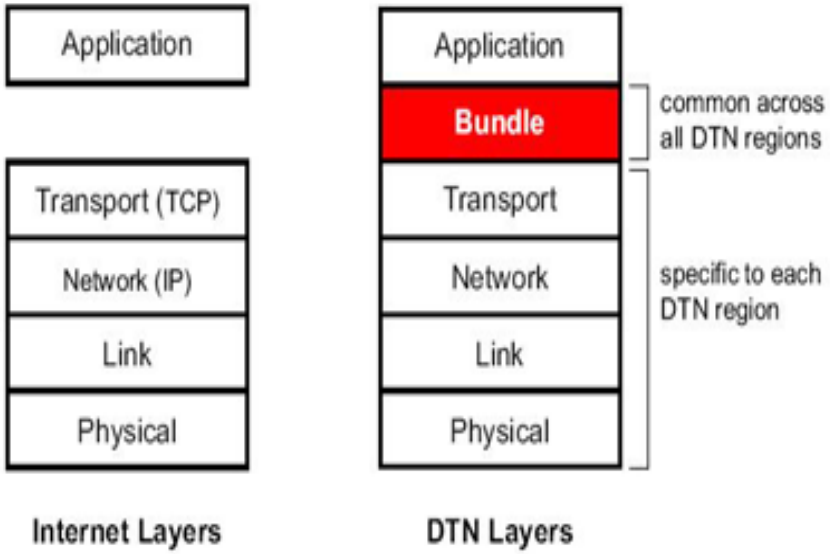
- Internet basata su un trasferimento di dati di tipo *packet switching*. La DTN trasferimento di tipo *store and forward message switching*
- Meccanismo di trasferimento basato su sessioni (non è necessaria conferma dal nodo ricevente)
- ogni nodo memorizza in modo persistente un messaggio (*store message*), o una sua frammentazione, per poi trasferirlo in blocco (*forward message*) al nodo successivo
- La memorizzazione persistente è fondamentale per
  - compensare eventuali lunghi periodi di perdita di comunicazione
  - compensare differenze anche notevoli di velocità di trasmissione/ricezione dei dati tra nodi
  - ritrasmettere il contenuto informativo in caso di necessità o di richiesta da parte del nodo successivo anche a distanza di tempo

Delay Tolerant Network (DTN) (cont.)



store-and-forward = i messaggi o frammenti di questi sono memorizzati all'interno dei vari nodi intermedi fino al raggiungimento della destinazione

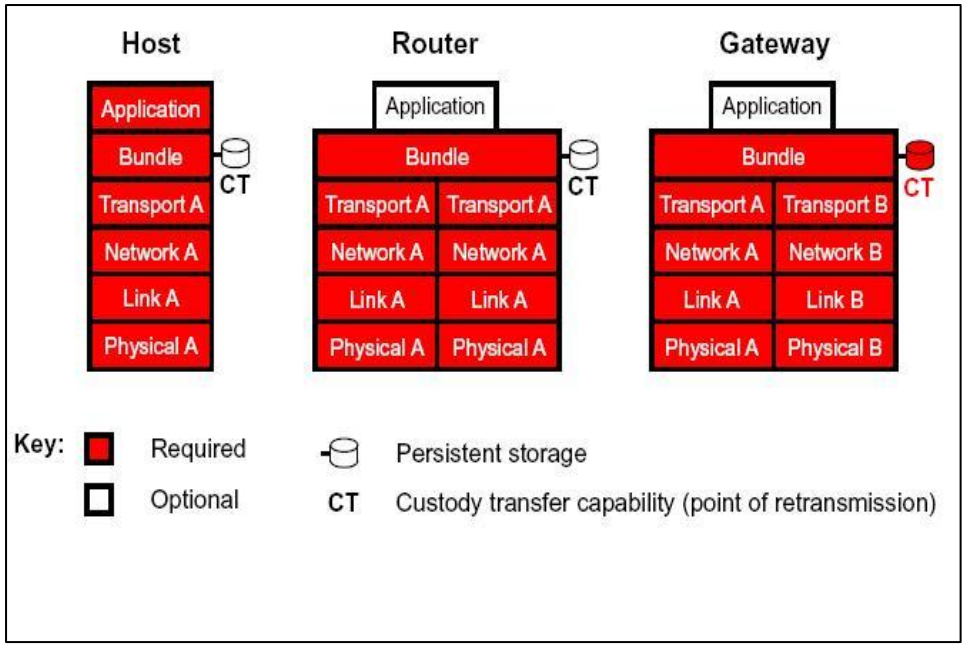
Per gestire ed implementare lo store-and-forward è introdotto un nuovo protocol layer: *bundle layer*



Il bundles layer effettua lo store dei messaggi e successivamente il forward quando possibile. I bundles sono i messaggi

DTN Nodes

in una DTN, un nodo è un'entità con un proprio bundle layer e può agire come *host*, *router* o *gateway* (o una combinazione di questi) e che agisce come una sorgente, una destinazione o un nodo intermedio



- *HOST* = invia e/o riceve bundles, ma non li inoltra come nodo intermedio. Un host può essere una sorgente o una destinazione di un trasferimento.
- *ROUTER* = inoltrano i bundles all'interno di una singola DTN region o possono essere opzionalmente un host.
- *GATEWAY* = inoltrano bundles tra due o più DTN regions e possono essere opzionalmente un host.

OppNets - Caratteristiche

- Le OppNets si basano sugli stessi principi delle Delay Tolerant Networks (DTNs) di cui sono una sottoclasse
- Le opportunità di comunicazione (contatti) sono intermittenti
  - i nodi sorgente e destinazione potrebbero essere disconnessi (mobilità dei nodi o sleep mode per risparmio energetico)
- OppNets tollerano un certo tipo di ritardo e un messaggio può essere trasmesso in seguito
- **Store-Carry-and-Forward routing pattern**
  - i messaggi possono essere immagazzinati fino a quando un percorso (parziale) verso la destinazione è disponibile grazie a: mobilità dei nodi, risveglio da sleep mode
  - lo memorizzazione del messaggio continua fino a quando non si verifica uno nuovo contatto opportunistico

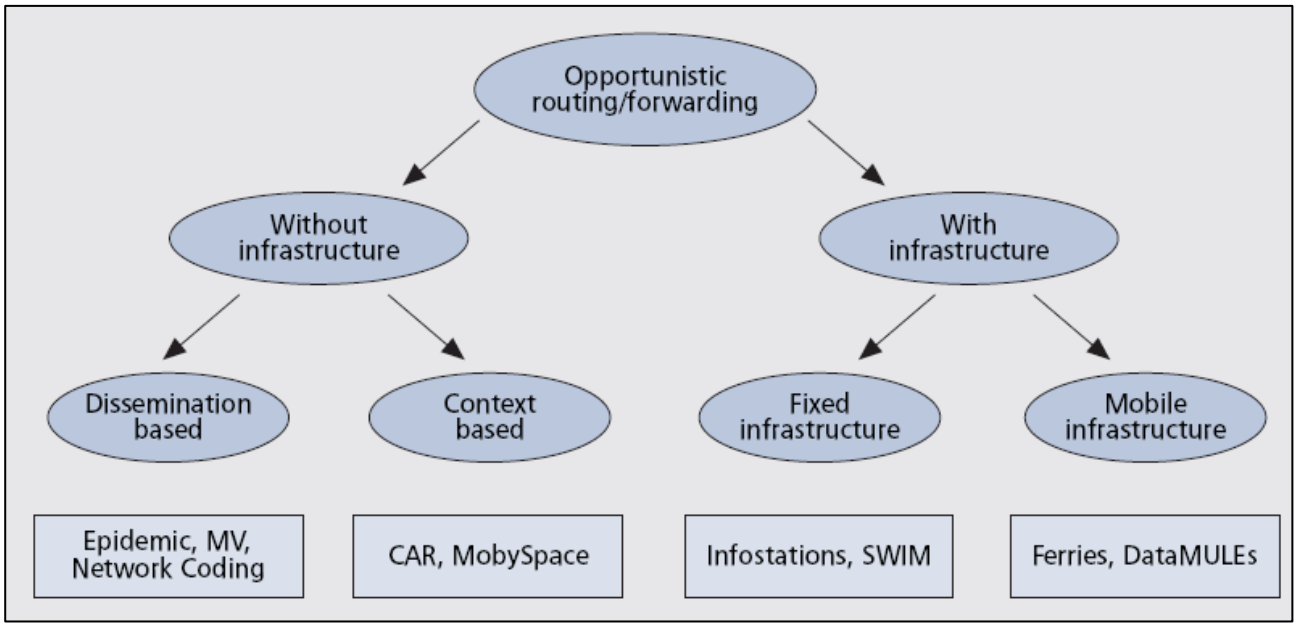
OppNets – Contatti Opportunistici

- le comunicazioni possono aver luogo soltanto durante i periodi di contatto
  - veloci ed efficaci
  - Un contatto è un'opportunità di comunicazione
- Caratteristiche di valutazione dell'opportunità di comunicazione:
  - ***Inter-contact time***, intervallo di tempo che intercorre tra due successivi contatti tra gli stessi devices.
  - ***Intra-contact time***, intervallo di tempo durante il quale due devices possono comunicare
  - ***Inter-any-contact time***, è il tempo che intercorre tra la fine di un contatto e l'inizio del contatto successivo con qualsiasi altro device

La conoscenza della distribuzione dell'inter-contact time fornisce una buona descrizione della connettività della rete

OppNets – Tecniche di routing

- Le tecniche di routing nelle OppNets, hanno lo scopo di massimizzare la probabilità di consegna del messaggio
- *delivery ratio*, rapporto tra la quantità totale di dati eventualmente ricevuta e il totale dei dati che erano stati inviati nella rete



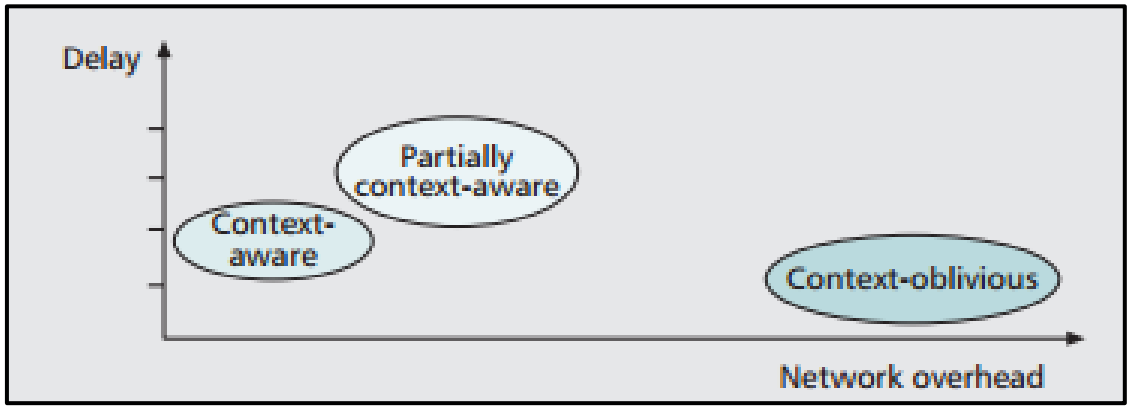
OppNets – Tecniche di routing (cont.)

- I protocolli che fanno parte delle OppNets *senza infrastruttura* si dividono in:
  - *Dissemination-based routing* si basa su forme di flooding controllato. Conosciuto con il nome di *Context oblivious*.
  - *Context-based routing* solitamente non adotta il flooding dei messaggi. Utilizza la conoscenza sul contesto in cui i nodi stanno operando per identificare il migliore next-hop.
- I protocolli che fanno parte delle OppNets *con infrastruttura* si dividono in:
  - fixed infrastructure
  - mobile infrastructure
  - In entrambi i casi si hanno nodi speciali che presentano maggiore potenza
    - capacità di storage elevata
    - possono collezionare messaggi da molti nodi anche per lungo tempo
    - livello di batteria migliore rispetto agli altri

Routing senza infrastruttura

una tassonomia alternativa (2008) effettua una separazione tra i protocolli *context-based routing*, ottenendo i seguenti 3 approcci:

- *Context-oblivious*, o dissemination-based routing. Non utilizzano alcuna informazione circa il comportamento degli utenti
- *Partially Context-aware*, conosciuto anche come *mobility-based*. Utilizzano le informazioni di contesto ed assumono uno specifico modello per il contesto
- *Fully Context-aware*, conosciuto come *social context-aware.* Si apprende dal contesto e lo si utilizza per la delivery di un messaggio

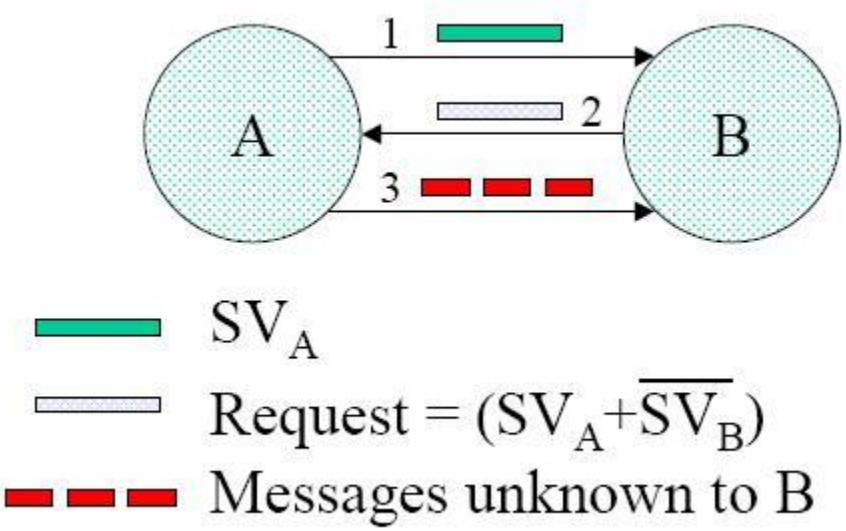


Context Oblivious Routing

- i nodi effettuano il flooding dei messaggi su tutti i link in uscita
- Un messaggio dovrebbe essere disseminato il più ampiamente possibile nella rete
- Unica soluzione quando non sono disponibili informazioni sul contesto
- Possono causare la congestione della rete e consumo elevato di batteria e memoria. Tecniche per limitare i consumi:
  - Limitazione del numero delle copie
  - Limitazione nel numero degli hops
- Esempi:
  - Epidemic routing
  - Spray & Wait
  - Network Coding

Context Oblivious Routing: Epidemic

- *Epidemic Routing*, simile alla diffusione di un virus in una comunità
- Obiettivi
  - Massimizzare il delivery rate
  - Minimizzare la latenza
  - Minimizzare le risorse consumate durante la consegna di un messaggio



• Quando due nodi vengono in contatto si scambiano un vettore (*summary vector*) che contiene informazioni sui messaggi correntemente memorizzati localmente nei nodi

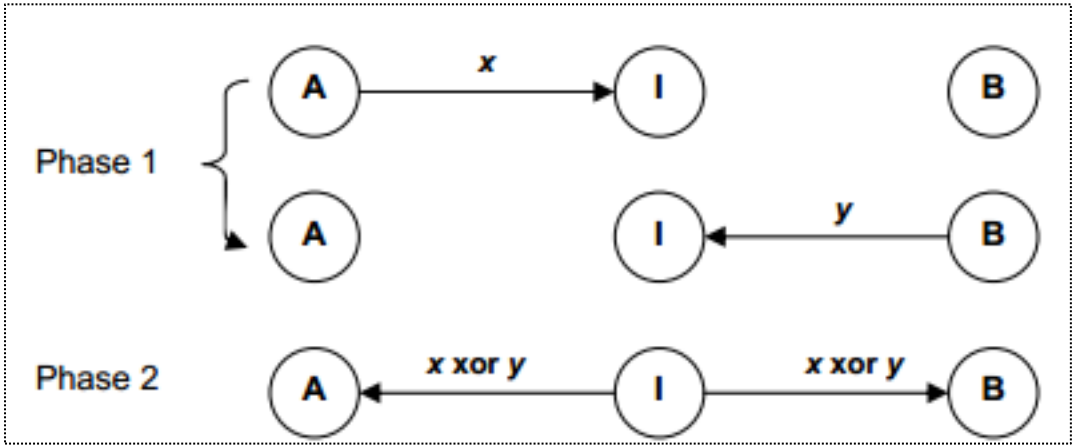
• Successivamente ogni nodo chiede all'altro i messaggi che non ha localmente (Request Vector)

Context Oblivious Routing: Spray&Wait

- Si distinguono due fasi: Spray phase e Wait phase
  - *Spray phase*: un numero (L) di copie dello stesso messaggio vengono diffuse all'interno della rete dalla sorgente
  - *Wait phase*: ogni nodo mantiene una copia del messaggio memorizzandola in attesa di poter trasmettere il messaggio alla destinazione
- L può essere scelto in base al ritardo medio calcolato per raggiungere la destinazione. Indipendente dalla dimensione della rete e dal raggio di trasmissione
- Problema aperto: come diffondere le L copie del messaggio
  - Source Spray&Wait, L corrisponde al numero di nodi direttamente connessi alla sorgente
  - Binary Spray&Wait, la sorgente invia L copie; ogni nodo A ha n > 1 copie (ottenute dalla sorgente o tramesse nei contatti); A incontra un nodo B. Passa a B n/2 copie del messaggio e mantiene n/2

Context Oblivious Routing: Network Coding

- Un messaggio (o un gruppo di messaggi) viene trasformato in un altro formato prima della trasmissione
- Si inseriscono delle informazioni addizionali all'interno dei blocchi codificati in modo da poter ricostruire il messaggio originale con solo un numero limitato di blocchi codificati
- Un blocco si considera consegnato correttamente quando il numero di blocchi necessari per ricostruire il messaggio originale è stato consegnato alla destinazione
- Robusto in caso di connettività scarsa. Limita il flooding dei messaggi



Mobility-based Routing

- Sfruttano alcune informazioni riguardo il contesto in cui i nodi operano (es. la mobilità dei nodi)
- Customized per uno specifico tipo di context-information
- Non viene fornito un insieme di algoritmi completamente sviluppati per ogni tipo di context-information
- Riduzione del numero dei messaggi rispetto al Context-oblivious routing, ma richiedono un elevato costo computazionale
- Incremento del ritardo nella consegna dei messaggi per possibili errori nel percorso e inaccuratezza nella selezione degli hops
- Esempi:
  - *PROPHET*
  - *MV e MaxPro*
  - *Last Encounter Routing e Spray&Focus*
  - *MobySpace Routing*
  - *Bubble Rap*

Mobility-based Routing (cont.)

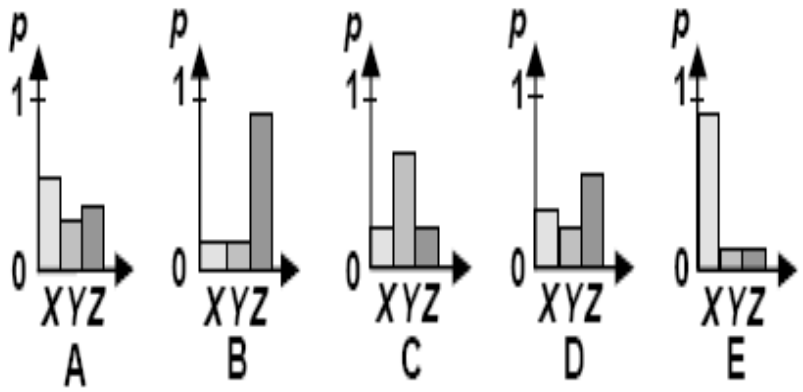
- Probabilistic Routing Protocol using History of Encounters and Transitivity (PROPHET)
  - Evoluzione dell'Epidemic Routing
  - *delivery predictability*, probabilità di incontrare una determinata destinazione
  - Routing probabilistico basato sulla storia dei contatti ("History of past Encounters and Transitivity")
  - Un messaggio viene trasmesso ad un nodo vicino se e solo se questo nodo ha una delivery predictability più alta
  - Legge transitiva: A incontra frequentemente B e B incontra frequentemente C, allora i nodi A e C hanno un'alta delivery predictability l'uno verso l'altro
  - durante un contatto i nodi si scambiano il proprio delivery predictability verso la destinazione per i messaggi che memorizzano. I messaggi sono inoltrati solo se il delivery predictability del nodo richiedente è più alto di quello in cui è attualmente memorizzato il messaggio

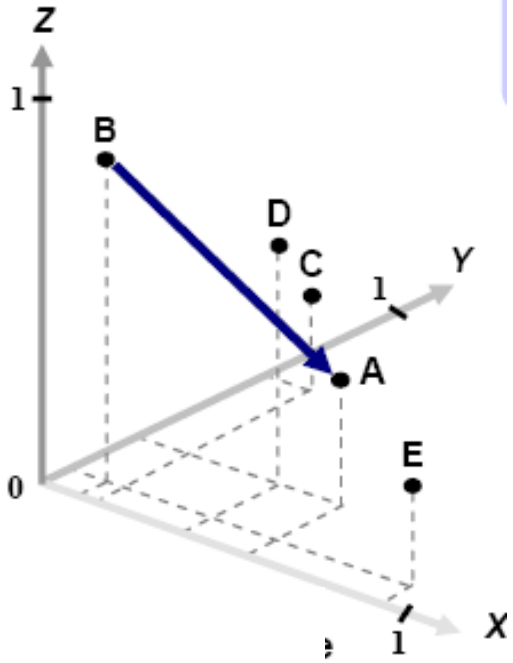
Mobility-based Routing (cont.)

- MV e MaxProp, routing probabilistico come in PROPHET. Utilizzano informazioni riguardo la frequenza di visite in locazioni fisiche specifiche
- Spray&Focus, incrementa la Wait Phase del protocollo Spray & Wait, con la Focus phase. Ogni nodo può inoltrare la propria copia del messaggio ad un nodo potenzialmente più appropriato utilizzando un utility-based scheme.
- MobySpace Routing
  - Pattern di mobilità utilizzato come context information
  - Costruisce uno spazio Euclideo (MobySpace), dove ogni asse rappresenta una locazione nello spazio fisico
  - Le coordinate di un nodo corrispondono alla probabilità di trovarsi in quella posizione

MobySpace Routing

B vuole inviare un messaggio al nodo E





B invierà il messaggio al nodo A che è più vicino alla destinazione

Mobility-based Routing (cont.)

- Bubble Rap
  - conoscenza delle comunità sociali degli utenti
  - le comunità sono automaticamente individuate attraverso dei patterns basati sul contatto tra nodi. Le comunità sono etichettate
  - ogni messaggio originato in una comunità diversa da quella in cui si trova la destinazione è trasmesso come segue:
    - il nodo A ha un messaggio indirizzato a D ed incontra il nodo B
    - Il messaggio è passato a B se la comunità a cui B appartiene è la stessa comunità di D, o se B ha un ranking più elevato rispetto ad A
    - Il ranking è misurato sulla base dell'insieme dei peers con cui un nodo è solitamente in contatto ed è dunque una misura della socialità di un nodo
  - Bubble Rap cerca i nodi appartenenti alla stessa comunità del nodo destinazione. Se non ne trova, si invia il messaggio in modo da incrementare sempre più la socialità dei nodi
  - l'utilizzo di informazioni relative alle relazioni sociali tra nodi è uno degli aspetti più promettenti della ricerca attuale

Modelli di mobilità

- Generano delle sequenze di contatti tra device basati su di un insieme di parametri che caratterizzano l'ambiente
- Simulano i movimenti dei nodi
  - *Random Walk Mobility Model*, direzione e velocità random
  - *Random Waypoint Mobility Model*, include delle pause tra i cambiamenti di destinazione e di velocità
  - *Random Direction Mobility Model*, un modello che obbliga i device a viaggiare sul lato dell'area di simulazione prima di cambiare direzione e velocità
  - *Probabilistic Version of Random Walk Mobility Model*, utilizza un insieme di distribuzioni di probabilità per determinare la successiva posizione dei devices
  - *City Section Mobility Model*, area di simulazione che rappresenta le strade all'interno di una città

Social Context-aware Routing

I social context-aware protocols non considerano soltanto informazioni di mobilità, ma prendono in considerazione anche gli aspetti di socialità tra i nodi

Approccio molto più generale di quelli analizzati in precedenza, in quanto possono sfruttare qualsiasi insieme di informazioni di contesto, permettendo al sistema di poter essere customizzato

Esempi:

- Context-Aware Routing (CAR)
- History-based Opportunistic Routing Protocol (HiBOp)
- Probabilistic Routing Protocol for Intermittently Connected Mobile Ad Hoc Network (PROPICMAN)

Context-Aware Adaptive Routing (CAR)

- I nodi sono divisi in partizioni
  - All'interno della stessa partizione i nodi sono connessi da un protocollo di routing proactive (es. DSDV), ed ogni nodo ha la propria Routing Table
  - I nodi posizionati in altre partizioni della rete non sono raggiungibili attraverso il protocollo di routing proactive
- La scelta del next-hop è effettuata sfruttando l'informazione di contesto, quale:
  - Connettività logica della rete:
    - La percentuale di cambiamento nella connettività
    - Il grado di mobilità
  - Informazioni sui dispositivi mobili
    - Livello di batteria residua
    - Memoria disponibile
- I nodi agiscono come intermediari per la comunicazione asincrona verso la destinazione o verso i gateways (fissi o mobili)

Context-Aware Routing (CAR)

- Si espande il classico protocollo di routing con l'aggiunta della "Delivery Probability" per ogni item
- Funzione di utilità calcolata usando Multi-attribute-utility-based
  - Ogni attributo di contesto ha associato un proprio peso (utilità)
- Ogni nodo mantiene una lista contenente: *destinazione, bestHost* e *delivery probability*
- la Delivery Probability è calcolata proattivamente e disseminata (DSDV) nelle varie partizioni periodicamente per garantire che l'invio dei messaggi sia sempre supportato da informazioni aggiornate
- CAR è efficiente quando le informazioni di contesto sono predicibili

HiBOp

- Simula l'instaurazione di relazioni sociali tra le persone
- Contiene dei meccanismi per poter gestire qualsiasi context information
  - Context information: collezione di informazioni che descrivono la comunità
  - Gli utenti descrivono se stessi
    - Gli utenti decidono quali informazioni circa se stessi e le loro abitudini rendere pubbliche
    - Gli utenti memorizzano tali informazioni all'interno dei propri device
- Gli utenti scambiano le informazioni personali durante i contatti. In questo modo apprendono il contesto in cui sono immersi

HiBOp (cont.)

- Ogni nodo mantiene una Identity Table (IT) in cui memorizza le proprie informazioni
- *History Table:* memorizza i valori che un nodo ha conosciuto nel passato attraverso le ITs dei vicini
- Informazioni in IT + contenuto delle tabelle dei vicini = *Current Context*
- Valutazione della similarità dei nodi per giungere a destinazione
  - Un nodo inoltra un messaggio ad un altro nodo se questo è più "simile" alla destinazione
  - Probabilità calcolata rispetto alla storia del nodo e del Current Context

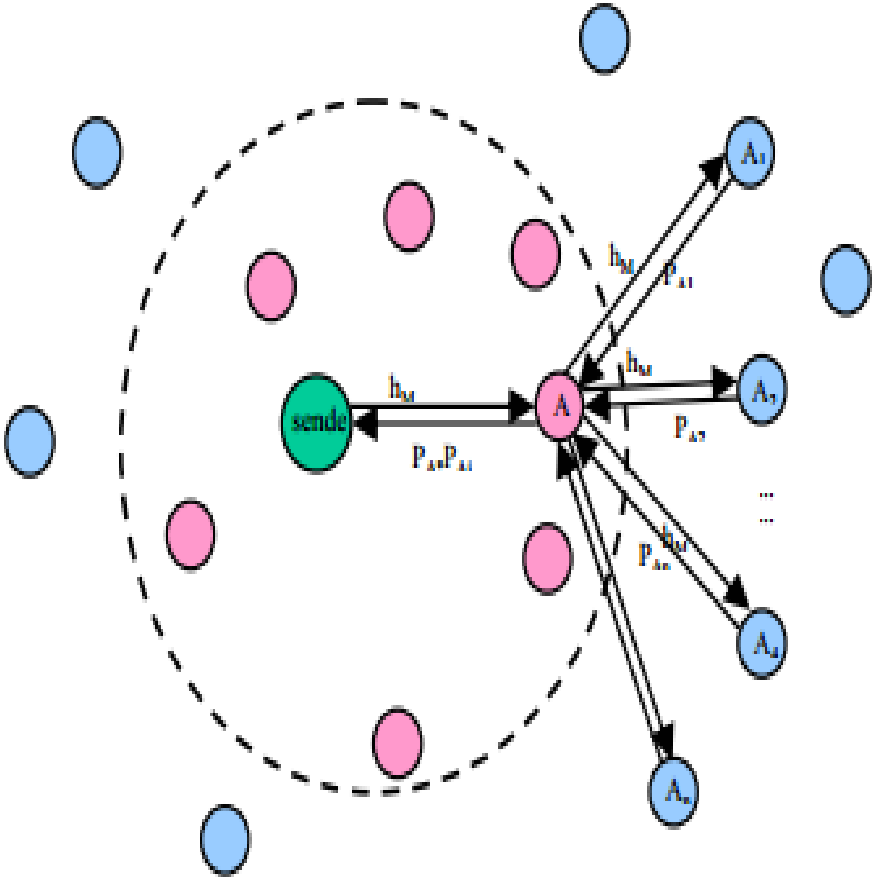
$$p = \alpha * p_H + (1-\alpha) * p_{CC}$$

Propicman

- Probabilistic Routing Protocol for Intermittently Connected Mobile Ad Hoc Network (PROPICMAN)
  - Ogni nodo ha un profilo utente in cui ad ogni attributo è associato un peso diverso
    - Il comportamento degli utenti non è casuale ma predicibile in base ai comportamenti ripetitivi degli utenti
    - il profilo di un utente è un elemento flessibile
  - Sicurezza all'interno della comunità
    - meccanismi di sicurezza sul contenuto del messaggio e sulle informazioni degli utenti
  - Obiettivi:
    - Efficienza nella distribuzione dei messaggi in modo probabilistico
    - Minimizzare il numero delle risorse consumate per ogni singolo messaggio
    - Massimizzare la percentuale dei messaggi consegnati alla destinazione

Propicman (cont.)

- Il nodo Sender vuole inviare un messaggio M verso un nodo D (destinazione)
- Sender invia ai propri vicini lo header di M contenente le informazioni su D (di cui è in possesso)
- I vicini di Sender calcolano la propria delivery probability su tali informazioni
- L'header è inoltrato ai vicini dei vicini di S (two-hop neighbors)
- Sender invierà il messaggio M attraverso two-hop routes con elevata delivery probability, solo se questa è più alta della sua delivery probability
- Sender dopo l'invio mantiene il contenuto del messaggio per sfruttare ulteriori vicini ottenuti nei contatti



Propicman (cont.)

Table 1: Common set of evidences T

Evidence Names(E)	Weights(W)
E_1	W_1
…	…
E_n	W_n

Table 2: A node profile (V_i can be empty)

Evidence name(E)	Value(V)	Hashed values
E_1	V_1	$H(E_1,V_1)$
…	…	…..
E_n	V_n	$H(E_n,V_n)$

Evidence Name (E)	Value (V)
Work Place	IAM
City	Bern

node profile, ogni nodo può creare il proprio profilo a partere da un insieme comune T di attributi (evidences). Ad ogni evidence è associato il relativo peso

Il profilo di un nodo è rappresentato dall'insieme delle coppie evidence/value. Ad ogni coppia è applicata una funzione hash per ragioni di sicurezza H(Ei,Vi)

Propicman (cont.)

- La sorgente costruisce l'header del messaggio M concatenando tutte le coppie evidences/values a cui è stata applicata la funzione hash come segue:

$$h_M = \underset{i=1}{\overset{n}{Concatenation}}(H(E_{D_i}, V_{D_i}), MAC_S, SN_M)$$

- Invia l'header ad ogni neighbour senza includere il contenuto del messaggio
- Ogni neighbour node A, quando riceve l'header confronta le coppie hashed di evidences/values con le proprie
- Per ogni elemento contenuto anche in A viene calcolata la delivery probability verso la destinazione come segue:

$$0 \leq P_A = \frac{\sum Matched(W_i)}{\sum W_{Mi}} \leq 1$$

somma dei pesi delle evidences in A di cui si ha un match in M

somma dei pesi di tutte le evidences in M

Routing con infrastruttura

- Si dividono in base alla struttura utilizzata
  - *fixed infrastructure*
  - *mobile infrastructure*
- nodi speciali con maggiore potenza, alta capacità di storage e possono collezionare messaggi da molti nodi anche per lungo tempo
- Fixed infrastructure, i nodi sono sparsi all'interno della rete in punti geografici specifici (es. base stations)
  - Le base stations sono generalmente dei gateways che ad esempio forniscono accesso ad internet o ad una LAN
  - *node-to-base-station communications (Infostation model)*
  - *Node-to-base-station* e *node-to-node* (SWIM- Shared Wireless Infostation Model)
  - Le base stations fisse giocano un ruolo passivo nel routing perchè semplicemente funzionano da sinks

Routing con infrastruttura (cont.)

- Mobile infrastructure, costituita da mobile nodes che si muovono all'interno della rete seguendo dei percorsi prefissati oppure completamente casuali
  - Base stations tipicamente giocano un ruolo attivo. Sono: *carriers, supports, forwarders, MULEs, o ferries*
  - node-to-carrier communications , le base stations sono le uniche entità responsabili del messages delivery
  - node-to-node and node-to-carrier communication, aiutano l'incremento della connettività nelle reti sparse e garantiscono il raggiungimento di nodi isolati
  - Permettono un risparmio di energia ai mobile nodes
  - Il routing nelle fixed infrastructure è molto costoso a causa dei costi dell'infrastruttura (aggiungere nuovi nodi implica l'espansione dell'infrastruttura)

Routing con infrastruttura(cont.)

- *Data-MULE system* , data retrieval in sparse wireless sensor networks
  - A basso livello ci sono i sensor nodes che periodicamente effettuano il data sampling dall'ambiente circostante
  - Il livello intermedio è costituito da mobile agents (MULEs), i quali si muovono nell'aria coperta dai sensori per raccogliere i loro dati
  - A livello più alto si hanno un insieme di wired Access Points (APs) e data repositories, i quali ricevono informazioni dai MULEs. Sono connessi ad un data warehouse centralizzato
- *Message Ferrying*, extra mobile nodes opportunisticamente utilizzati per offrire un servizio di trasmissione dei messaggi
  - Questi nodi (*Message Ferries)* si muovono nella rete e collezionano messaggi dai nodi sorgente
  - *La collezione dei dati può avvernire in due modi:*
    - ***Node Initiated Message Ferrying*** , ogni nodo conosce il percorso effettuato dagli active ferriers e si muove per incontrarli quando ha dati da consegnare
    - ***Ferry-Initiated Message Ferrying,*** il ferry node si muove attraverso un default path. Ogni source node invia una ServiceRequest ad un ferry in cui include la propria posizione. Ricevuta la richiesta il ferry cambia la propria traiettoria per incontrare il nodo

Data Dissemination

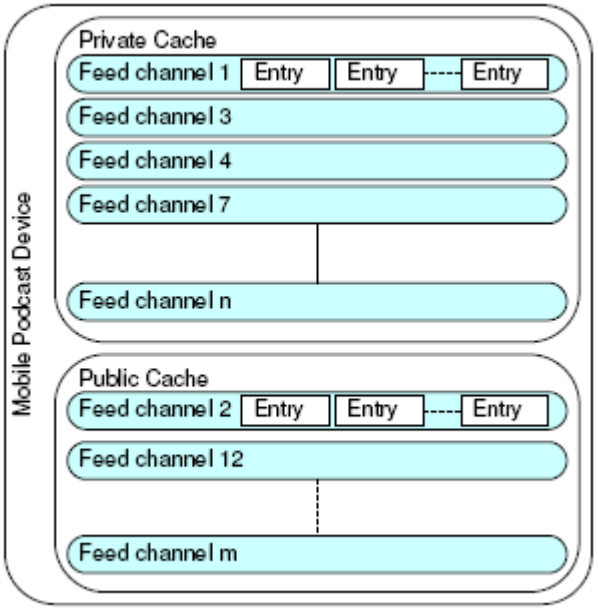
- Data forwarding, sorgente e destinazione sono noti
- Data dissemination
  - seguito dei protocolli di routing
  - supporta applicazione nelle quali l'insieme degli utenti interessati alla ricezione di un dato non sono noti a priori
  - data-centric networking, le topologic-centric networks non sono applicabili
  - Approccio gossip
  - sono utilizzati nelle applicazioni contenenti UGC (User Generated Content)
  - variazione del classico paradigma publish/subscribe
    - Web 2.0 User Generated Content (UGC) paradigm

PodNet – Podcasting for Ad Hoc Network

- Podcasting è divenuto famoso in Internet per la distribuzione di files multimediali (canzoni, video, …)
- uno o più Access Point (AP) disponibili per recuperare informazioni da internet e inviarle a tutti i nodi che si trovano all'interno del raggio utilizzato per la comunicazione
- i contenuti devono essere generati dagli utenti stessi in accordo al paradigma UGC
- i *content items* sono organizzati in canali e si basano sull'informazione che essi contengono
  - I contents sono unità atomiche
- Ogni item è un entry di un canale
- Ogni nodo ha un proprio *discovery channel* che elenca tutti i canali di cui un nodo fornisce delle entries;
  - La lista potrebbe essere vuota

PodNet – Podcasting for Ad Hoc Network

- I canali possono essere di due tipi:
  - *subscribed channels, quelli che l'utente riceve*
  - *public channels , utilizzati per l'invio ad altri nodi*



Dimensione dipendente dalla capacità di storage del dispositivo e da tutte le entries dei canali sottoscritti

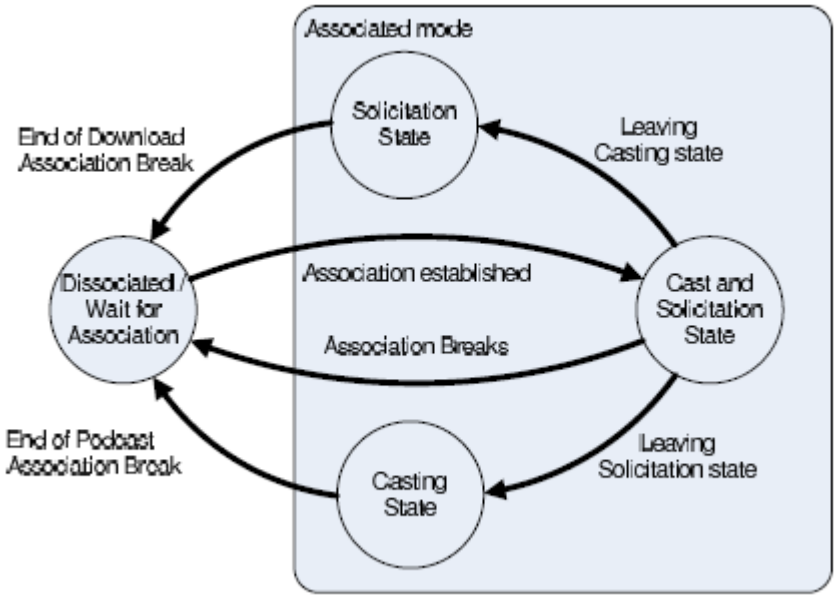
• A e B entrano in contatto

• il nodo A richiede a B il discovery channel e viene cosi a conoscenza dei canali posseduti dal nodo B

• A può recuperare le entries di suo interesse

Questo semplice meccanismo permette ad un nodo di poter entrare a far parte del Wireless Ad Hoc Podcasting appena incontra un nodo che ha dei canali di suo interesse. Potrebbe richiedere tempo.

PodNet – Podcasting for Ad Hoc Network



Casting State (server) = stato in cui possibile accettare delle solicitazioni
Solicitation state (client) = stato in cui è possibile generare messaggi di richiesta.

1. Il nodo soddisfa la richiesta dell'utente inviando il contenuto dei subscribed channels
2. Quando tutte le entries dei subscribed channels sono downloaded, può aggiornare e scaricare i contenuti dei canali pubblici

SOLICITATION STRATEGIES:

Un nodo deve sempre scaricare entries di suo interesse

• *Most Solicited*, ha lo scopo di incrementare la probabilità che un nodo incontrato possa essere interessato a canali presenti nella public cache. Un nodo prenderà sempre I canali più importanti

• *Least Solicited*, favorisce i canali meno importanti

• *Uniform*, la scelta dei canali avviene in maniera random

• *No Caching*, non si ha una cache pubblica e si ritornano solo canali della cache privata

PodNet – Podcasting for Ad Hoc Network

- Content-centric
  - Popolarità di un canale, vi sono canali con più subscribers e altri con meno
  - Le statistiche sulla popolarità dei canali sono locali
- Ogni politica in PodNet è un'euristica calcolata in funzione della popolarità del canale
  - Con nodi in movimento ed eterogenei, che tendono a clusterizzarsi potrebbe essere limitativo

ContentPlace

- Data dissemination system
  - utenti logicamente raggruppati in base agli interessi
  - i movimenti degli utenti sono guidati dalle relazioni sociali
  - UGC paradigm
  - Sfrutta la conoscenza delle sottoscrizioni degli utenti e la disponibilità dei data objects per decidere quali dati replicare localmente
- Funzione di utilità, ogni oggetto ha un valore
  - Quando un nodo incontra un peer, calcola il valore di utilità per tutti i dati nella propria cache e in quella del peer
  - Seleziona l'insieme dei dati che massimizzano l'utilità locale della cache
  - L'insieme degli oggetti da memorizzare dopo ogni contatto si ottiene attraverso il multi-constrained 0-1 knapsack problem

ContentPlace (cont.)

- multi-constrained 0-1 knapsack problem

$$\begin{cases} \max & \sum_k U_k x_k \\ \text{s.t.} & \sum_k c_{jk} x_k \leq 1 \quad j = 1, \ldots, m \\ & x_k \in \{0,1\} \quad \forall k \end{cases}$$

  - Fino a quando m non è grande, la risoluzione del problema è veloce

- Funzione di utilità
  - Gli utenti sono raggruppati in comunità
  - *Home community*, un utente può appartenere a più comunità (working community, family community)
  - *Acquainted community*, l'utente può avere relazioni con delle non Home community
  - Gli utenti si muovono in base alle proprie relazioni sociali

ContentPlace (cont.)

- L'utilità di un data object è ottenuta in base alle relazioni sociali rispetto alle varie comunità
  - Most Frequently Visited (MFV), ogni comunità ha un peso proporzionale al tempo medio speso dall'utente all'interno della comunità
  - Most Likely Next (MLN), il peso della comunità *ith* è uguale alla probabilità di visitare la comunità ith considerando il fatto che l'utente si trova nella comunità corrente
  - Future (F), il peso della comunità corrente è 0, le altre comunità sono settate come in MFV
  - Present (P), il peso della comunità corrente è 1, le altre sono 0
  - Uniform Social (US), a tutte le comunità con cui l'utente è entrato in contatto viene dato lo stesso peso

ContentPlace (cont.)

- Architettura:
  - Non vengono fatte assunzioni su di una specifica tecnologia di comunicazione

  - All'inizio di un contatto è stabilita una nuova associazione tra i nodi
  - I nodi si scambiano lo stato dei rispettivi buffers
  - Si effettua la richiesta dei dati e si memorizzano i dati ricevuti
  - Si chiude l'associazione tra i nodi

Haggle

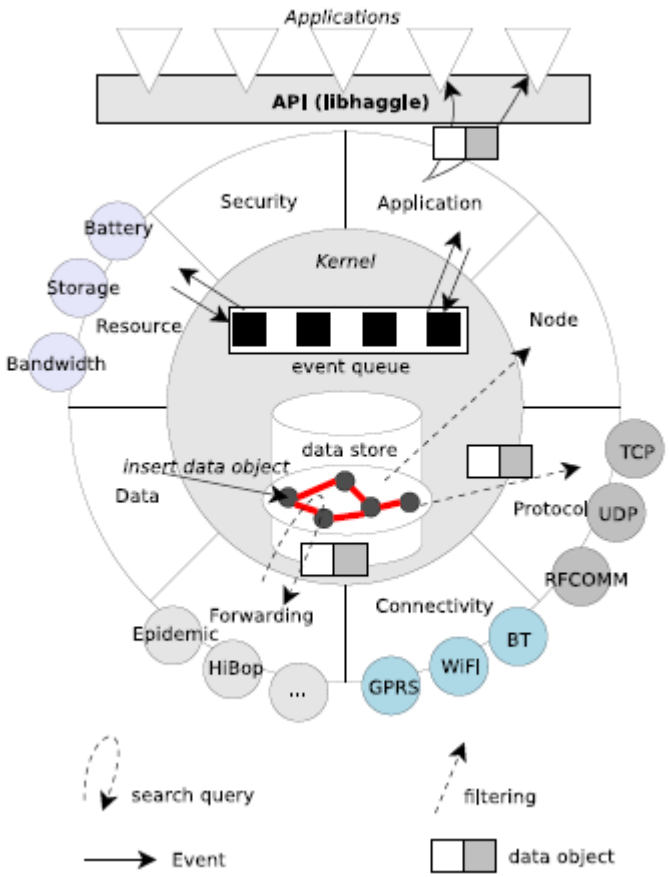
Progetto Europeo (2006-2010)

http://www.haggleproject.org/

Haggle (cont.)

- consente la comunicazione in presenza di reti wireless self-organized, che utilizzano in modo "opportunistico" tutte le possibilità di comunicazione offerte per realizzare una rete che sfrutti e si adatti al comportamento e ai bisogni delle persone
- Haggle deriva da un gioco di parole (Google e Ad-hoc). Inoltre significa “negoziazione”
- un’architettura layer-less, event-drive
  - Kernel, implementa una coda ad eventi. Contiene inoltre delle strutture dati condivise (active neighbors, listening sockets)
  - I managers sono responsabili di task specifici e interagiscono solo attraverso la produzione e la consumazione di eventi

Haggle (cont.)



- Il Data Manager è responsabile dei Data Objects (DO) che sono il formato dei dati utente
- The Name Manager è responsabile di classi speciali appartenti ai DOs che hanno un attributo ”Name” (NO) (es. Email address)
- Il Connectivity Manager contiene oggetti che rappresentano l’interfaccia della rete fisica. Funzionalità di neighbours discovery, apertura/utilizzo/chiusura canali e stima dei costi di rete (tempo ed energia)
- Il Protocol Manager incapsula I protocolli per lo start-up e ne provvede l’accesso
- Il Forwarding Manager fornisce metodi per inviare dati e contiene gli algoritmi di forwarding
- Il Resource Manager controlla le risorse utilizzate valutandone I costi

Haggle (cont.)

- Data Object (DO)
  - Un header (XML) che comprende il numero di attributi a cui il metadato si riferisce (tipicamente: data type, data size, keywords)
  - Dati opzionali
  - timestamp che rappresenta l'età (limitazione nel numero dei dati diffusi)
- Eventi
  - Public (New Contact, Receive Data Object, …)
  - Private, registrati dai managers (utilizzati per garbage collector e beaconing)
  - Callback, one-time events generati in risposta ad una precedente call (es. ritorno risultati ad una call)
- Data dissemination
  - Utility value e rank per ogni oggetto
  - ContentPlace

Haggle (cont.)

- device mobili, dotati di Wifi (802.11) e interfaccia Bluetooth. Supportata connessione dati GSM (GPRS e/o EDGE), oppure 3G (UMTS/HSDPA)
- Linux, Mac OS X, Android, iPhone (funzionalità limitate), Windows, Windows Mobile
- http://sourceforge.net/projects/haggle/