ALMA MATER STUDIORUM - UNIVERSITA' DI BOLOGNA

CAMPUS DI CESENA

SCUOLA DI INGEGNERIA E ARCHITETTURA

CORSO DI LAUREA IN INGEGNERIA ELETTRONICA, INFORMATICA e TELECOMUNICAZIONI - AMBITO INFORMATICO

Bio-inspired Networking: analisi della letteratura applicata al modello MoK

Relazione finale in

Sistemi Distribuiti

Relatore

Prof. ANDREA OMICINI

Correlatore

Ing. STEFANO MARIANI

Presentata da

ALESSANDRO CORBI

Sessione II di Laurea

Anno Accademico 2017/2018

INDICE

1. Introduzione

Negli ultimi decenni abbiamo assistito alla nascita e allo sviluppo delle tecnologie di comunicazione e informazione, le quali ci hanno fornito numerose architetture di reti informatiche. Uno dei maggiori prodotti di questa evoluzione, Internet, ha mostrato l'enorme potenziale delle reti informatiche in termini di impatto sulla società, economia e qualità della vita.

Allo stesso tempo, i paradigmi di networking convenzionali non sono più sufficienti per affrontare le sfide dettate dalla complessità dei sistemi informatici di rete esistenti, e possibilmente futuri [1].

Molte delle nuove sfide derivanti dall'utilizzo di network su scala sempre più grande (complessità elevata, natura dinamica dei nodi, risorse limitate...) presentano numerose analogie con problemi del mondo biologico, che sono stati brillantemente risolti dalla natura, come risultato di milioni di anni di evoluzione.

Scopo della tesi è quello di esplorare la letteratura in cerca di alternative applicabili rispetto agli approcci tradizionali per quanto riguarda la comunicazione nei sistemi distributi, specialmente nel caso di reti p2p con network altamente dinamiche, derivate da paradigmi bio-inspired.

In particolare verranno analizzati i requisiti di MoK, un modello per la coordinazione di agenti software basato sulla distribuzione di informazioni in rete, e verranno isolate delle proprietà che possiamo ritenere adatte a risolvere le problematiche attaccate da tale modello, per poi ricercare determinati algoritmi, meccanismi e processi che dispongano di tali proprietà per soddisfare i requisiti di Mok.

2. Bio-inspired networking e il modello MoK

2.1 Mok (Molecules of Knowledge)

Da [2]:

> Molecules of Knowledge (MoK for short) is a model for knowledge self-organisation, exploiting the biochemical metaphor for its basic abstractions, and biochemical coordination as its coordination model.

Un sistema Mok può essere visto come una rete di repository in cui vengono depositate informazioni condivisibili, e in cui è possibile aggiungere continuamente e spontaneamente nuovi dati. Questi dati possono aggregarsi autonomamente per costruire nuove strutture più complesse e fornire possibilmente nuove informazioni che prima erano inaccessibili o sconosciute, e diffondersi autonomamente verso possibili consumatori di tali informazioni (invece che essere cercate) [30].

2.1.1 MoK Purpose

Lo scopo del modello MoK [3] è quello di costruire un sistema software che attinga dai sistemi naturali quali ad esempio ispirati alla chimica o alla sociologia umana, per operare in contesti caraterizzati da determinate problematiche dei sistemi distribuiti, quali:

- Networking su larga scala
- Instabilità
- Struttura decentralizzata
- Necessità di soluzioni stocastiche

Il contesto di MoK è per noi molto semplice: si tratta di un sistema distribuito p2p per distribuire/ricercare dati in una rete potenzialmente mobile e instabile. Questo è il nostro livello di astrazione minimo.

Le astrazioni base del modello MoK sono le seguenti:

- **Atomi**: l'unità di conoscenza più piccola in Mok, contiene informazioni di una fonte e appartiene ad un Compartimento, ed è soggetta alle "leggi di natura" di quest'ultimo.

- **Molecole**: l'unità di Mok per l'aggregazione di conoscenza, che unisce insieme gli Atomi che sono in qualche modo "relazionati".
- **Enzimi**: vengono emessi dai Catalizzatori di MoK, rappresentano le azioni dei consumatori e partecipano alle reazioni di Mok per influenzare come Atomi e Molecole si evolvono.
- **Reazioni**: guidano l'evoluzione di ogni Compartimento Mok, controllando in che modo le Molecole si aggregano, sono rinforzate, si diffondo e decadono.
- **Compartimenti**: astrazione spaziale di Mok, rappresentano il loci concettuale per tutte le entità e processi biochimici di Mok, provvedendo anche le nozioni di località e vicinanza.
- **Fonti**: ognuna associata ad un Compartimento, le Fonti rappresentano le origini della conoscenza in Mok, che viene continuamente inserita in Mok attraverso gli Atomi.
- **Catalizzatore**: l'astrazione per i consumatori di conoscenza, i Catalizzatori emettono Enzimi per attrarre gli elementi di conoscenza a cui sono interessati.

Le Reazioni di Mok sono le seguenti:

- **Aggregazione**: con questa Reazione 2 Molecole/Atomi correlati vengono uniti in una nuova Molecola:
  $$molecole\big(Atoms_1\big) + molecole\big(Atoms_2\big) \\ \rightarrow^{r_{agg}} molecole\big(Atoms_1 \uplus Atoms_2\big) + Residual\big(Atoms_1 \cup Atoms_2\big)$$
- **Rinforzo**: Reazione che aumenta la concentrazione di Molecole consumando degli Enzimi conformi dal Compartimento locale: $enzyme(Molecule_1) + Molecule_1^c \rightarrow^{r_{reinf}} Molecule_1^{c+1}$
- **Decadimento**: astrazione che rappresenta il passare del tempo, le Molecole evaporano col passare del tempo: $Molecule^c \rightarrow^{r_{decay}} Molecule^{c-1}$
- **Diffusione**: astrazione che rappresenta lo spazio, le Molecole si diffondono col passare del tempo:
  $$\big\{Molecule_1 \cup Molecules_1\big\}_{\sigma i} + \big\{Molecules_2\big\}_{\sigma ii} \\ \rightarrow^{r_{diffusion}} \big\{Molecules_1\big\}_{\sigma i} + \big\{Molecules_2 \cup Molecule_1\big\}_{\sigma ii}$$

I dati in un sistema Mok possono aggregarsi se vengono rilevati dei "knowledge-related patterns", come ad esempio collegare 2 storie che parlano della stessa persone o scritte dallo stesso autore, o lette dallo stesso consumatore, e diffondersi attraverso gli spazi di rete condivisi di Mok verso altri utenti interessati, ad esempio un documento relativo a MAS dovrebbe cercare di raggiungere i repository dei ricercatori MAS [30].

Gli utenti a loro volta possono interagire col sistema Mok attraverso azioni epistemiche, come leggere un post, contribuire ad una wiki, evidenziare parole in un articolo ecc... Queste azioni sono tracciate dal sistema Mok e usate per influenzare l'evoluzione della conoscenza in maniera trasparente all'utente, ad esempio incrementando la posizione in una query di ricerca di argomenti relativi a parole evidenziate.

Un esempio più "formale" di Mok: degli utenti sono impegnati a compiere del lavoro, come cercare delle news, in questo modo rilasciano degli Enzimi dentro il loro Compartimento, che vengono poi usati per aumentare la concentrazione di Molecole a cui hanno accesso. Nel frattempo, le Molecole non accesse decadono nel tempo, e altre Molecole possono migrare nei Compartimenti vicini, dando la possibilità che altri utenti le rinforzino [3].

2.2 Problematiche

Le problematiche affrontate dal modello Mok sono problematiche comuni a molte tipologie di network odierne [4, 5, 6, 7, 8].

2.2.1 Networking su larga scala

L'aumento di utenti e dispositivi ha portato ad un incremento vertiginoso della dimensione delle reti informatiche, sia in termini di nodi che di volume di dati scambiati, e la portata di questo fenomeno è destinata ad ampliarsi, aumentando esponenzialmente la complessità di tali reti [1]. Riportando alcuni numeri di questi "big data": Google processa dati nell'ordine di centinaia di Petabyte, Facebook genera all'incirca 10 Petabyte di log data al mese, e su Youtube vengono caricati in media 72 ore di video, ogni minuto [9], e molto altro. Questo rende complesso ricercare le informazioni di interesse in una mole non strutturata ed eterogenea di dati, e l'acquisizione deve seguire un processo di profilazione per distinguere l'informazione utile da quella inutile, e un ulteriore processamento in caso di reti con grande rindondanza di dati, come in [5]. Metodi di computazione ispirati da esempi biologici possono essere utilizzati in questo campo, basti pensare che il cervello umano riesce a processare circa un migliaio di TB di dati, e nessun neurone del cervello elabora più velocemente di 1 kHz, che è la velocità media di un computer generico degli anni '80 [22].

In determinate tipologie di reti (wireless sensor network, o WSN), il numero di nodi/sensori può raggiungere le centinaia o migliaia di unità, con una densità di alcune centinaia in pochi metri al quadrato [7], nodi a basso costo con potenza limitata, quindi i nuovi paradigmi dovranno essere sufficientemente scalabili per garantire un'efficiente trasmissione dei dati all'aumentare della complessità della topologia della rete e dei percorsi di instradamento. Queste ultime possono portare inoltre a problemi di overhead e congestione della rete, dovute sia ai problemi di frequenza di trasmissione dei pacchetti di protocollo, che ai problemi di connessione di natura fisica [11, 4].

2.2.2 Instabilità

L'ambiente in cui vengono eseguite le operazioni di networking odierne è spesso dinamico e instabile, in seguito a variazioni nell'ambiente la rete può incorrere in bruschi cambiamenti, che possono alterare la densità di traffico, le capacità di comunicazione e anche la robustezza della rete in seguito ad input erronei/inaspettati [4, 7]. La variazione di piccoli parametri può portare a cambiamenti nel comportamento dell'intero sistema, quindi le nuove reti dovranno presentare caratteristiche di adattività a tali cambiamenti e tolleranza ai guasti in seguito a variazioni. Nell'ottica di Mok, si possono individuare alcune proprietà desiderabili nei nuovi paradigmi ricercati [13]:

- self-configuration per configurare e installare automaticamente nuove procedure e software, per rispondere ai parametri
- self-optimization per permettere ad una rete (mobile) di ottimizzare i propri algoritmi di networking e ottenere performance ottimali
- self-healing per permettere alla rete di recuperare autonomamente operatività in seguito a guasti

2.2.3 Struttura decentralizzata

L'utilizzo di una struttura centralizzata è spesso impossibile per via della dimensione della rete stessa, o dovuta al fatto che la topologia e i nodi della rete sono potenzialmente mobili e instabili. La necessità di una struttura fissa di supporto (es. access point) limita l'adattabilità dei sistemi wireless [12], queste tipologie di rete richiedono quindi una capacità di connessione distribuita, e algoritmi di networking in grado di funzionare senza l'input di un'unità centrale. Tale problema è ricorrente in particolare nel caso di mobile ad hoc networks (MANET), dove ogni nodo della rete può fungere da router e comunicare in modo paritario (P2P) con ogni altro nodo della rete, senza bisogno di un'infrastruttura e tramite connessioni wireless, formando network dinamiche e temporanee [10, 12]. In una ad hoc network il problema della mobilità è particolarmente accentuato poichè ogni nodo è un dispositivo di un utente finale (es. smartphone) e la topologia e la capacità di connessione della rete dipendono dalla posizione degli utenti della stessa. [8] Algoritmi e protocolli di routing che non dipendono dalla topologia della rete sono richiesti per far fronte

alla mobilità e all'aggiunta dinamica di nuovi nodi [7], un paradigma "data centric" invece che "network centric" può essere proposto come possibile direzione [11].

2.2.4 Necessità di soluzioni stocastiche

Il mondo naturale è un mondo in cui i processi sono principalmente di natura stocastica, che diventano necessità quando il centro del modello si sposta dalle reti alle azioni degli utenti, le cui azioni/ricerche in una rete diventano difficili da prevedere, e le soluzioni deterministiche non funzionano più [21].

I modelli stocastici, benchè di più difficile implementazione, risultano anche essere adatti a risolvere problemi di ottimizzazione in grandi reti di dati [22], e sono necessari per svolgere analisi statistiche ed estrarre informazioni utili dai dati, organizzarli e interpretarli.

Metodi probabilistici possono essere anche usati per la predizione di link futuri in analisi strutturali link-based, dove i linked data sono principalmente strutture grafiche che descrivono la comunicazione fra 2 entità, e, insieme ai content data, costituiscono il fulcro dei dati e dell'analisi nei servizi di social networking (Twitter, Facebook, LinkedIn ecc...) [9].

3. Paradigmi bio-inspired

In questa sezione verranno analizzati i paradigmi bio-inspired presenti nella letteratura che rispondono alle problematiche di Mok e possono fornire possibili soluzioni ai problemi citati.

3.1 Swarm Intelligence

Col termine Swarm Intelligence si indica una famiglia di paradigmi bio-inspired sviluppato osservando il comportamento di alcune tipologie di insetti sociali, formiche/api in particolare. Possiamo definire "Swarm Intelligence" come la proprietà di un sistema in cui l'interazione collettiva dei suoi componenti (semplici) con quella dell'ambiente causa l'emergere di comportamenti ricorrenti funzionali complessi sulla scala globale del sistema [14]. Ad esempio, questo comportamento emergente presente nelle colonie di formiche, in cui ogni individuo può essere visto come un agente con semplici funzionalità e limitate capacità di apprendimento, permette lo svolgimento di operazioni complesse usando solo semplici strumenti e informazioni locali (ricerca del cibo, divisione dei compiti, costruzione di nidi, trasporto cooperativo, ecc...) [1, 13].

3.1.1 ACO (Ant Colony Optimization)

Negli algoritmi di tipo ACO una colonia di formiche artificiali cooperano tra di loro per ricercare una soluzione a problemi di ottimizzazione discreta, ad esempio esplorare una network e comunicare alle altre formiche le informazioni acquisite (come farebbero le formiche reali alla ricerca del cibo). La ricerca cooperativa del "cibo" avviene tramite un metodo di comunicazione definito "stigmergia", in cui gli individui di un sistema comunicano tra loro modificando l'ambiente circostante. Nel caso delle formiche, queste comunicano indirettamente tra di loro lasciando tracce di feromone nell'ambiente, ed è stato osservato che le formiche sono in grado di trovare il percorso verso il cibo seguendo tali tracce, che risultano avere maggiore intensità sul percorso più breve [13].

Gli agenti comunicano quindi in maniera indiretta e fanno uso di un sistema di decisione stocastica per trovare la miglior soluzione al problema, usando delle politiche di rinforzo (elitist ant) per gli agenti e i path con risultati migliori [16, 17].

Queste caratteristiche di controllo decentralizzato, decision-making senza informazioni sull'ambiente globale ed efficiente ripartizione delle risorse sono estremamente desiderabili nei sistemi distributi di grandi dimensioni, e costituiscono un obbiettivo di progettazione in wireless sensor networks e ad hoc networks [15]. Nelle sottosezioni seguenti verranno analizzati alcuni dei più importanti algoritmi ACO che soddisfano i requisiti di Mok.

3.1.1.1 AS (Ant System)

AS è un algoritmo general purpose creato con lo scopo di risolvere il problema del commesso viaggiatore (Travel Salesman Problem, o TSP) per trovare il percorso più breve tra una serie di città. Può essere visto come il precursore degli algoritmi ACO, ed è stato il prototipo iniziale di vari algoritmi ant-based che si sono sviluppati nel tempo e hanno ottenuto successo [16].

A differenza delle formiche reali, le formiche artificiali usate disporranno di una limitata memoria, non saranno completamente cieche e si troveranno in un ambiente in cui il tempo è discreto.

L'algoritmo è descritto come segue [17]:

Dato un set di n città, d_{ij} è la lunghezza del percorso tra la città i e j, $b_i(t)$ rappresenta il numero di formiche nella città i al tempo t, e $m = \sum_{i=1}^{n} b_i(t)$ sarà il numero totale delle formiche.

Ogni formica possiede le seguenti caratteristiche:

- sceglie in quale città andare con una probabilità in funzione del percorso più breve e della quantità di feromone più elevata che si trova sul percorso.
- ogni formica ha un tabu list in cui tiene traccia delle città già visitate e non può rivisitarle fino al completamento del tour attuale (un tour è completato quando vengono visitate tutte le città).
- una volta completato un tour, la formica deposita su ogni percorso visitato (i, j) una traccia di feromone (più è breve il percorso più forte sarà la traccia depositata).

La probabilità di scelta di un determinato percorso (i, j) è formalmente definita come:

$$p_{ij}^{k}(t) = \begin{cases} \frac{\left[\tau_{ij}(t)\right]^{\alpha} \cdot \left[\eta_{ij}\right]^{\beta}}{\sum_{k \in allowed_k} \left[\tau_{ik}(t)\right]^{\alpha} \cdot \left[\eta_{ik}\right]^{\beta}} & if(j) \in allowed_k \\ 0 & otherwise \end{cases}$$

dove $allowed_k = \{N - tabu_k\}$ (le città ancora da visitare), $\alpha \geq 0$ e $\beta \geq 0$ sono parametri che rappresentano rispettivamente l'importanza relativa della traccia e della visibilità rappresentata da η_{ij}, che è l'inverso della distanza tra 2 città $\frac{1}{d_{ij}}$.

A tempo t ogni formica sceglie il percorso da seguire per la prossima città dove si troverà a $t + 1$, e una volta completato il tour, verrà depositato su ogni percorso scelto la traccia:

$$\Delta\tau_{ik}^{k} = \begin{cases} \frac{Q}{L_k} & se(i, j) \in T^k(t) \\ 0 & otherwise \end{cases}$$

dove Q è una costante relativa alla quantità di feromone rilasciato dalle formiche reali, e L_k la lunghezza del tour fatto.

A fine tour, l'intensità totale della traccia sarà:

$$\tau_{ij}(t + n) = \rho \cdot \tau_{ij}(t) + \Delta\tau_{ij}$$

dove $0 \leq \rho < 1$ è il coefficiente che rappresenta la persistenza della traccia, che evapora di $(1-\rho)$ tra il periodo t e $t+n$.

Rappresentazione dell'algoritmo dopo vari cicli, che ci mostra gli effetti della proprietà di stigmergia, comune a tutti gli algoritmi ACO:

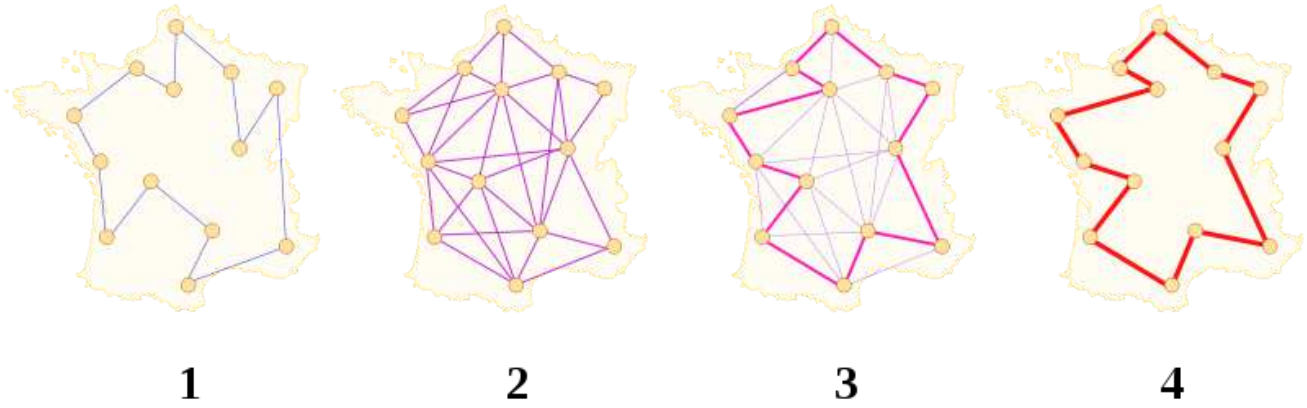


Figura 0 [Author: Johann Dréo (User:Nojhan)]

1. Una formica sceglie delle tappe e completa un tour, depositando feromone sui percorsi.
2. Altre formiche completano il proprio tour, un numero di possibili percorsi viene delineato.
3. Dopo varie iterazioni dell'algoritmo, la traccia del percorso migliore comincerà a rafforzarsi e sempre più formiche cominceranno a sceglierlo con maggiore probabilità.
4. Inevitabilmente, meno formiche sceglieranno gli altri percorsi, e la traccia sui percorsi peggiori inizierà ad evaporare, e il percorso più corto sarà quello scelto dalle formiche.

I risultati delle simulazioni di AS sono stati ottimali per problemi di dimensione ridotta (30 città circa), in cui AS è risultato migliore di altre euristiche general purpose, mentre per problemi di grandi dimensioni sono state trovate buone soluzioni ma non è stata raggiunta la soluzione ottimale a differenza di altri algoritmi special purpose [17]. I risultati ottenuti hanno comunque spinto la ricerca verso lo sviluppo di altri algoritmi ACO più sofisticati.

3.1.1.2 AntNet

AntNet è un algoritmo di routing per packet switched IP network (wired), ed è stato uno dei primi algoritmi a seguire l'approccio di Swarm Intelligence, basandosi sui meccanismi del framework ACO per l'ottimizzazione combinatoria e il suo shortest past behaviour basato sulle colonie di formiche [18].

Rispetto ad Ant System, oltre a presentare le caratteristiche tipiche degli algoritmi ACO, AntNet applica anche delle strategie di reinforcement learning, risultando più adattivo e più robusto ai cambiamenti dinamici della rete.

In una rete AntNet, ogni nodo k tiene traccia di 2 strutture dati [19]:

- Una tabella di routing T_k che contiene, per ogni destinazione d, un vettore con una elemento per ogni nodo n vicino; Per ogni elemento viene indicato un valore di probabilità P_{nd} ($\sum_{n \in N_k} P_{nd} = 1$) che ne esprime la desiderabilità.
- Un array $M_k(\mu_d, \sigma_d^2, W_d)$ di strutture dati che definiscono un modello statistico per individuare la distribuzione del traffico nella rete, per come viene vista dal nodo k. W_d viene usato per contenere il

miglior valore di trip time degli agenti (il tempo impiegato da un agente per attraversare un percorso tra 2 nodi), mentre μ_d e σ_d^2 sono una media e una varianza per stimare il tempo di viaggio e l'accuratezza di tale valore.

Ad intervalli regolari Δt ogni nodo s manda in esplorazione un agente mobile (forward ant) verso un nodo destinazione d per scoprire il percorso più breve per raggiungerlo e per indagare sullo stato del traffico della rete (usando la stessa coda dei normali datagrammi, in modo da sperimentare la stessa densità di traffico). La scelta della destinazione è basata sul data flow f_{sd} verso di essa (numero di datagrammi o bits) e la probabilità di tale scelta è data da:

$$p_d = \frac{f_{sd}}{\sum_{d'=1}^{N} f_{sd'}}$$

Gli agenti tengono traccia del percorso seguito e dello stato del traffico verso i nodi destinazione, attraverso uno stack di memoria, ed ogni hop verso un nodo n intermedio viene scelto basandosi sulla tabella di routing del nodo attuale k, usando la tabella di probabilità per la scelta dei nodi vicini non attraversati (o per tutti i nodi nel caso siano stati tutti visitati), applicando una correzione euristica:

$$P'_{nd} = \frac{P_{nd} + \alpha l_n}{1 + \alpha\left(|N_k| - 1\right)}$$

dove la correzione euristica $l_n = 1 - \frac{q_n}{\sum_{n'=1}^{|N_k|} q_{n'}}$

tiene conto della lunghezza della coda di k verso n, mentre α è un valore che serve a soppesare l'importanza della correzione euristica contro il valore probabilistico P_{nd} nella routing table.

Se la forward ant entra in un ciclo in cui inizia a rivisitare dei nodi in cui è già stata in precedenza, le informazioni su questi nodi vengono cancellate, e se il ciclo risulta essere più lungo della metà della vita della formica, essa viene distrutta, questo per evitare che vengano usate informazioni dovute a scelte erronee dell'agente, se invece la formica raggiunge la sua destinazione d, essa genere un altro agente (backward ant) e gli trasferisce le sue informazioni in memoria, dopodichè la forward ant si distrugge.

La backward ant percorre lo stesso percorso seguito dalla forward ant, ma in direzione opposta, in modo da aggiornare le routing table dei nodi usando le informazioni raccolte dalla forward ant (usando una coda ad alta priorità, in modo da propagare velocemente le informazioni).

Arrivando ad ogni nodo k sul percorso inverso, da un nodo f, vengono aggiornate le entry per il nodo destinazione d: vengono aggiornate le informazioni sul traffico basandosi sul trip time t per ogni nodo vicino k' percorso dalla formica, usando una variabile di rinforzo positivo $r \equiv r\left(T, M_k\right)$ (dove T rappresenta il tempo di attraversamento di tutti i sotto-percorsi) per indicare i sotto-percorsi migliori nella tabella di routing:

$$P_{fd'} \leftarrow P_{fd'} + r\left(1 - P_{fd'}\right)$$

di conseguenza le altre informazioni ricevono un rinforzo negativo

$$P_{nd'} \leftarrow P_{nd'} - rP_{nd'}.$$

Come nota finale, AntNet è stato testato insieme ad altri algoritmi considerabili allo stato dell'arte (OSPF, SPF, BF, Q-R, PQ-R) su 3 reti di comunicazione reali di varie dimensioni, in cui AntNet si è dimostrato spesso superiore agli altri algoritmi, dimostrando un basso e costante packet delay molto vicino a valori ideali, soprattutto sulla rete di maggiori dimensioni [19].

3.1.1.3 AntHocNet

A differenza dei primi algoritmi di ACO, come AntNet, pensati per reti wired, AntHocNet [10] mantiene le stesse caratteristiche che rende ACO desiderabile, ma sfrutta delle caratteristiche aggiuntive per cercare il percorso migliore, in modo da ridurre l'overhead ed essere più adatto all'uso in reti wireless.

La differenza principale si riscontra nella strategia di esplorazione dei nodi: nelle reti wired viene usato un comportamente principalmente proattivo, in cui si tiene traccia di tutti i nodi della rete, comportamento costoso da mantenere nel caso di una rete mobile, in cui la topologia dei nodi cambia continuamente.

AntHocNet utilizza una strategia ibrida multipath, ossia fa uso di componenti sia reattivi che proattivi, non mantenendo una tabella di routing per tutte le destinazioni in ogni singolo momento, ma settando i path necessari alla comunicazione verso gli altri nodi all'inizio di una sessione.

All'inizio di una sessione di comunicazione con un nodo destinazione d, avviene un reactive path setup: un source node s broadcast una **reactive forward ant** F_d^s; ogni nodo vicino i riceve una copia di F_d^s, e se il nodo dispone nella sua tabella di routing T^i delle informazioni per raggiungere la destinazione d attraverso un nodo intermedio n, allora la formica viene instradata su quel percorso, altrimenti viene trasmessa di nuovo tramite broadcast ai nodi vicini.

La probabilità che una formica venga instradata da i a n per raggiungere d ($T_{nd}^i \in \Re$) è:

$$P_{nd} = \frac{\left(T_{nd}^i\right)^\beta}{\sum_{j \in N_d^i} \left(T_{jd}^i\right)^\beta}, \beta \geq 1$$

dove N_d^i è il set di nodi i che ha informazioni sul path per d, e β è un parametro per controllare il comportamento esploratorio delle formiche. Anche i datagrammi useranno sempre questa formula per scegliere il path, usando un parametro β più elevato per influenzare la probabilità di scegliere i path migliori.

Per ridurre overhead, ogni generazione di forward ant che raggiunge un nodo viene sottoposta ad un test di valutazione sul trip time e numero di hop conseguito dall'agente, se questi valori non rientrano in un acceptance factor a_1, basato sulle performance della miglior formica della generazione precedente, allora la formica viene scartata. Per ottenere un buon numero di path usabili, per motivi di fault tolerance, si può usare un valore di acceptance a_2 meno restrittivo, per quelle formiche che hanno fatto un hop verso un nodo diverso da quello delle generazioni precedenti.

Una volta giunta a destinazione, una formica si trasforma in una **reactive backward ant** (come visto già per AntNet), e ripercorre il percorso inverso per aggiornare le tabelle stocastiche di routing dei nodi, in caso un nodo del percorso non sia più raggiungibile per via della mobilità o altri motivi, la backward ant viene distrutta.

Il valore della tabella viene aggiornato come segue:

$$T_{nd}^i = \gamma T_{nd}^i + (1 - \gamma)\tau_d^i$$

dove γ è un parametro settato a 0,7 (si vedrà negli esperimenti successivi) e τ_d^i è la traccia di feromone apportata dalla backward ant:

$$\tau_d^i = \left(\frac{\hat{T}_d^i + hT_{hop}}{2}\right)^{-1}$$

T_{hop} rappresenta il tempo necessario a fare un hop verso un nodo in condizioni di poco traffico, h è il numero di hop, e $\hat{T}_d^i$ è il trip time calcolato dalla formica.

Inoltre, ogni n datagrammi inviati, il nodo fonte invia anche delle **proactive forward ant**, che si muovono stocasticamente (seguono sempre la formula P_{nd}) come le reactive forward ant, ma ad ogni hop hanno una probabilità (0,1 negli esperimenti) di essere diffusi in maniera broadcast ai nodi vicini, in modo da continuare a esplorare la rete alla ricerca di nuovi path e aggiornare con frequenza le tabelle dei nodi già esistenti.

Per aiutare le formiche e confermare la presenza dei nodi vicini, un nodo invia tramite broadcast dei messaggi ogni t secondi, quando un nodo riceve tale messaggio, aggiunge il mittente alla sua routing table, se invece il nodo n non risponde al messaggio entro un determinato periodo di tempo, o una formica fallisce nel raggiungerlo, il nodo viene cancellato dalla routing table e si avvia una procedura di link failure:

il nodo che ha rilevato l'inagibilità di un percorso, invia tramite broadcast una notifica in cui comunica quale nodo n per quale destinazione d è andato perso, e il nuovo miglior percorso agibile; se il nodo perso risultava il miglior percorso per altri nodi, questi comunicano a loro volta la notifica ai loro vicini.

Se il percorso inagibile risultava l'unico percorso possibile per l'invio di datagrammi verso una destinazione d, il nodo cerca localmente di riparare il path inviando una **path repair ant** (il comportamento è quello degli altri agenti, ma ne viene limitata la possibilità di broadcast), se non viene ricevuta una **backward repair ant** il percorso viene ritenuto non riparabile, i datagrammi da instradare su quel percorso vengono scartati e una notifica di link failure viene inoltrata ai nodi vicini per aggiornarne le politiche di routing.

Risultati delle simulazioni [10]:

AntHocNet è stato comparato ad AODV, un algoritmo di routing allo stato dell'arte che funge da standard in MANET. I test sono stati svolti in 2 setting diversi, ognuno svolto con 5 iterazioni con posizione dei nodi e pattern di movimento diversi.

1. Nel primo setting di test sono stati posizionati 100 nodi in un'area di 3000 * 1000 m^2, con una durata di 900 s. Il traffico di dati è generato da 20 constant bit rate (CBR) fonti che spediscono 1 pacchetto da 64 byte ogni secondo, data rate a 2 Mbit/s. Nel modello di propagazione radio viene usato un two-ray pathloss model, il range radio dei nodi è di 300 m e al MAC layer viene usato il protocollo DCF 802.11b. I test sono stati eseguiti usando il modello di mobilità RWP (random waypoint) [28] con velocità massima e tempo di pausa variati, e col modello di mobilità GM (Gauss-Markov) [29] con velocità massima variata. La frequenza di update è stata settata a 2,5, l'angolo standard di deviazione a 0,5. Gli scenari di GM sono stati generati con BonnMotion software.
2. Nel secondo setting è stato usato il setup per gli esperimenti di AODV [20], sono stati fatti esperimenti con 100, 500, 1000 e 1500 nodi rispettivamente in aree quadrate con lati di 1500, 3500, 5000 e 6000 m, mantenendo la densità dei nodi costante (7,5 circa), e di durata 500s. Il traffico di dati è generato da 20 constant bit rate (CBR) fonti che spediscono 4 pacchetti da 512 byte ogni secondo, data rate a 2 Mbit/s. I nodi si muovono in accordo col modello RWP [28], con velocità minima a 0 m/s e massima di 10 m/s e periodo di pausa di 30 s, il range radio è di 250 m, con free space model (sempre un pathloss model), e al MAC layer viene usato il protocollo DCF 802.11b.

I risultati delle 5 iterazioni sono riassunti dai seguenti grafici:

- GM Mobility

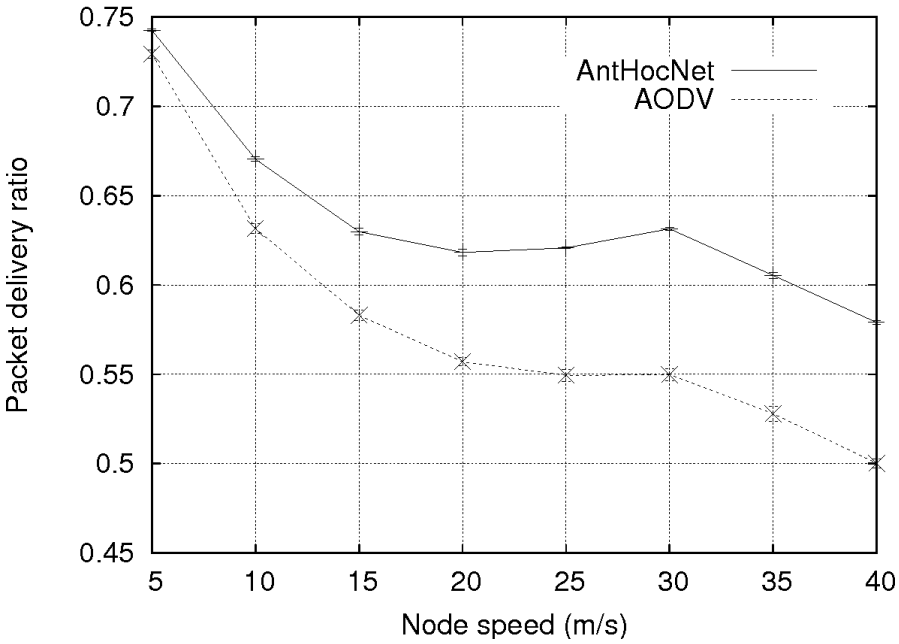


Figura 1[10]: Frequenza di consegna dei pacchetti con nodi a varie velocità nel modello GM.

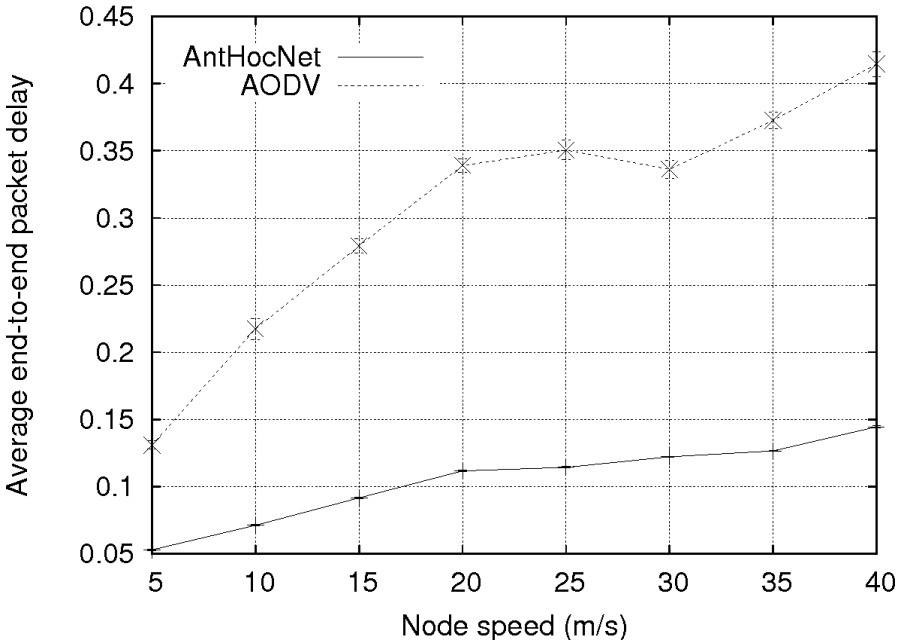


Figura 2[10]: Ritardo di consegna dei pacchetti con nodi a varie velocità nel modello GM.

- RWP Mobility

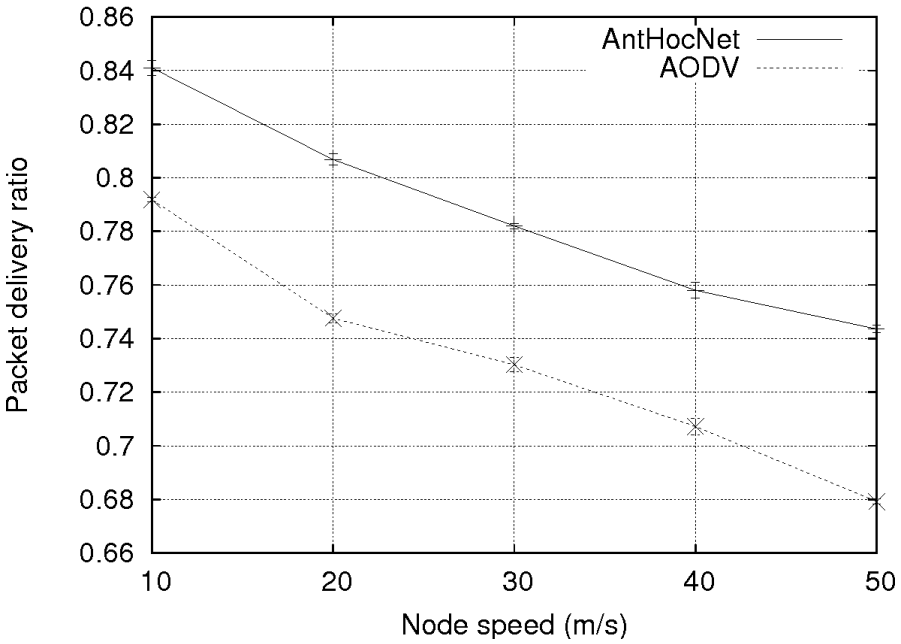


Figura 3[10]: Frequenza di consegna dei pacchetti con nodi a varie velocità nel modello RWP.

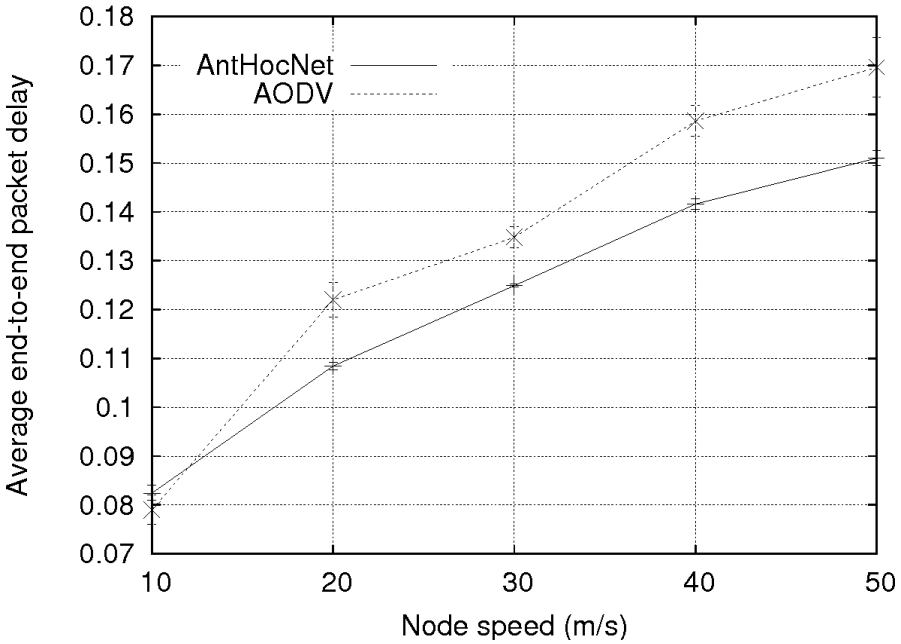


Figura 4[10]: Ritardo di consegna dei pacchetti con nodi a varie velocità nel modello RWP.

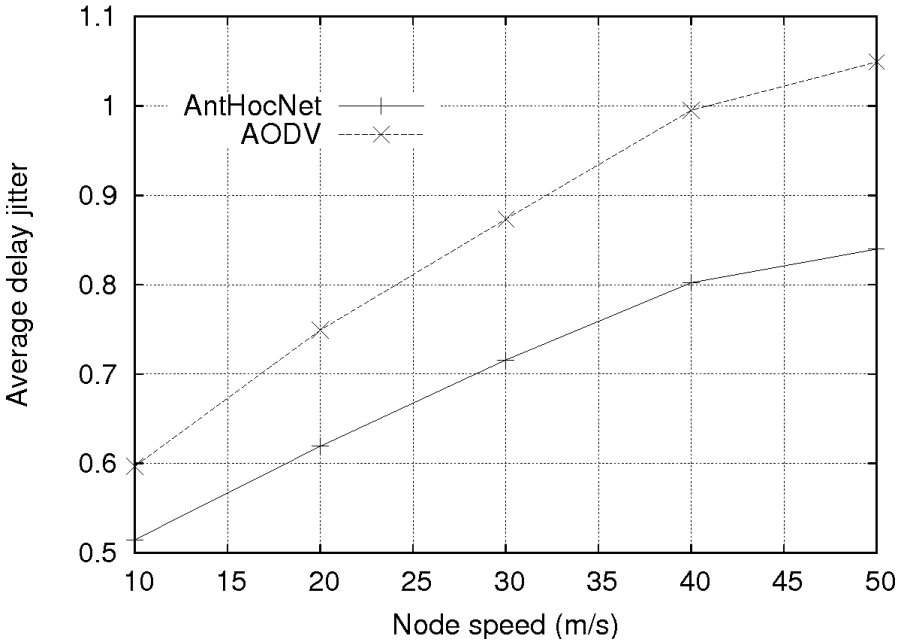


Figura 5[10]: Variazione media del ritardo di consegna con nodi a varie velocità nel modello RWP.

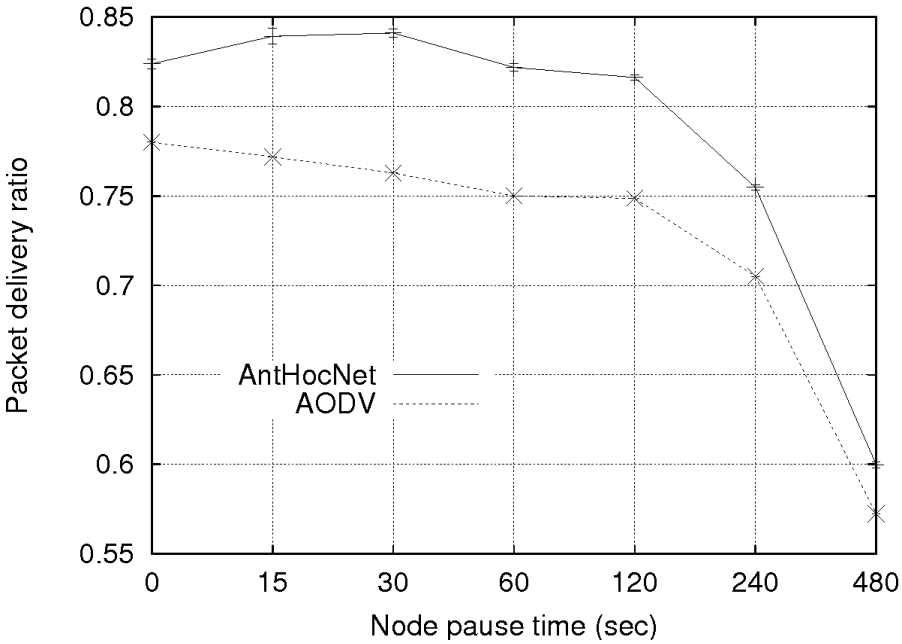


Figura 6[10]: Frequenza di consegna dei pacchetti con nodi fermi nel modello RWP.

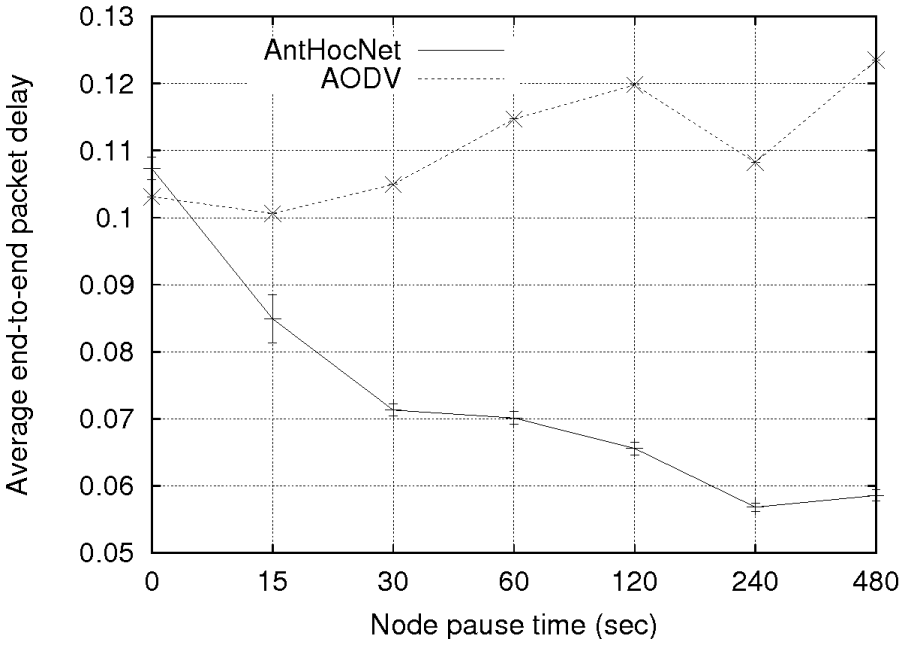


Figura 7[10]: Ritardo di consegna dei pacchetti con nodi fermi nel modello RWP.

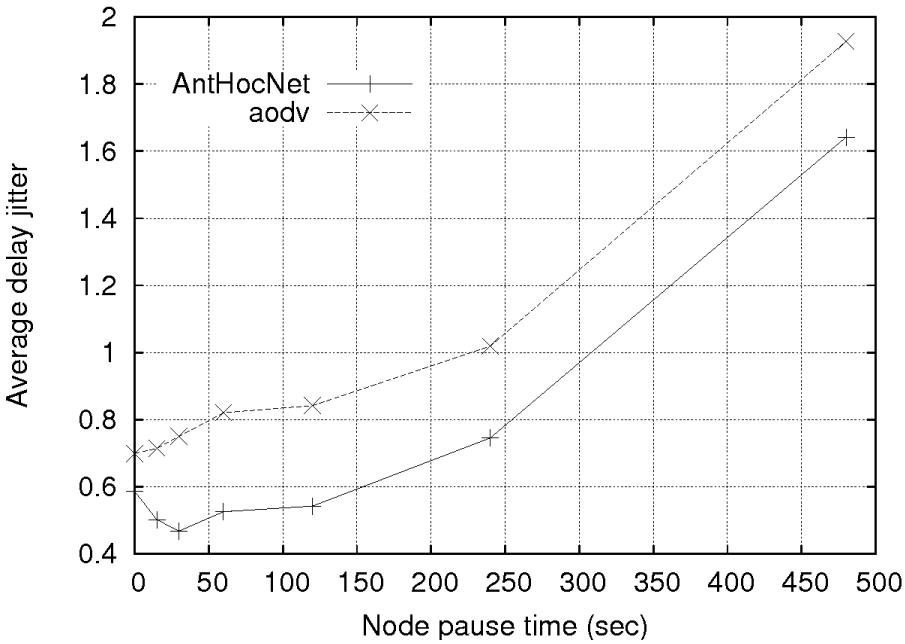


Figura 8[10]: Variazione media del ritardo di consegna con nodi fermi nel modello RWP.

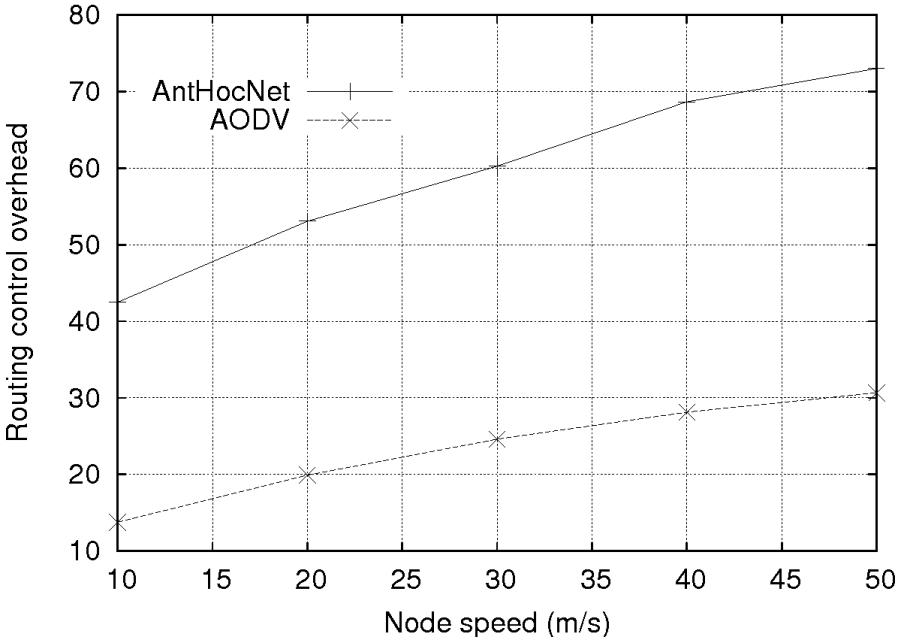


Figura 9[10]: Overhead della rete con nodi a varie velocità nel modello RWP.

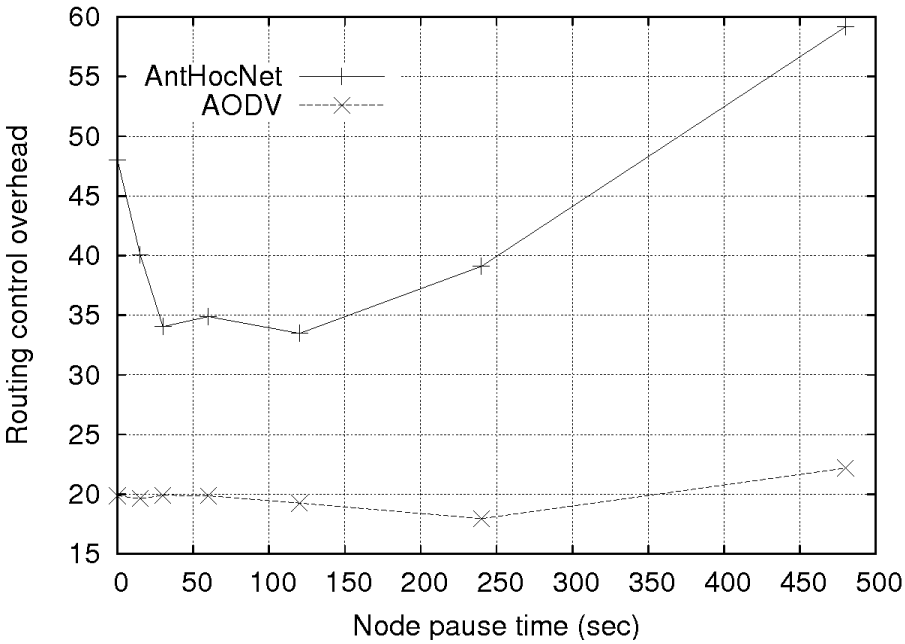


Figura 10[10]: Overhead della rete con nodi fermi nel modello RWP.

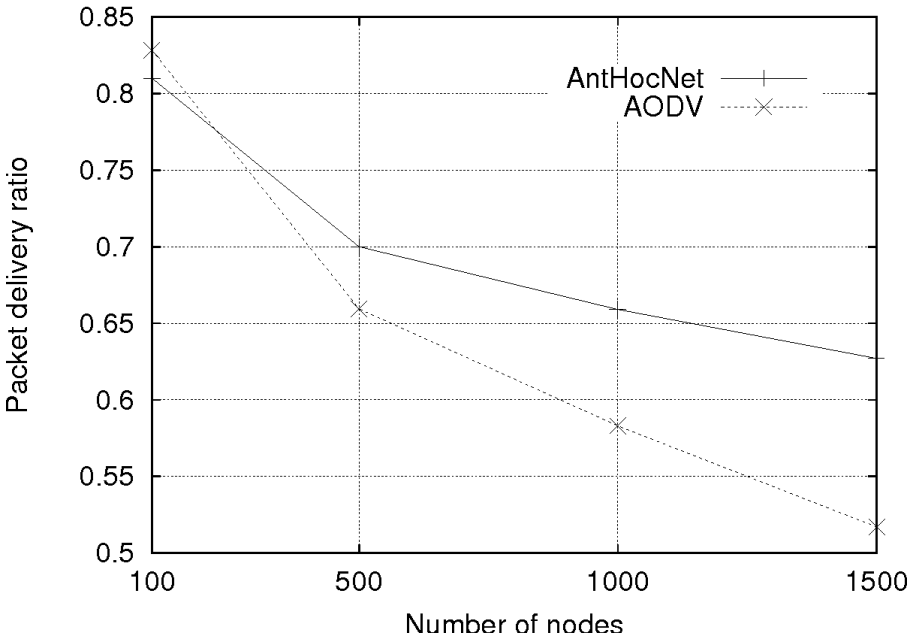


Figura 11[10]: Frequenza di consegna dei pacchetti in base al numero di nodi nel RWP.

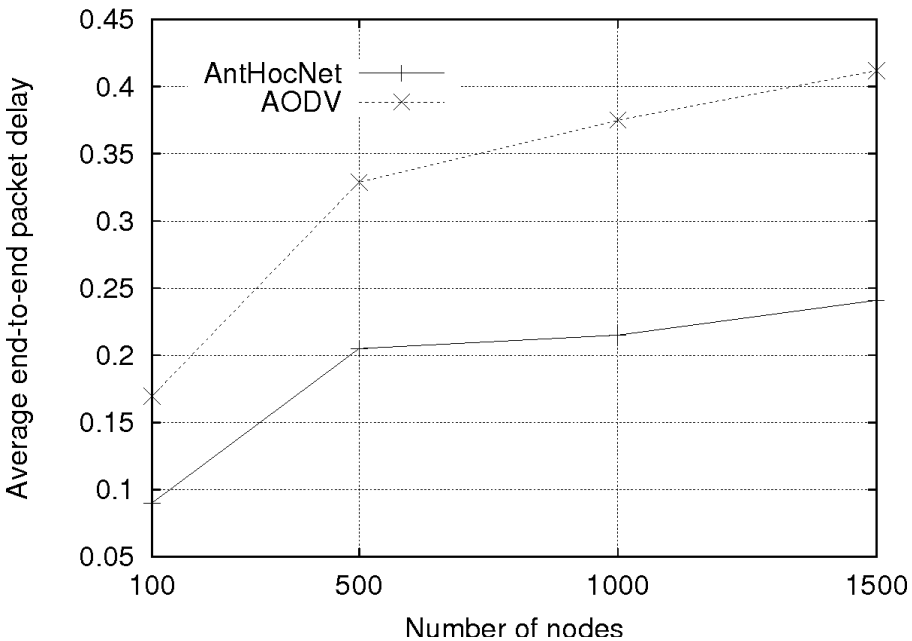


Figura 12[10]: Ritardo di consegna dei pacchetti in base al numero di nodi nel RWP.

Come si può osservare dai risultati, AntHocNet risulta migliore di AODV in tutti i modelli, soprattutto in delivery ratio and jitter, mentre nel delay la differenza è inferiore ma AntHocNet risulta migliore a velocità dei nodi più elevate. Il fatto che questi valori peggiorino durante i periodi di pausa dei nodi, può indicare che certi nodi si siano fermati in posizioni più lontane rispetto al resto della network, essendo la densità di nodi piuttosto sparsa.

Queste prestazioni migliori vengono a discapito del fattore di overhead, che in AntHocNet risulta più elevato di AODV, per via della generazione e del lavoro di esplorazione delle formiche. Questo fattore di overhead può essere potenzialmente migliorato riducendo il numero di agenti o migliorando l'efficienza delle formiche proattive.

Si può evincere quindi che AntHocNet performa meglio in ambienti in cui la mobilità dei nodi è maggiore, con nodi più sparsi e percorsi più lunghi, fattori per noi molto interessanti date le problematiche di instabilità e mobilità previste dal modello MoK, dove AntHocNet può essere usato nelle operazioni di routing tra 2 nodi, e trovare il miglior percorso usabile per cercare/inviare informazioni.

3.2 Epidemic Spreading

Paradigma bio-inspired basato sulla trasmissione dei virus reali e delle epidemie, la sua applicazione principale è nella network security, dove viene usato come paradigma per arginare la disseminazione dei virus informatici (malware). Un altro utilizzo "benefico" in cui viene usato è lo studio della disseminazione di informazioni nelle network informatiche [1], che rientra nel contesto dei requisiti del modello Mok.

Il concetto fondamentale di epidemic spreading è la distribuzione di informazioni verso una destinazione attraverso la spedizione di un pacchetto che "infetta" gli altri nodi: quando un nodo spedisce un pacchetto ad un nodo che non dispone di tale pacchetto, il pacchetto viene messo nel suo buffer e il processo continua, fino a che il pacchetto non raggiunge la destinazione voluta. Quando la congestione della rete è sufficientemente bassa, questo paradigma riesce a raggiungere tempi di delay di spedizione minimi a costo di un maggiore consumo di spazio di buffer nei nodi, larghezza di banda e potenza di trasmissione [23].

Verranno prese in considerazione alcune variazioni di questo paradigma, che presentano diversi tradeoff tra delay di spedizione e consumo di risorse:

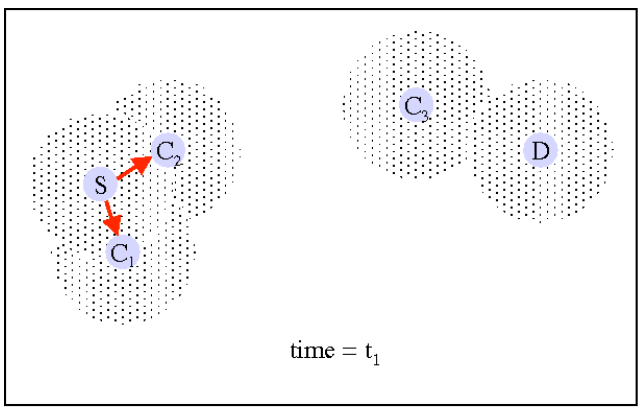
3.2.1 Epidemic Routing

Epidemic Routing [24] è un protocollo per reti mobili sviluppato sulla base del paradigma di Epidemic Spreading per:

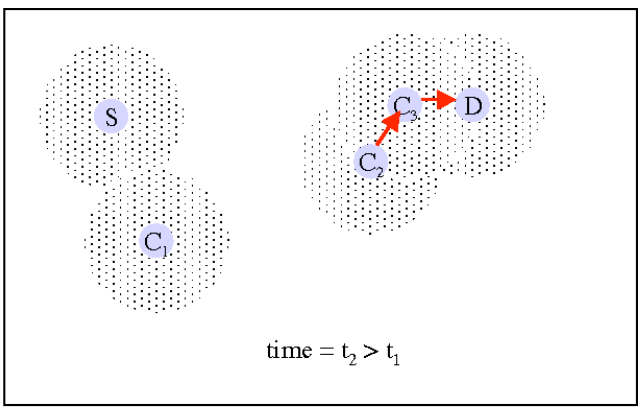
- Distribuire messaggi in una rete ad hoc parzialmente connessa, in maniera probabilistica
- Minimizzare il consumo di risorse nella consegna di ogni singolo messaggio
- Massimizzare la percentuale di messaggi che vengono eventualmente consegnati alla propria destinazione

Il punto di interesse di questo protocollo è il fatto che il messaggio arrivi **eventualmente** alla sua destinazione, cioè non è necessario che il nodo destinazione si trovi entro il campo di ricezione degli altri nodi, e che quindi sia possibile stabilire un percorso verso di esso.

Sfruttando il processo di "infezione" di Epidemic Spreading, un nodo fonte passa il suo messaggio ai nodi vicini, che lo conservano nel proprio buffer, e grazie alla mobilità di nodi, si porterà eventualmente entro il campo di ricezione del nodo destinazione, consegnando il messaggio.



(a)



(b)

Figura 13[24]: a) Un nodo fonte S vuole inviare un messaggio a D, non potendo raggiungere la destinazione, infetta i nodi vicini C1 e C2. b) Il nodo C2 eventualmente si sposta in prossimità di D, inviando il messaggio infettando un nodo vicino C3.

Ogni nodo mantiene un buffer che contiene i messaggi originati da sè stesso, e i messaggi trasportati per conto di altri nodi, indicizzati in una struttura hash con degli identificatori, e un vettore di bit, detto summary vector, che indica quali identificatori sono settati. Quando 2 nodi entrano in contatto, il nodo con l'identificatore più basso inizia una "anti-entropy session" (descritta in [25]) con il nodo con identificatore più grande, e i messaggi non presenti in lista vengono scambiati reciprocamente.

Ogni nodo ha totale autonomia su quali messaggi accettare, in base alle proprie policy (dimensione del messaggio, destinatario, ecc...).

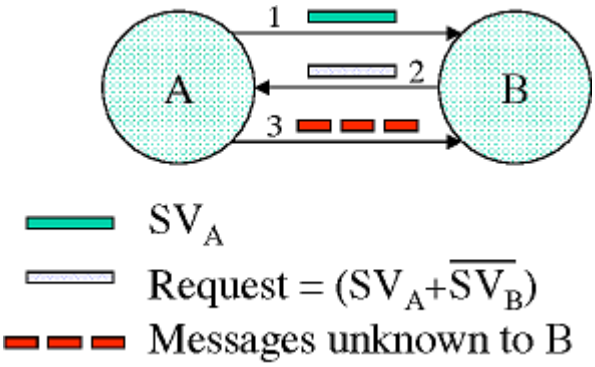


Figura 14[24]: 2 nodi che comunicano attraverso il protocollo Epidemic Routing appena entrano in contatto.

Dalla Figura 14: quando il nodo A entra in contatto con il nodo B, viene iniziata una comunicazione, A trasmette il suo summary vector SV_A a B, che performa un NAND logico tra il suo vettore negativo $\neg SV_B$ e SV_A, e tramite la differenza del set, B richiede i messaggi di cui non dispone, che verranno forniti da A.

Risultati simulazione [24]:

Ogni nodo usa un "agente epidemico", layerizzato sopra l'Internet MANET Encapsulation Protocol (IMEP) layer, che consiste in un buffer per contenere i messeggi, un summary vector per il buffer e il codice necessario ad iniziare le sessioni di comunicazione anti-entropy. Il layer IMEP informa l'agente quando un altro nodo entra od esce dal raggio di comunicazione del campo radio.

Lo scenario di simulazione consiste in 50 nodi mobili in un'area di 1500*300 m, ogni nodo si muove in un punto casuale dell'area con una velocità uniformamente distribuita tra 0-20 m/s (velocità media di 10 m/s) e ogni messaggio è lungo 1 KB. Un sottogruppo dei 50 nodi è scelto come fonte/destinazione, con 45 nodi che mandano ognuno un messaggio ad altri 44, per un totale di 1980 messaggi. Ogni nodo ha un buffer in grado di contenere 2000 messaggi, quindi la capacità può essere considerata infinita, ma verranno sperimentati anche buffer limitati.

Range	Delivery Rate (%)	Baseline Rate	Latency		Hops		Coverage Floor
			Avg (s)	Max (s)	Avg	Max	
250 m	100.0	98.2	0.2	1	2.4	8	10.91%
100 m	100.0	34.3	12.8	177	6.3	21	1.75%
50 m	100.0	0.9	153.0	760	3.7	14	0.44%
25 m	100.0	0.0	618.9	3758	3.3	9	0.11%
10 m	89.9	0.0	44829.7	198107	3.4	9	0.02%

Figura 15[24]: Epidemic Routing in funzione del raggio di trasmissione dei nodi.

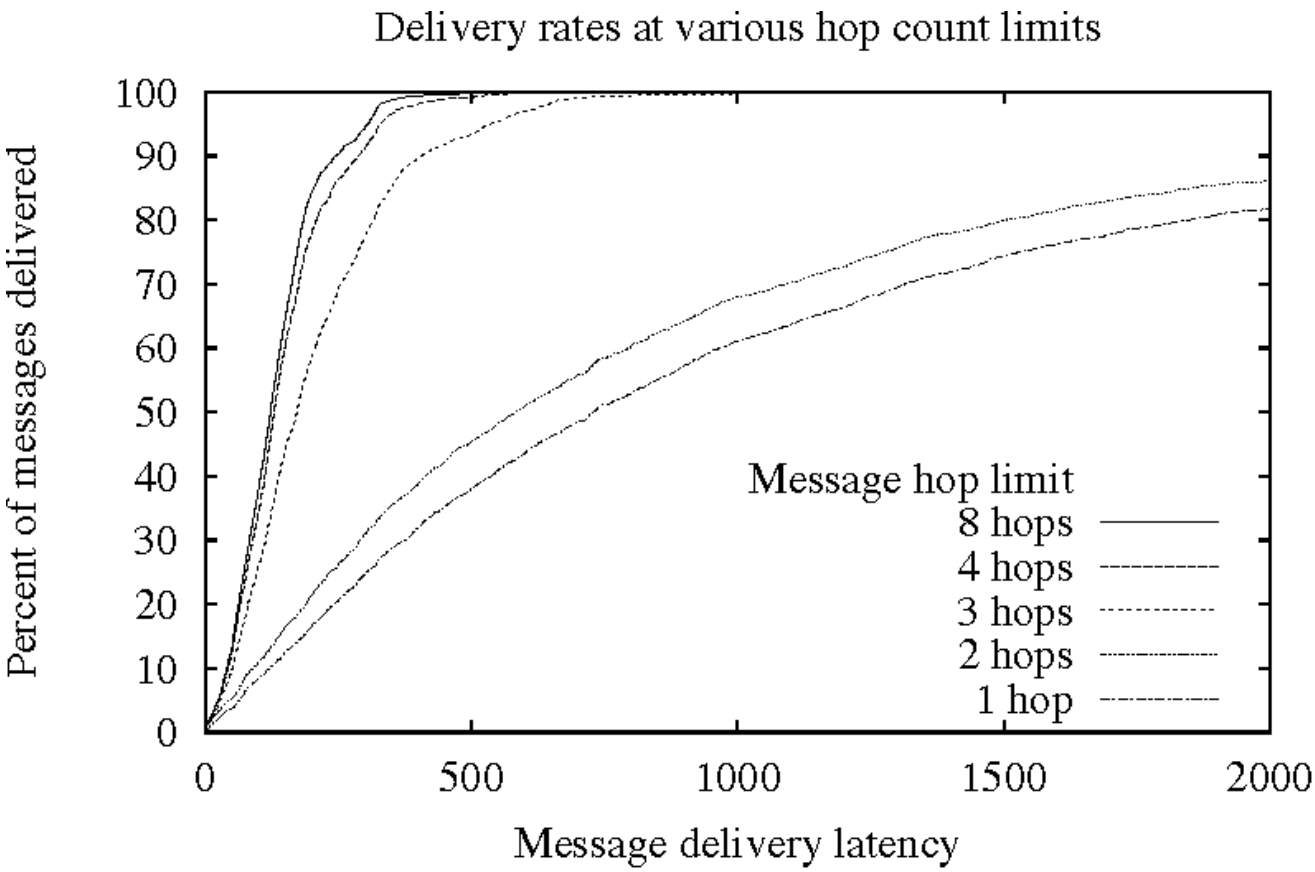


Figura 16[24]: CDF in funzione della percentuale di messaggi consegnati per numero di hop, con raggio di trasmissione di 50 metri.

Coverage Floor rappresenta la copertura dell'area di disseminazione dei nodi in base al raggio di trasmissione, mentre Baseline Rate è la percentuale di messaggi trasmessi con successo usando il protocollo DSR descritto in [25].

I valori ottenuti nel raggio di 25 e 50 m risultano molto interessanti poichè entro questo raggio molti protocolli non riuscirebbero a garantire un percorso stabile per distribuire i messaggi, mentre Epidemic Routing riesce eventualmente a consegnare i messaggi con buona probabilità, anche con copertura molto limitata.

Altro fattore importante da tenere in considerazione è la dimensione del buffer di ogni nodo, che richiede dovute policy dei messaggi e ottimizzazione per garantire un funzionamento ottimale di Epidemic Routing:

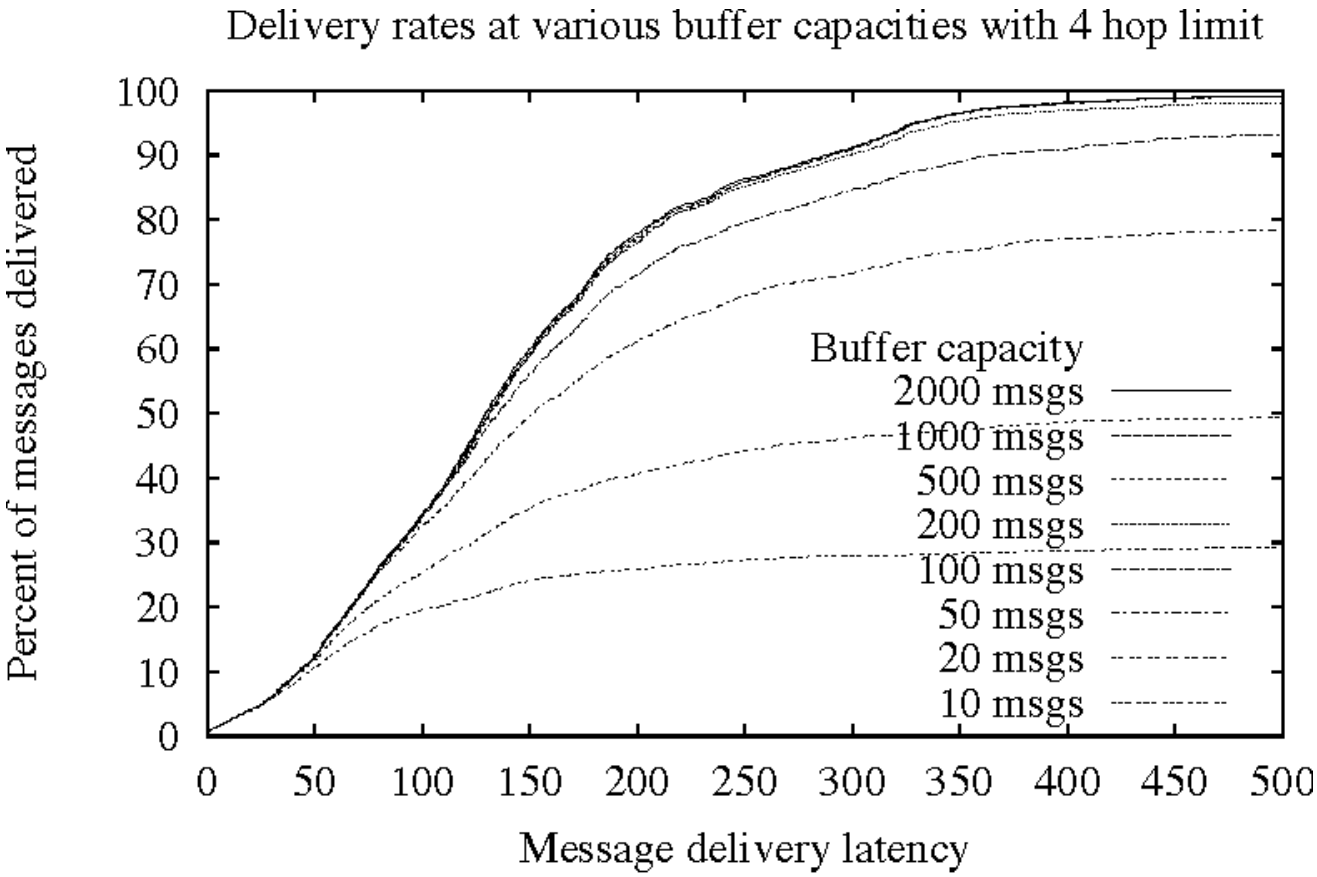


Figura 17[24]: CDF in funzione della percentuale di messaggi consegnati per dimensione del buffer, con raggio di trasmissione di 50 metri.

3.2.2 Spray and Wait

Spray and Wait [26] è un algoritmo di routing pensato per reti mobili wireless con caratteristiche ICMN (Intermittently connected mobile networks) dove i convenzionali algoritmi di routing ad-hoc, sia reattivi che proattivi, falliscono nello scoprire un percorso completo tra i nodi o nel raggiungerli (algoritmi come AODV o per estensione anche AntHocNet, probabilmente).

Spray and Wait rispetto ad algoritmi di "flooding" (Epidemic Routing) performa un numero minore di connessioni ed è meno oneroso in termini di risorse impiegate nella rete, e non dipende strettamente dalla dimensione dei buffer disponibili in ogni nodo, risultando più scalabile e performante con minori ritardi di trasmissione.

l'instradamento in Spray and Wait consiste in 2 fasi:

- Spray phase: per ogni messaggio originato in un nodo fonte, vengono diffuse L copie a L nodi differenti. Il miglior modo per diffondere le copie è il seguente: il nodo fonte inizia con L copie, e quando un nodo A con n>1 copie incontra un nodo B senza copie del messaggio, gli passa $n/2$ copie. Quando un nodo rimane con una sola copia, passa alla comunicazione per trasmissione diretta.
- Wait phase: se la destinazione non è trovata durante la Spray phase, ogni nodo che trasporta una copia passerà alla comunicazione diretta ("binaria"), ossia invierà il messaggio solo alla sua destinazione.

Nella fase iniziale si comporta in maniera simile ad Epidemic Routing, diffondendo messaggi nella rete, rendendo altamente probabile trovare la destinazione, mentre nella seconda fase si limita a inviare direttamente il messaggio. In altre parole può essere visto come un compromesso tra schemi a singola copia e schemi a multi-copia.

Se M nodi con raggio di trasmissione K, si muovono in maniera casuale in un'area N, il delay stimato risulterà [27]:

$$ED_{dt} = 0,5N\left(0,34logN - \frac{2^{K+1}-K-2}{2^K-1}\right)$$

per la trasmissione diretta, e:

$$ED_{sw} \leq \left(H_{M-1} - H_{M-L}\right)ED_{dt} + \frac{M-L}{M-1}\frac{ED_{dt}}{L}$$

per il delay totale. Dove H_n è l'n^{th} numero della serie armonica $H_n = \sum_{i=1}^{n}\frac{1}{i} = \Theta\left(log(n)\right)$.

Per ottenere questi valori di delay è necessario definire un numero opportuno di L copie, e per fare ciò possiamo ipotizzare di avere un valore massimo di delay da rispettare, che possiamo rappresentare essere a volte la funzione di delay ottimale $ED_{opt}\left(a > 1\right)$ con:

$$ED_{opt} = \frac{H_{M-1}}{M-1}ED_{dt}$$

Da ciò otteniamo che per avere un numero L_{min} di copie dobbiamo risolvere $ED_{sw} = aED_{opt}$, che approssimiamo al termine del secondo ordine della serie di Taylor:

$$\left(H_M^3 - 1,2\right)L^3 + \left(H_M^2 - \frac{\pi^2}{6}\right)L^2 + \left(a + \frac{2M-1}{M\left(M-1\right)}\right)L = \frac{M}{M-1}$$

Table 1: minimum L to achieve expected delay

a	1.5	2	3	4	5	6	7	8	9	10
exact	21	13	8	6	5	4	3	3	3	2
bound	N.A.	N.A.	11	7	6	5	4	3	3	2
taylor	N.A.	N.A.	10	7	5	4	3	3	3	2

Figura 18[26]: Valori minimi di L per ottenere il delay aspettato.

Risultati simulazione [26]:

Diversi protocolli di routing verrano testati con un simulatore ad eventi discreto: Epidemic Routing ("epidemic"), Randomized flooding con p = (0.02 − 0.1) (“random-flood”), Utility-based routing con un utility threshold U_{th} = (0.005 − 0.2) (“utility-flood”), Optimal (binary) Spray and Wait con L copie (“spray&wait(L)”), Seek and Focus single-copy routing (“seek&focus”), Oracle-based Optimal routing (“optimal”).

- Scenario **A**:

100 nodi si muovono casualmente in base al modello RWP (random waypoint model) in una griglia 500*500 con barriere riflettenti, ogni nodo ha un raggio di trasmissione K uguale a 10, ed ogni nodo genera un nuovo messaggio per una destinazione casuale con distribuzione di tempo uniforme tra $[1, T_{max}]$ fino a 10000. T_{max} verrà variato da 10000 a 2000 e verranno create situazioni di traffico basso (200 messaggi inviati) e alto (1000 messaggi inviati).

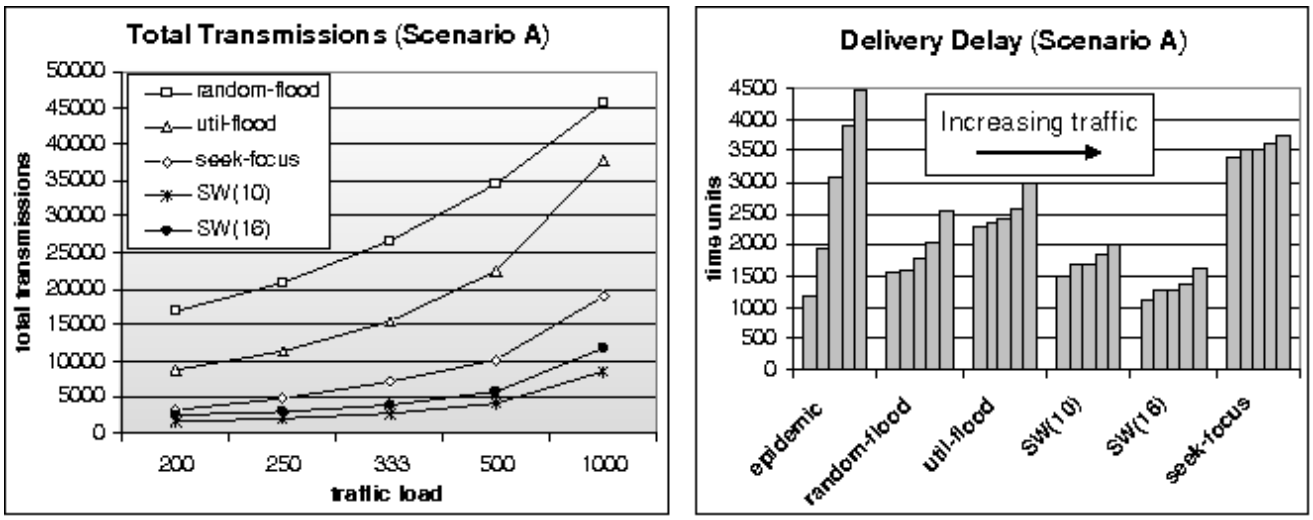


Figura 19[26]: Scenario A: performance dei vari algoritmi sotto diversi carichi di traffico.

- Scenario **B**:

Griglia di 200*200 con T_{max} di 4000 e traffico medio. I numeri di nodi M e il raggio di trasmissione K verranno variati e verranno cambiati le caratteristiche di connettività della rete, passando da molto sparse, molto disconnesse, fino a reti quasi connesse.

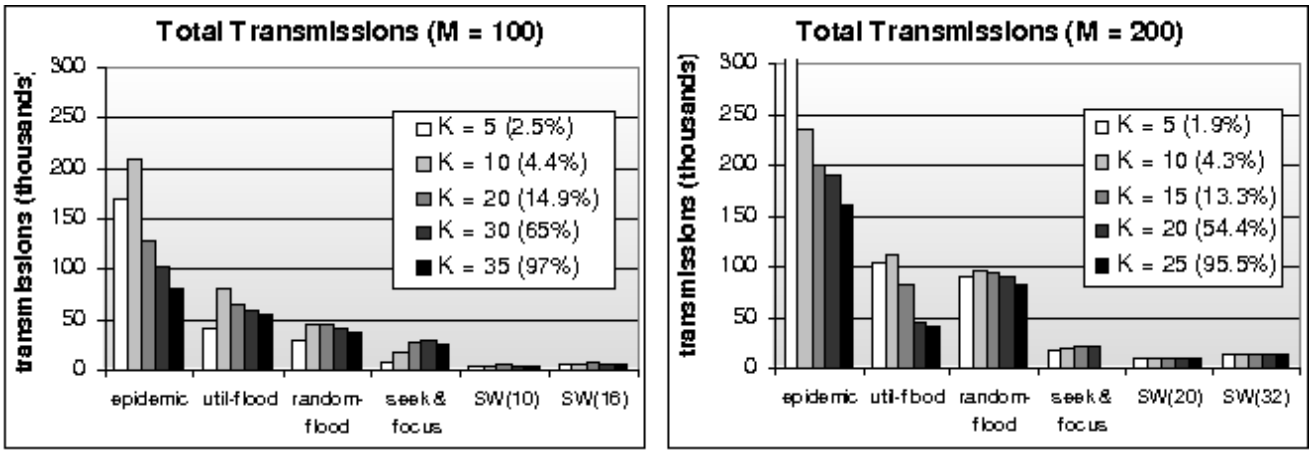


Figura 20[26]: Scenario B: trasmissioni totali in funzione di M e K.

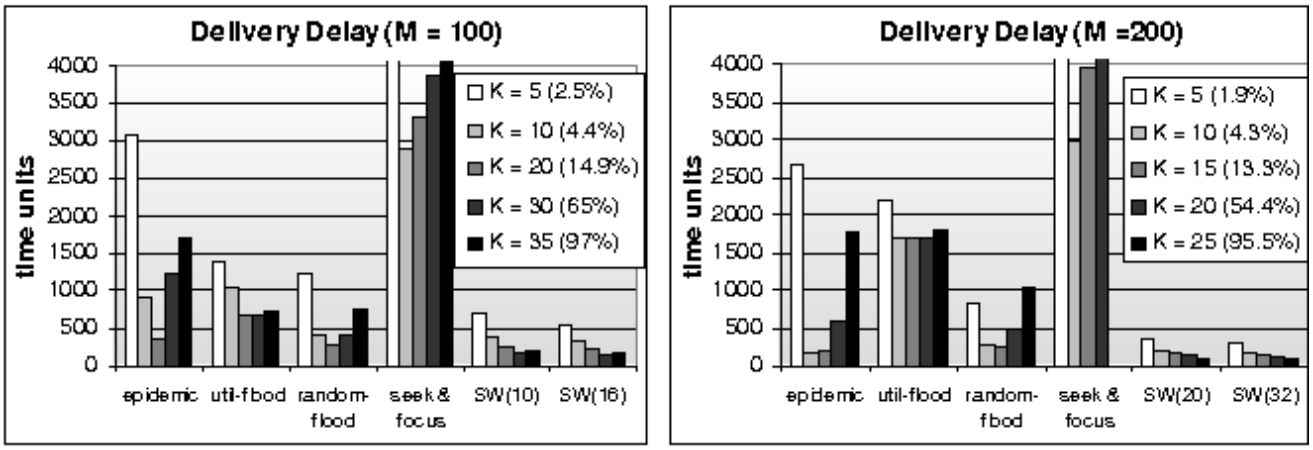


Figura 21[26]: Scenario B: ritardo di consegna in funzione di M e K.

Come si può vedere dai risultati di entrambi gli scenari, Spray and Wait performa in maniera migliore sia in termini di ritardi di consegna con diversi livelli di congestione e connettività, sia in minore consumo di risorse di rete.

3.3 Activator-Inhibitor Systems

Paradigma usato principalmente per coordinazione di sistemi distribuiti e organizzazione di sistemi autonomi, in [11] si può trovare l'implementazione di un sistema "network centric" per reti di sensori (WSN) e reti di sensori/attuatori (SANET), dove si cerca di ridurre il carico e il delay incorso da questo reti dovuto al loop di controllo centrale impiegato: misurazioni dei sensori, trasmissione ad una stazione di controllo, analisi esterna e trasmissione degli ordini agli attuatori.

Per ridurre il consumo di risorse, il loop di controllo avviene in un ridotto numero di sensori, che svolgono le operazioni in maniera "network centric", ossia le operazioni di controllo sono locali e basate su una macchina a stati e un sistema di regole, che specifica le reazioni in base ai dati ricevuti.

Ispirato allo scambio di informazioni tra cellule, presenta caratteristiche che possono essere inerenti ai requisiti Mok:

- Operazioni self-organized senza controllo centrale
- Capacità di self-learning
- Funzionamento in network con nodi mobili e di grandi dimensioni

Ogni informazione scambiata all'interno del modello è basata su diffusione diretta, ed ogni informazione sulla topologia e sugli indirizzi è sostituita da operazioni data-centriche. I messaggi vengono inviati da un sensore a quelli vicini.

Ogni messaggio è codificato come segue:

$$M := type, region, confidence, content$$

Esempio: **{temperatureC, [10,20], 0.6, 20}** :: Temperatura di 20C misurata alle coordinate [10,20]. La confidenza è 0.6, quindi un sensore di bassa qualità è stato usato.

Esempio2: **{pictureJPG, [10,30], 0.9, "binary JPEG"}** :: Immagine scattata in formato JPEG alle coordinate [10,30].

Con questa descrizione è possibile indicare sia i dati (misurati) che le informazioni sull'attuatore da utilizzare ("type" e content). "Region" indica se il messaggio viene da un nodo vicino, mentre "confidence" è usato per indicare la priorità del messaggio.

Per valutare i messaggi ricevuti viene usato un sistema di regole, che specificano la risposta cellulare dell'attuatore. Ogni regola è costituita da 2 parti: un numero di valori di input e un certo output, $INPUT \rightarrow OUTPUT$.

Il set di regole è come segue:

- $A \rightarrow B$ :: messaggio A è convertito in messaggio B
- $C \rightarrow \{\}$ :: il messaggio C è scartato
- $A \wedge B \rightarrow C$ :: se vengono ricevuti sia A che B, viene creato il messaggio C

Esempio: **A(content> 10)** → **A(confidence:= 0.9)** :: se viene misurato un valore più grande di 10, una copia di A viene creata con confidence a 0,9.

Esempio2: **A(content= x)** ∧ **A(content= y)** → **A(content:= x + y)** :: 2 messaggi di tipo A vengono aggregati in un messaggio che ne contiene la somma dei valori.

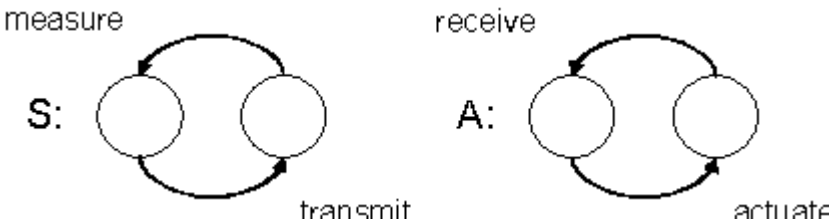


Figura 22[11]: Macchina a stati che mostra i comportamenti dei sensori (S) e attuatori (A).

Usando solo queste informazioni e regole, e usando il metodo di aggregazione, è possibile trovare un attuatore adatto, in base al sistema di regole, e far giungere le informazioni ricevute da più fonti in un singolo messaggio.

In conclusione, il sistema sviluppato risulta più reattivo in quanto il tempo tra misurazione e attuazione non passa per un sistema di controllo centralizzato, riducendo quindi anche i problemi di traffico incorso nella rete, tuttavia la mancanza di informazioni globali implica che potrebbe non essere possibile trovare una soluzione ottimale a livello di rete, e che il set di regole debba essere implementato in ogni nodo. Una possibile soluzione potrebbe essere quella di implementare un set di regole variabile in base alle situazioni locali dei nodi (una sorta di reinforced learning), e propogare le regole aggiornate dove necessario.

4. Analisi critica letteratura

Ogni approccio analizzato nella sezione 3 presenta possibili usi all'interno del modello MoK. Nella sezione 3.1 vengono analizzati i paradigmi di **Swarm Intelligence**, le cui applicazioni principali nei sistemi odierni risultano: routing, ricerca e ottimizzazioni in reti distribuite e allocazione di risorse [1]. **ACO** in particolare è stato preso come framework di riferimento per la sua proprietà di stigmergia, astrazione simile agli **Atomi** di MoK che evaporano col tempo quando non vengono acceduti, o viceversa si rinforzano. Gli agenti di ACO (formiche) visto in 3.1.1, e in particolare le **forward ant** usate in **Antnet** in 3.1.1.2 e **AntHocNet** in 3.1.1.3, si comportano in maniera simile agli **Enzimi** di MoK, dove gli **Enzimi** rappresentano le azioni dei consumatori, e influenzano in maniera proattiva l'evoluzione dei nodi/Atomi, ricercando e attirando informazioni verso il **Compartimento** del consumatore (**Catalizzatore**) nel caso degli **Enzimi**, o cercando un percorso nuovo verso altri nodi e raccogliendo informazioni per rinforzare i percorsi migliori nel caso delle **forward ant**.

Oltre ai possibili usi dati dalle somiglianze semantiche dei 2 modelli, gli algoritmi ACO hanno dimostrato buoni risultati nelle sperimentazioni in ambito di routing, comparati ad algoritmi allo stato dell'arte, AntHocNet specialmente si è dimostrato performante in ambienti distribuiti con reti instabili e mobili, ambiente in cui un sistema Mok si trova ad operare per ricercare e diffondere informazioni.

In tutti i casi però, gli algoritmi di ACO hanno presentato problemi di overhead (a parte Ant System, che però viene presentato come caso di studio per presentare le caratteristiche di ACO e non risulta abbastanza performante), dovuti al carico aggiuntivo sulla rete per via del comportamento proattivo dei suoi agenti, specialmente quando la ricerca dei nodi compiuta dalle formiche avviene in modalità broadcast (AntHocNet), usando una strategia di tipo "flooding". Benchè l'uso del flooding sia solo una parte della strategia di ricerca impiegata da AntHocNet, se questa strategia fosse ritenuta imperante in 3.2 vengono mostrati esempi del paradigma Epidemic Spreading che potrebbero risultare migliori in tale contesto, soprattutto il protocollo Spray and Wait in 3.2.2. Simulazioni di confronto dovrebbero essere effettuate per avallare l'ipotesi.

Gli algoritmi di **Epidemic Spreading** in 3.2 sono di nostro interesse per il loro uso nel diffondere informazioni/messaggi, attraverso strategie di "flooding". Un possibile interessante uso del paradigma non proposto nella letteratura [23, 24, 26] potrebbe essere quello del ricercare **Atomi** simili attraverso la diffusione epidemica: se un'operazione di **Aggregazione** tra 2 Atomi richieda che gli Atomi siano semanticamente correlati, potrebbe essere possibile ricercare tale correlazione facendo un flooding di "messaggi" in un **Compartimento** o Compartimenti vicini usando un identificatore semantico nel messaggio invece dell'identificatore della destinazione, come visto in **Epidemic Routing** in 3.2.1 in cui viene usato allo scopo un summary vector. Resta però da verificare se lo spazio virtuale e l'overhead creato dal flooding di messaggi possa risultare la strategia migliore.

Per ridurre l'overhead può essere implementata una strategia di flooding "binaria" come visto in **Spray and Wait** in 3.2.2, dove viene ridotto il numero di connessioni e risolto il problema di dimensionamento del buffer. Le simulazioni degli algoritmi di Epidemic Spreading mostrano buone prestazioni con reti sparse e solo parzialmente connesse (dove algoritmi come AntHocNet risultano inferiori, vedere diagrammi con nodi fermi rispetto a nodi mobili) dove un nodo destinazione viene eventualmente raggiunto da un messaggio con buona probabilità nella maggior parte dei casi (vedere Figure 16 e 17 in 3.2.1), e potrebbe essere impiegato nei casi in cui in un sistema MoK non sia possibile assicurare "situatedness", ossia la disponibilità spaziale e temporale delle informazioni ricercate (**Fonti, Atomi o Molecole**).

In 3.3 viene analizzato **Activator-Inhibitor Systems** che, a differenza di Swarm Intelligence e Epidemic Spreading che trattano di routing e ricercare/distribuire informazioni, è usato come paradigma di organizzazione e coordinazione di sistemi autonomi e distribuiti. L'implementazione analizzata in 3.3.1 è progettata per una rete mobile e decentralizzata simile ai requisiti richiesti da MoK (caratteristiche presentate anche dagli algoritmi di 3.1 e 3.2). Il punto di interesse è l'analogia riscontrabile tra il sistema delle regole implementato dagli attuatori per l'aggregazione di messaggi e le **Reazioni** di **Aggregazione** impiegate dal modello Mok. In 3.3 si ha un sistema di sensori che raccoglie informazioni e le decodifica in un singolo messaggio (possiamo immaginare la decodifica come una possibile rappresentazione di un **Atomo/Molecola** in MoK) e questi messaggi vengono inviati ad un possibile attuatore che valuta e in caso aggrega tali messaggi basandosi su un sistema di regole (operazione di **Aggregazione** per unire 2 Atomi in una **Molecola**). Purtroppo, oltre a fornire una possibile idea d'implementazione non sono state eseguite simulazioni del sistema per poterne valutare effettivamente l'efficacia e le performance.

Riassumendo:

	Swarm Intelligence	Epidemic Spreading	Activator-Inhibitor Systems
Networking su larga scala	Degli agenti (formiche) vengono creati dai nodi ed esplorano proattivamente la rete a intervalli regolari o quando risulta necessario creare una connessione. Gli agenti comunicano indirettamente tra loro tramite stigmergia, creando possibili soluzioni al problema. Eventualmente, la traccia lasciata dagli agenti rinforza le soluzioni migliori e limita quelle peggiori, determinando così la miglior soluzione globale.	La diffusione delle informazioni dipende dal modello epidemico usato: si possono diffondere molto velocemente informazioni in una rete densa e numerosa se non si pongono limiti all'infettività dei nodi (con opportune politiche di buffer) o ridurre la diffusione e renderla "binaria" per ottenere migliori prestazioni.	Gli attuatori elaborano le informazioni di interesse usando un set di regole, scartando quelle che non si vogliono propagare o aggregando più informazioni da fonti diverse in un singolo messaggio. Il loop di controllo viene quindi definito localmente e la rete è in grado di risolvere problemi di aggregazione/decisione senza una visione globale del comportamento del sistema.
Instabilità	La rete viene proattivamente sondata da degli agenti e le informazioni sul suo stato vengono raccolte localmente e diffuse, in modo da rispondere con soluzioni ottimali ai cambiamenti dell'ambiente.	I meccanismi di flooding e di infezione garantiscono che un buon numero di nodi conservino le informazioni da distribuire, e che tali informazioni vengano distribuite a nodi lontani grazie alla mobilità dei nodi stessi.	La centralità del sistema è basata sui dati e sulla loro aggregazione. I dati non contengono informazioni su indirizzi e topologia, dunque è solo necessaria una sufficiente ridondanza dei dati (come in una WSN).

	Swarm Intelligence	Epidemic Spreading	Activator-Inhibitor Systems
Struttura decentralizzata	Le decisioni di networking vengono prese dai nodi basandosi sull'aggregazione di informazioni locali, raccolte dagli agenti basandosi sul meccanismo di stigmergia.	I meccanismi di flooding e infezioni sono implementati nei nodi stessi, e la diffusione delle informazioni avviene partendo da un nodo, che poi le diffonde infettando altri nodi e il processo continua in maniera potenzialmente esponenziale.	Il sistema è basato sull'accoppiata attivatori-inibitori (sensori-attuatori) dove i sensori raccolgono informazioni che verranno inviate ai nodi vicini, mentre gli attuatori processano le informazioni in base ad un set di regole locale, senza l'ausilio di un'unità centrale.
Necessità di soluzioni stocastiche	La scelta del percorso/nodo per instradare informazioni avviene in maniera probabilistica, influenzata dalla desiderabilità (definita da parametri) del percorso/nodo e da meccanismi di rinforzo.	La destinazione viene raggiunta dalle informazioni attraverso meccanismi di flooding, con probabilità fino al 100% in presenza di parametri ottimali.	Possibilità di impiegare un set di regole variabile in base alla situazione locale dei nodi.

5. Conclusioni

In questa tesi viene presentata un'analisi della letteratura riguardante bio-inspired networking, correlata al modello MoK. Viene descritto il modello MoK, il suo scopo ed elementi costituenti, e vengono analizzate le problematiche riscontrate nel suo ambito di utilizzo, e le possibili caratteristiche ricercate.

In seguito vengono isolati alcuni paradigmi bio-inspired inerenti alle caratteristiche ricercate, e viene analizzata la letteratura riguardante tali paradigmi in cerca di algoritmi, processi e approcci che possano essere sperimentati nel contesto MoK.

I paradigmi isolati risultano 3, Swarm Intelligence, Epidemic Spreading e Activator-Inhibitor Systems. Ognuno di questi paradigmi presenta caratteristiche sovrapposte inerenti ai requisiti, ma risultano alcune caratteristiche peculiari, in base all'ambiente specifico di utilizzo e allo scopo del paradigma, che li rendono utilizzabili in MoK per elaborare/avanzare le varie astrazioni che compongono il modello, o possono fornire una possibile ispirazione implementativa.

Le proprietà e i risultati più interessanti e inerenti sono state ritrovate nel paradigma di Swarm Intelligence, in particolare nel framework di ACO, che racchiude in sè numerose altre implementazioni oltre agli algoritmi mostrati presentati, che si possono dimostrare migliori in altri approcci.

In questa tesi sono stati analizzati solo una parte dei paradigmi che possono essere inerenti a Mok, altri paradigmi come gli algoritmi genetici e le reti neurali presentano interessanti proprietà che possono essere sfruttate nel modello Mok. La letteratura offre numerosi spunti bio-inspired e sempre più ne vengono studiati e approfonditi dati i promettenti risultati e qualità riscontrati in numerosi campi di ricerca, e si ritiene quindi che col tempo altri possibili paradigmi bio-inspired, con caratteristiche inerenti a Mok, possano essere sviluppati.

Riferimenti

[1] Dressler Falko and Özgür B. Akan. “A survey on bio-inspired networking.” Computer Networks 54 (2010): 881-900.

[2] http://apice.unibo.it/xwiki/bin/view/MoK/WebHome

[3] Mariani Stefano and Andrea Omicini. “Molecules of Knowledge: Self-organisation in Knowledge-Intensive Environments.” IDC (2012).

[4] I. F. Akyildiz, D. Pompili, T. Melodia, Underwater acoustic sensor networks: research challenges, Elsevier Ad Hoc Net- works 3 (3) (2005) 257–279.

[5] I. F. Akyildiz, I. H. Kasimoglu, Wireless Sensor and Actor Networks: Research Challenges, Elsevier Ad Hoc Networks 2 (2004) 351–367.

[6] I. F. Akyildiz, W.-Y. Lee, M. C. Vuran, S. Mohanty, NeXt gen- eration/dynamic spectrum access/cognitive radio wireless net- works: a survey, Elsevier Computer Networks 50 (13) (2006) 2127–2159.

[7] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, E. Cayirci, Wireless sensor networks: a survey, Elsevier Computer Net- works 38 (2002) 393–422.

[8] I. F. Akyildiz, X. Wang, W. Wang, Wireless mesh networks: a survey, Elsevier Computer Networks 47 (4) (2005) 445–487.

[9] Chen S., Liu Y., & Mao S. (2014). Big Data: A Survey. MONET, 19, 171-209.

[10] Caro G.A., Ducatelle F., Gambardella L.M. (2005). AntHocNet: an adaptive nature-inspired algorithm for routing in mobile ad hoc networks. European Transactions on Telecommunications, 16, 443-455.

[11] Dressler Falko. “Bio-inspired Network-Centric Operation and Control for Sensor/Actuator Networks.” Trans. Computational Systems Biology 8 (2007): 1-13.

[12] Bharghavan V., Ghosekar P., Ghorpade P., Johnson D.B., Katkar G., Maltz D.A., Sinha P., Sivakumar R. (2010). Mobile Ad Hoc Networking : Imperatives and Challenges.

[13] Dressler F., Long K., Wang J., Zhang Z. (2014). On Swarm Intelligence Inspired Self-Organized Networking: Its Bionic Mechanisms, Designing Principles and Optimization Approaches. IEEE Communications Surveys and Tutorials, 16, 513-537.

[14] Webb B. (1999). Swarm Intelligence: From Natural to Artificial Systems. IEEE Trans. Evolutionary Computation, 4, 192-193.

[15] Krings A.W., Ma Z. (2009). Insect sensory systems inspired computing and communications. Ad Hoc Networks, 7, 742-755.

[16] Caro G.A., Dorigo M., Gambardella L.M. (1999). Ant Algorithms for Discrete Optimization. *Artificial life, 5 2*, 137-72.

[17] Colorni A., Dorigo M., Maniezzo V. (1996). The Ant System: Optimization by a colony of cooperating agents.

[18] http://www.giannidicaro.com/antnet.html

[19] Caro G.A., Dorigo M. (1998). AntNet: Distributed Stigmergetic Control for Communications Networks. *J. Artif. Intell. Res., 9*, 317-365.

[20] Belding-Royer E.M., Lee S., Perkins C.E. (2003). Scalability study of the ad hoc on-demand distance vector routing protocol. *Int. Journal of Network Management, 13*, 97-114.

[21] Omicini A., Viroli M. (2011). Coordination models and languages: From parallel computing to self-organisation. *The Knowledge Engineering Review, 26*(1), 53-59. doi:10.1017/S026988891000041X

[22] Chen C., Zhang C. (2014). Data-intensive applications, challenges, techniques and technologies: A survey on Big Data. *Inf. Sci., 275*, 314-347.

[23] Kurose J.F., Neglia G., Towsley D.F., Zhang X. (2006). Performance Modeling of Epidemic Routing. *Networking*.

[24] Becker D., Vahdat A. (2000). Epidemic Routing for Partially-Connected Ad Hoc Networks.

[25] Josh Broch, David A. Maltz, David B. Johnson, Yih-Chun Hu, and Jorjeta Jetcheva. A Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols. In Proceedings of the Fourth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom), October 1998.

[26] Psounis K., Raghavendra C.S., Spyropoulos T. (2005). Spray and wait: an efficient routing scheme for intermittently connected mobile networks. *WDTN '05*.

[27] T. Spyropoulos, K. Psounis, and C. S. Raghavendra. Single-copy routing in intermittently connected mobile networks. In Proc. of IEEE Secon'04, 2004.

[28] Johnson DB, Maltz DA. Mobile Computing, Chapter Dynamic Source Routing in Ad Hoc Wireless Networks. Kluwer: Norwell, MA, 1996; 153–181.

[29] Camp T, Boleng J, Davies V. A survey of mobility models for ad hoc network research. Wireless Communications & Mobile Computing; special issue on mobile ad hoc networking: research, trends and applications 2002; 2(5):483–502.

[30] http://apice.unibo.it/xwiki/bin/view/MoK/Overview