

Relazione sul Progetto di Sistemi di Simulazione: Analisi della stabilità del protocollo di DHT Symphony

MATTEO BRUCATO e MIRO MANNINO, Università di Bologna

Symphony è un protocollo di overlay per sistemi peer-to-peer in grado di mantenere una tabella hash distribuita (DHT) attraverso nodi che risiedono in una rete geografica.

1. INTRODUZIONE

Symphony [?] è un protocollo per tabelle hash distribuite (DHT) che, attraverso una rete di overlay costruita in base a particolari distribuzioni armoniche, è in grado di garantire lookup efficienti, poli-logaritmici nel numero di hop (salti), ovvero nella sua *latenza*¹.

Attraverso un concetto denominato *greedy routing*, è stato mostrato come sia possibile indirizzare e consegnare un messaggio ad un qualsiasi nodo in una rete in al più $O(\log^2 n)$ salti (un fenomeno chiamato anche *Small World*) [?]. Symphony si basa esattamente su questi concetti, i quali vengono applicati magistralmente al caso di reti peer-to-peer e al problema di creare DHT. In particolare, Symphony è in grado di garantire una latenza di $O(\frac{1}{k} \log^2 n)$ salti, dove $k = O(1)$ è il numero di link che ogni nodo mantiene verso altri nodi della rete.

Nell'articolo in cui Symphony è stato presentato [?], vengono evidenziati i molteplici vantaggi del protocollo rispetto ai suoi predecessori (Chord, Viceroy, ecc.). Il vantaggio più evidente è quello di richiedere un numero costante $k = O(1)$ di link verso altri nodi della rete, a differenza di protocolli come Chord [?] che richiedono un numero logaritmico di link uscenti.

Symphony, inoltre, è dimostrato essere *scalabile* nella dimensione della rete,

Infine, gli autori dichiarano anche il sistema *stabile*, ovvero capace di funzionare nel caso in cui gli host si uniscano alla rete ed escano dalla rete in maniera del tutto arbitraria, tipicamente con brevi periodi di vita. Ciononostante, un'approfondita analisi dell'articolo mostra come gli esperimenti effettuati per stabilire la stabilità di Symphony offrano solamente una panoramica limitata sui possibili scenari che possano veramente essere fonte di instabilità. I test effettuati non sono tali da stressare il sistema fino ai punti più critici, in cui un'evidente instabilità possa essere misurata.

Lo scopo del presente elaborato è quello di studiare a fondo la stabilità del protocollo Symphony, e di stabilire in maniera più accurata se tale protocollo può essere veramente considerato stabile e in che misura. Più precisamente, ci poniamo l'obiettivo di valutare la stabilità del protocollo sotto carichi di stress causati da *churn*, ovvero da alte frequenze di entrata e uscita dei nodi della rete. Studieremo sotto quali condizioni di stress la stabilità del protocollo vale e in quali condizioni essa comincia a vacillare.

Attraverso una serie di esperimenti in simulazione, mostreremo come il protocollo Symphony risponda molto bene a situazioni di stress causato da *churn* simili a quelle testate dagli autori del protocollo. E' solo con l'ausilio di un *motore di churn* molto sofisticato che siamo in grado invece di stressare il sistema fino a fargli perdere la sua stabilità. I nostri esperimenti, infatti, mostrano che

¹Col termine latenza intendiamo il numero medio di hop per l'operazione di lookup, così come viene normalmente utilizzato negli articoli della letteratura sui sistemi peer-to-peer.

La relazione è organizzata nel seguente modo. In Sezione 2 descriveremo brevemente il protocollo Symphny. In Sezione 3 descriveremo formalmente il concetto di stabilità che sarà oggetto di valutazione del protocollo in esame. In Sezione 4, presenteremo i dettagli sul simulatore realizzato, il livello di astrazione del simulatore stesso e dettagli sulla sua realizzazione. La Sezione 5 raccoglierà i risultati sperimentali delle simulazioni, mentre la Sezione 6 illustrerà le nostre conclusioni.

2. IL PROTOCOLLO SYMPHONY

3. ANALISI DELLA STABILITÀ

Nell'articolo originale [?], gli autori Manku et al. testano la stabilità del protocollo Symphony in una cosiddetta *rete dinamica*, ovvero una rete in cui i nodi entrano ed escono arbitrariamente. In particolare, gli autori studiano una rete di centomila nodi, ognuno dei quali ha un numero logaritmico di vicini e alterna uno stato di attività ad uno stato di inattività. I tempi di attività e inattività sono tratti da due distribuzioni esponenziali diverse, tali che i nodi restino attivi per poco tempo (in media mezz'ora) e inattivi per il resto della giornata (23 ore e mezza in media). Le distribuzioni esponenziali fanno sì che la stragrande maggioranza dei nodi rispecchino questo comportamento.

Questo setup rispecchia certamente alcuni scenari veritieri di utilizzo dei sistemi DHT. I nodi entrano nella rete una volta al giorno e vi rimangono per pochissimo tempo (forse il tempo necessario per fare una semplice ricerca e scaricare uno o due file). Ciononostante non è certamente atto a coprire la maggioranza dei casi di utilizzo possibili, specialmente quelli in cui la rete venga stressata in maniera molto più pesante. Si pensi ad esempio ad un utilizzo di una DHT da parte di altri sistemi, in sistemi distribuiti integrati fra loro. In tali situazioni, non si può fare nessuna supposizione né sugli orari di utilizzo, né sulla durata d'utilizzo o sulla frequenza di entrata nella rete.

Non solo lo scenario usato nei test dagli autori di Symphony prevede una frequenza di entrata e uscita piuttosto basse, ma gli autori assumono anche una crescita della rete molto controllata. Nei loro test, infatti, essi fanno crescere la rete in maniera lineare durante l'arco di una giornata. Alla fine della prima giornata, tutti i centomila nodi sono entrati nella rete, e vi rimangono durante tutto l'arco della giornata successiva. Durante il terzo giorno, invece, i peer vengono fatti uscire dalla rete a intervalli regolari. Non siamo riusciti ad immaginare, in questo caso, un esempio di utilizzo reale che rispecchi questo particolare test. Ma soprattutto, ancora, questo test non è atto a stressare la rete con un'adeguata frequenza di churn, o in casi di altissima e imprevedibile attività.

Infine, i test riguardanti la stabilità presenti nell'articolo considerano solo reti senza re-linking. Riteniamo molto interessante stabilire cosa accade in una rete sotto churn nel caso in cui il re-linking sia attivo, in quanto esso forza molti peer a distruggere e ricreare tutti i propri long link, i quali sono quelli che contribuiscono maggiormente alle performance del protocollo stesso.

3.1 Modello di stabilità

Il modello di stabilità che vogliamo considerare nel presente elaborato è più complesso e a grana più fine di quello utilizzato nel paper originale. Innanzitutto, osserviamo come il tempo di permanenza di un certo peer nella rete è influente rispetto la misura di stabilità della rete. Sapere che un peer sta nella rete per mezz'ora o per un giorno non fa alcuna differenza. Ciò che può maggiormente influire sulla stabilità è il fatto che una grande quantità di peer chiedano un lookup mentre la rete è in fase di creazione a seguito di *join* (entrate) e *leave* (uscite) dei peer.

L'ipotesi alla base dei nostri test è che durante i join (e/o i leave) l'anello di Symphony possa avere potenziali punti di inefficienza dati dal fatto che i peer che stanno entrando non hanno ancora terminato la fase di linking necessaria alla rete per funzionare nei tempi logaritmici. Aumentare il numero di peer che fanno join o leave per istante di tempo dovrebbe quindi evidenziare tale inefficienza nelle

misurazioni delle latenze dei lookup.

Ciò che quindi sembra poter avere un impatto più preponderante sulla stabilità non è il tempo di permanenza del singolo peer, ma la *frequenza del fenomeno di churn* (sia esso di leave o di join). Il nostro modello, quindi, prevede di potere valutare la stabilità del sistema al variare della frequenza di join e/o di leave nel sistema.

Questo approccio offre anche numerosi vantaggi in termini di flessibilità nei test. [poter concentrarci sulla frequenza di churn ignorando dettagli come tempo di permanenza, orario di entrata e uscita, effettiva grandezza della rete ad ogni istante di tempo permettono di effettuare test di stabilità più generali]

3.2 La misura di stabilità

Una buona rete peer-to-peer senza struttura garantisce in generale che un lookup non richieda un numero di salti maggiore del numero di nodi della rete. Nel caso pessimo, infatti, la rete invia il messaggio di lookup a tutti i peer della rete finché il peer destinatario non viene contattato. L'upper bound del lookup è cioè $O(n)$ per qualunque rete peer-to-peer non banale.

Una rete strutturata come Symphony, in generale, garantisce molto meglio: un upper bound logaritmico nel numero di salti necessari ad un singolo lookup. Ciononostante, se n peer dovessero entrare tutti contemporaneamente, prima che la rete sia riuscita a creare la struttura necessaria a garantire l'upper bound teorico, le prestazioni potrebbero essere molto peggiori del previsto, fino al caso pessimo di lookup in $O(n)$ salti, come nel caso di rete senza struttura.

Bisogna inoltre considerare che l'upper bound teorico non corrisponde per forza al numero di hop necessari, e può essere anche molto più alto (pessimistico) rispetto a ciò che avviene nella realtà. Una buona struttura e un buon protocollo possono, in media, garantire numero di salti molto minori dell'upper bound teorico.

Ciononostante, è plausibile aspettarsi che uno stress maggiore o minore del sistema possa modificare, in media, il numero effettivo di salti necessari al processamento dei lookup. Ciò potrebbe portare le prestazioni del sistema ad essere nella media più vicine all'upper bound teorico. Nel seguito ci riferiamo a questo trend, se presente e misurabile, come una forma di *instabilità* di un protocollo di lookup. Di contro, un sistema è più stabile tanto più le sue prestazioni rimangono eguali indipendentemente dallo stress momentaneo subito dal sistema, e tanto più il numero di salti dell'algoritmo di lookup misurati sperimentalmente si discosti dall'upper bound teorico. Tale stress è caratterizzato, come detto in precedenza, dalla frequenza di join o di leave dei peer della rete.

Introduciamo quindi, ai fini della nostra analisi, la seguente definizione di *stabilità* di un singolo lookup:

Definition 3.1. (STABILITÀ DI UN LOOKUP) Dato un certo protocollo DHT e una misurazione relativa ad un singolo lookup (h, n) dove h è il numero di hop (salti) del lookup e n è il numero di peer presenti nella rete all'istante della terminazione del lookup, la stabilità del lookup è definita come:

$$stability^{h,n} = 1 - \frac{h}{n}$$

L'instabilità di un singolo lookup è data da $\frac{h}{n}$ se h è il numero di hop del lookup e n il numero di nodi al momento presenti nella rete. Infatti, più h si avvicina ad n , più tale numero si avvicina ad 1, più significa che il numero effettivo di hop si avvicina all'upper bound teorico e quindi alla nostra idea intuitiva di instabilità. La stabilità, così come appena definita, quindi è data dall'opposto della instabilità, ovvero $1 - \frac{h}{n}$.

La stabilità di una singola misurazione è zero nel caso in cui $h = n$, ovvero nel caso di massima instabilità. Non può mai essere minore di zero, in quanto $O(n)$ è l'upper bound dei singoli lookup, come

descritto in precedenza, ovvero $h \leq n$. Essa è uguale ad 1 nel caso in cui il numero di hop effettivi sia zero (ad esempio, il file ricercato si trova nel nodo stesso che lo sta cercando). Non può essere maggiore di 1 poiché sia h che n sono sempre numeri positivi. Pertanto, vale il seguente lemma:

LEMMA 3.2. CARATTERIZZAZIONE DELLA STABILITÀ

$$0 \leq \text{stability}^{h,n} \leq 1$$

Dato un insieme di misurazioni su diversi lookup, definiamo la stabilità delle misurazioni come la media delle stabilità misurate:

Definition 3.3. (STABILITÀ DI UN INSIEME DI MISURAZIONI) Dato un certo protocollo DHT e un insieme di r misurazioni della funzionalità di lookup $H = \{h_1, h_2, \dots, h_r\}$, dove h_i è il numero di hop (salti) relative all' i -esimo lookup, e dato un insieme di r misurazioni $N = \{n_1, n_2, \dots, n_r\}$, tale che n_i corrisponde al numero di peer presenti nella rete all' i -esimo lookup, la *stabilità* del protocollo è definita come:

$$\text{stability}(H, N) = 1 - \frac{\sum_i h_i/n_i}{r}$$

Diciamo che un protocollo è stabile se la sua stabilità, così come appena definita, non varia di molto al variare delle condizioni di stress della rete causate dai fenomeni di churn più o meno frequenti. In altre parole, utilizziamo la seguente definizione:

Definition 3.4. (STABILITÀ DEL PROTOCOLLO) Dato un certo protocollo DHT e un insieme di R insiemi di misurazioni $M = \{(H_1, N_1), (H_2, N_2), \dots, (H_R, N_R)\}$ tale che ogni insieme di misurazioni (H_i, N_i) sia stato effettuato ad un certo livello di stress della rete dato da una certa frequenza f_i di churn, il protocollo è detto essere *stabile* se

$$\text{Var}(\text{stability}(H_i, N_i)) \leq \epsilon$$

per un certo $0 \leq \epsilon < 1/4$.

4. IL SIMULATORE E IL LIVELLO DI ASTRAZIONE
5. RISULTATI
6. CONCLUSIONI