Progetto di L.P.R.
Laurea Triennale in Informatica Applicata
Anno Accademico 2007/2008

Mini-Kazaa:
Un Sistema P2P per lo Scambio di File

1 Scopo del Progetto

Il progetto richiede la realizzazione di *Mini-Kazaa*, un sistema *P2P* per lo *scambio di file* su *WAN*, ispirato alla rete *P2P Kazaa*.
Ogni peer partecipante alla rete *Kazaa* condivide un insieme di *files* con gli altri peer connessi alla rete, può ricercare un file all'interno della rete ed eventualmente effettuarne il download.
Kazaa prevede due diversi tipi di peer, i *Super Nodes (SN)* e gli *Ordinary Nodes (ON)*. Ogni *ON* è collegato, in ogni istante, ad un solo *SN* e comunica ad esso i descrittori di tutti i files che intende condividere. Lo scopo principale dei *SN* è quindi quello di *indicizzare* i files offerti dai propri *ON*.
Ogni *SN* stabilisce connessioni, oltre che con un insieme di *ON*, anche con un numero limitato di *SN* vicini. I *SN* ricevono le query dagli *ON* connessi, verificano se le query individuano un files condiviso da un *ON* direttamente connesso ed in caso positivo comunicano l'identità di tale peer. In caso negativo la query viene inoltrata ad un sottoinsieme dei *SN* vicini. La query viene propagata tra i *SN* fino a che il file ricercato non viene individuato. Il trasferimento del file avviene comunque *in modo diretto* tra il peer che ha richiesto il file e quello che lo possiede.
La toplogia dell'overlay network risultante è mostrata nella figura 1

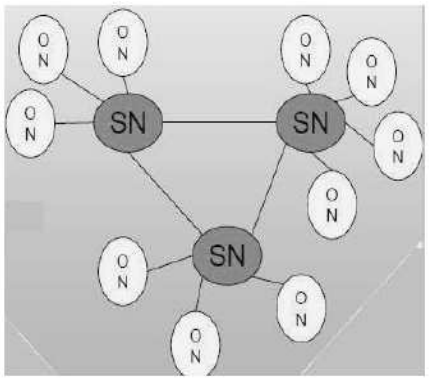

Figure 1: Mini-Kazaa: Topologia

2 Mini-Kazaa: Specifica del Protocollo

Ogni peer *Mini-Kazaa* viene *configurato esplicitamente* dall'utente come *SN* oppure come *ON* al momento della prima connessione alla rete *Kazaa*. Successivamente non è possibile cambiare tale configurazione.

La rete *Mini-Kazaa* prevede un insieme di *Bootstrap Servers* che contengono una lista di indirizzi *IP* di *SN*. Al momento della connessione alla rete *Kazaa* un *SN* o un *ON* contatta uno dei *Bootstrap Server* che gli fornisce un insieme di riferimenti a *SN*. Un *ON* sceglie uno dei *SN* e si connette ad esso. Un *SN* si connette ad un sottoinsieme dei *SN*. I rimanenti riferimenti ricevuti dal bootstrap Server vengono memorizzati in una *cache locale*. Quando la permanenza sulla rete di un *SN* supera un *intervallo di tempo T* prestabilito, *SN* notifica la propria identità al bootstrap server che lo inserisce nella lista dei *SN* che comunica ai peer.

Il Bootstrap Server *notifica periodicamente* ai *SN* la presenza di nuovi *SN*. In questo modo ogni peer può aggiornare periodicamente la lista dei *SN* conosciuti, in modo da stabilire con essi nuove connessioni, nel caso di fallimento/abbandono volontario della rete da parte dei *SN* a cui si è connessi.

La scelta dei *SN* con cui stabilire le connessioni, può essere guidata da diversi fattori. Ad esempio, è possibile considerare il *carico dei SN* o la *prossimità* dei *SN*, misurata in termini di *Round Trip Time, RTT* o in termini di *similarità* tra l'*indirizzo IP* proprio e quello del *SN*.

Una volta stabilita la connessione con un *SN*, l'*ON* comunica i descrittori dei files che intende condividere. Successivamente, ON può comunicare i descrittori di ulteriori files. Inoltre un *ON* riceve dall'utente richieste di ricerca di files e le inoltra al proprio *SN*. *SN* controlla se la query può essere soddisfatta da uno dei propri *ON*. In caso negativo, associa alla query un *Time To Live* il cui valore é definito staticamente e la propaga ad un sottoinsieme di *SN* vicini *scelti in modo casuale*. La propagazione della query *termina* quando il file richiesto viene individuato oppure quando il valore del *TTL* diventa uguale a 0.

Il *SN S* che individua il file ricercato deve notificare l'identità dell'*ON OW* che possiede il file all'*ON OR* che ha richiesto il file. A tale scopo *S* genera un messaggio di risposta contenente indirizzo *IP* e *porta* di *OW*. Il messaggio viene *spedito a ritroso* lungo lo stesso percorso seguito dalla query. Il *SN* collegato a *OR* propaga il messaggio ad *OR* che apre una connessione verso *OW* e scarica il file.

3 Implementazione dal Sistema

La soluzione implementata deve soddisfare i seguenti vincoli:

- utilizzare il *protocollo RMI* e le *callbacks RMI* per l'interazione tra i peer ed il Bootstrap Server. Un peer contatta inizialmente, mediante *RMI*,

un *Bootstrap Server* sia per ottenere liste di *SN* che per registrare una *callback* presso tale Server. La callback viene utilizzata dal *Server* per notificare successivamente l'identità di nuovi *SN* che si collegano alla rete.

- utilizzare il protocollo *TCP* per implementare le connessioni tra i *SN* e tra i *SN* e gli *ON*. Il protocollo *TCP* deve essere utilizzato anche per implementare la connessione diretta tra *ON* tramite cui avviene il download dei files.

- utilizzare il protocollo *UDP* per implementare il meccanismo di *ping-pong* che consente di valutare il *RTT* con i *Super Nodes* appartenenti alla lista restituita dal *Bootstrap Server*.

Le seguenti funzionalità sono *opzionali*. L'implementazione di tali funzionalità inciderà ovviamente sulla valutazione finale del progetto.

- realizzare un supporto per query complesse. A questo scopo è possibile utilizzare un package JAVA che consente di manipolare espressioni regolari. La documentazione può essere reperita alla seguente URL:

  http://java.sun.com/docs/books/tutorial/essential/regex/index.html

- in *Kazaa* i collegamenti tra *SN* e *SN* ed *ON* sono altamente *dinamici*. Ogni *SN*, rs. *ON* cambia le sue connessioni con i *SN* vicini, rs. con il *SN* molto frequantemente. Questo consente l'esplorazione di una porzione più ampia della rete. Implementare questa funzionalità in *Mini Kazaa*.

Occorre inoltre prevedere un insieme di meccanismi che consentano di monitorare in modo diretto il funzionamento dell'host *Mini-Kazaa*. Ad esempio, definire una semplice interfaccia che consenta all'utente di monitorare le connessioni stabilite da ogni peer, di visualizzare lo stato della cache del peer, lo stato di download dei file, una maschera per effettuare le query, etc.

4 Modalità di svolgimento del Progetto

Il progetto può essere svolto in gruppo. Ogni gruppo deve essere composto al massimo da *due studenti*. Il materiale consegnato deve comprendere:

- La stampa di tutto il codice dello strumento e di eventuale programmi utilizzati per il test delle funzionalità dello strumento.

- una stampa della relazione *in formato pdf* che descriva tutte le scelte effettuate. La relazione deve contenere

  - una descrizione generale della architettura del sistema e della scelte di progetto effettuate.
  - una descrizione delle strutture dati utilizzate

- uno schema generale dei threads attivati da ogni *peer* e delle modalità di interazione di tali threads
- un manuale d'uso che indichi chiaramente le modalità di interazione con il peer *Mini-Kazaa*.

L'organizzazione e la chiarezza dell'esposizione della relazione influiranno sul voto finale dell'esame. L'utilizzo di metodologie di documentazione del software quali diagrammi UML (delle classi, di sequenza,...) influirà positivamente sulla valutazionde del progetto.

Relazione e codice devono essere consegnati in formato elettronico, via e-mail.

Il progetto deve essere consegnato una settimana prima della data dell'orale. L'orale verterà sia sulla discussione del progetto che sul programma svolto durante il corso. La prova del corretto funzionamento del programma verrà effettuta utilizzando la rete locale del laboratorio del Polo Didattico Marconi.