Algoritmo Bully con Blocchi di Richieste

Relazione del progetto di Sistemi Distribuiti

Casazza Marco
Matricola 771605

Sommario

Molti sistemi distribuiti necessitano della presenza di processi con ruoli speciali che esercitano la funzione di leader della rete. Questi processi vengono chiamati Coordinatori *e la loro scelta non è casuale ma si basa sul risultato di apposite procedure chiamate* procedure di elezione *o* algoritmi di elezione*. L'*algoritmo Bully (AB) *è uno dei più famosi e semplici algoritmi per l'elezione di un coordinatore ma ha lo svantaggio di generare $O(n^2)$ messaggi che circolano nella rete durante la sua esecuzione, dove n è il numero di processi della rete. Questo documento propone una modifica all'algoritmo Bully in grado di ridurre il numero di messaggi inviati durante l'elezione di un nuovo coordinatore a $O(n)$ e mostra, a supporto di questa tesi, una confronto tra l'algoritmo Bully e l'algoritmo modificato chiamato* algoritmo Bully con Blocchi di Richieste (ABBR)*.*

1 Introduzione

L'elezione di un leader in una rete è un problema fondamentale quando si parla di algoritmi distribuiti. In una rete di processi paritari è spesso necessario che uno di questi assuma un ruolo speciale in modo da svolgere determinate operazioni, come ad esempio la sincronizzazione dei processi. Questo particolare processo è detto *coordinatore* della rete.

Purtroppo non è possibile identificare a priori un coordinatore ed è necessario che siano i nodi stessi della rete a decidere chi assumerà questo ruolo comunicando tra di loro e utilizzando un algoritmo comune per la scelta. Questi algoritmi vengono chiamati *algoritmi di elezione* e il loro scopo è far sì che, al termine dell'elezione, tutti i processi siano concordi sulla scelta del coordinatore.

Uno degli algoritmi di elezione più utilizzati per la sua semplicità e robustezza è l'*algoritmo Bully (AB)*[1]. Questo algoritmo ha però lo svantaggio di generare un numero consistente di messaggi per ogni processo che esegue l'elezione. Ne deriva che la rete viene saturata di messaggi necessari esclusivamente alla scelta del nuovo coordinatore.

In questo documento viene proposto un algoritmo basato sull'AB chiamato *algoritmo Bully con Blocchi di Richieste (ABBR)* studiato per ridurre il traffico di messaggi scambiati tra i processi durante l'elezione di un coordinatore nelle rete. Oltre ad una dimostrazione teorica delle proprietà dell'ABBR sono anche proposti i risultati di una campagna di test che mette a confronto il numero di messaggi generati dai due algoritmi.

Nel capitolo 2 viene descritto l'AB proposto originariamente in [1]. Nel capitolo 3 sono descritte brevemente alcune modifiche dell'AB già proposte in letteratura. I dettagli dell'algoritmo ABBR sono descritti nel capitolo 4, mentre i risultati della comparativa tra AB e ABBR sono mostrati nel capitolo 5.

2 Algoritmo Bully

L'idea alla base dell'AB è che i processi competano per assumersi il ruolo di coordinatore quando questo diventa inattivo.

Affinché si possa procedere ad una elezione è necessario però fare delle supposizioni senza le quali non sarebbe possibile eseguire l'AB:

- ogni processo è identificato univocamente nella rete con un proprio *id*;
- ogni processo conosce quali sono gli altri processi della rete e come comunicare con loro;
- l'invio dei messaggi sul canale di comunicazione è garantito ma un processo potrebbe non essere attivo;
- un processo P non conosce a priori se un processo Q è attivo o inattivo;
- ad ogni processo è possibile attribuire una priorità differente da quella degli altri, in genere (ed anche in questo caso) è utilizzato l'id.

Nell'AB un processo che si accorge dell'inattività del coordinatore inizia la propria procedura di elezione alla ricerca di processi attivi con identificatore maggiore. Se non ne trova si autoelegge a coordinatore comunicandolo agli altri processi della rete, altrimenti si metterà in attesa di ricevere il nuovo coordinatore.

I dettagli dell'algoritmo sono riportati nello pseudocodice 1 mentre in figura 1 è mostrato un esempio di elezione.

I principali vantaggi dell'AB sono la semplicità di implementazione e la sua robustezza. Purtroppo però l'AB ha lo svantaggio di usare un numero di messaggi $O(n^2)$ per l'esecuzione della procedura di elezione, dove n è il numero di processi della rete. Questo fa sì che, quando la procedura di elezione è utilizzata da più processi, la rete venga inondata di messaggi di elezione e di risposta che aumentano i tempi di comunicazione tra processi.

3 Letteratura

In letteratura sono state proposte diverse modifiche all'AB per porre rimedio ai suoi problemi. In [3] è stato proposto un algoritmo chiamato *algoritmo Fast Bully (AFB)* che delega la scelta del coordinatore ad un processo della rete: quando il processo P si accorge dell'inattività del coordinatore invia un messaggio di *Elezione* a tutti i processi con identificatore superiore. Se P non riceve risposte l'AFB funziona come l'AB e P si autoelegge inviando un messaggio *Coordinatore* a tutti i processi della rete. Se invece P riceve delle risposte allora sceglie, tra quelli che hanno risposto, il processo Q con l'identificatore più alto. P invia allora a Q un messaggio *Nomina* e se Q accetta invia a tutti un messaggio *Coordinatore*.

Similmente all'AFB anche l'algoritmo proposto in [4] delega l'elezione ad un solo processo che sceglierà chi sarà il nuovo coordinatore. La differenza tra i due algoritmi consiste nel fatto che mentre il primo nomina solamente il nuovo coordinatore, il secondo invia direttamente un messaggio a tutti i processi della rete indicando quale processo è stato scelto come coordinatore.

Anche in [5] è stato proposta una modifica che delega l'elezione del nuovo coordinatore: l'algoritmo definisce un gruppo di processi che costituisce una *commissione di elezione*. Nel caso il coordinatore sia inattivo, il processo che se ne accorge interpella la commissione che innanzitutto verifica l'effettiva inattività del coordinatore e successivamente ricerca nella rete il processo con identificatore maggiore. Quando il nuovo coordinatore è stato trovato viene comunicato a tutti i processi della rete dalla commissione stessa.

Un'ulteriore modifica è stata proposta in [2] e si basa sulla scelta anticipata di un numero limitato di alternative al coordinatore conosciute da tutti i processi della rete. Quando un processo si accorge dell'inattività del coordinatore si rivolge direttamente alla prima alternativa che verificherà l'effettiva inattività del coordinatore e si autoeleggerà comunicandolo a tutti i processi della re-

1. il processo P si accorge che il coordinatore non risponde:
   (a) P invia un messaggio *Elezione (E)* ai processi con identificatore più alto del suo;
   (b) se P non riceve risposte significa che non ci sono altri processi con identificatore più alto del suo e P si autoelegge coordinatore inviando un messaggio *Coordinatore (C)* a tutti gli altri processi della rete. P termina la procedura di elezione;
   (c) se P riceve risposte si mette in attesa di ricevere chi sarà il nuovo coordinatore e termina la sua procedura di elezione.
2. un processo Q riceve un messaggio *Elezione (E)* da un processo P con identificatore più basso:
   (a) Q invia una risposta *Ok (O)* a P;
   (b) Q inizia la sua procedura di elezione.
3. un processo Q riceve un messaggio *Elezione (E)* da un processo P con identificatore più alto:
   (a) Q ammette P come possibile nuovo coordinatore e interrompe la procedura di elezione.

Pseudocodice 1: Algoritmo Bully

te. Nel caso la prima alternativa sia anch'essa inattiva il processo passerà alla seconda e così via. Nel caso in cui tutte le alternative siano inattive si eseguirà una procedura d'elezione simile a quella proposta in [4] che, oltre al coordinatore, sceglierà anche le nuove alternative.

Tutti gli algoritmi sopra descritti riescono a diminuire il numero di messaggi generati durante l'esecuzione della procedura di elezione da $O(n^2)$ a $O(n)$, dove n è il numero di processi nella rete.

3.1 Alternative all'algoritmo Bully

In letteratura non esiste solo l'AB: l'alternativa più nota è l'*algoritmo ad Anello (AA)*, molto utile quando la rete dei processi è organizzata fisicamente o logicamente ad anello.

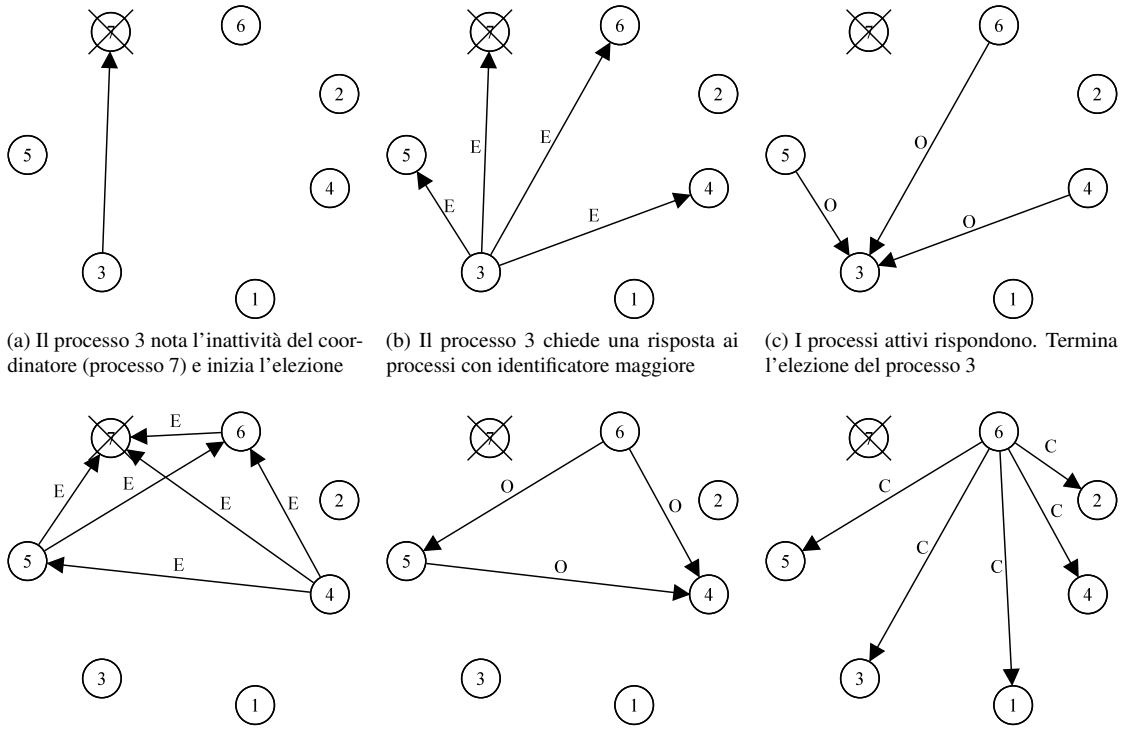
Il processo P che si accorge dell'inattività del coordinatore prepara un messaggio, chiamato *token*, nel quale scrive il proprio identificatore e lo passa al processo successivo nell'anello. Il token viene fatto girare all'interno della rete ed ogni processo aggiunge al token il proprio identificatore. Quando il token ritorna al processo P il nuovo coordinatore viene scelto da P stesso in base alla lista di identificatori presenti nel token. A questo punto P fa circolare nella rete un nuovo messaggio che indica chi è il nuovo coordinatore.

L'AA rispetto all'AB ha il vantaggio di generare $O(n)$ messaggi senza ulteriori modifiche ma soffre di altri problemi come ad esempio la perdita del token nella rete.

4 Algoritmo Bully con blocchi di richieste

Il principio alla base della modifica che propongo è quello di provare a chiedere l'elezione subito a processi con identificatore alto (similmente a quanto fatto in [2] con le alternative) in modo da evitare di dover inviare messaggi anche a processi la cui elezione fallirà per via dell'identificatore troppo basso.

L'*algoritmo con blocchi di richieste (ABBR)* non invia il messaggio *Elezione* a tutti i processi con identificatore superiore ma solo ai primi k processi con identificatore più alto, dove k è un parametro di ingresso. Nel caso nessuno dei processi risponda si procederà con un se-



(a) Il processo 3 nota l'inattività del coordinatore (processo 7) e inizia l'elezione

(b) Il processo 3 chiede una risposta ai processi con identificatore maggiore

(c) I processi attivi rispondono. Termina l'elezione del processo 3

(d) I processi 4, 5 e 6 richiedono la propria elezione

(e) I processi 4 e 5 ricevono risposta e terminano l'elezione, il 6 invece non riceve risposte e può eleggersi coordinatore

(f) Il processo 6 comunica ai processi della rete di essere il nuovo coordinatore

Figura 1: Esempio di elezione con l'algoritmo Bully

condo blocco di richieste e così via fino ad ottenere una risposta o fino ad esaurire i processi con identificatore superiore.

Come si può notare l'ABBR non è molto differente dall'AB, che può essere visto come un caso particolare dell'ABBR in cui $k = n$, dove n è il numero dei processi della rete.

L'algoritmo è descritto in dettaglio nello pseudocodice 2 e un esempio di elezione è mostrato in figura 2.

4.1 Analisi dell'algoritmo

È possibile fare un'analisi del numero di messaggi che vengono inviati nella rete durante l'esecuzione della procedura: sia P il processo che si accorge dell'inattività del coordinatore, n il numero dei processi nella rete, k il numero di messaggi *Elezione* inviati per ogni blocco di richieste e r il numero di blocchi di richieste che il processo effettua. Possiamo suddividere l'analisi in 3 casi separati:

1. P è il processo con identificatore maggiore: P non invia nessun messaggio poiché nella sua lista di processi non ha processi con identificatore più alto. Si autoelegge coordinatore e invia n messaggi *Coordinatore*. In questo caso vengono inviati $O(n)$ messaggi;

2. P è un generico processo ma ogni processo Q con identificatore maggiore non è attivo: P invia rk messaggi per scoprire che non esiste un processo attivo con identificatore più alto e altri n messaggi per comunicare ai processi della rete di essere il

1. il processo P si accorge che il coordinatore non risponde:
   (a) P ordina la lista dei processi conosciuti (compreso se stesso) in ordine decrescente in base all'identificatore;
   (b) P seleziona un blocco di processi composto dai primi k processi della lista e invia un messaggio *Elezione (E)* solo a quelli con identificatore più alto del proprio;
   (c) se P non riceve risposte
      i. se P faceva parte del blocco si autoelegge coordinatore e invia un messaggio *Coordinatore (C)* a tutti i processi. P termina la procedura di elezione;
      ii. se P non faceva parte del blocco esclude i processi contattati dalla lista e riparte dal punto 1b;
   (d) se P riceve risposte si mette in attesa di ricevere chi sarà il nuovo coordinatore e termina la sua procedura di elezione.
2. un processo Q riceve un messaggio *Elezione (E)* da un processo P con identificatore più basso:
   (a) Q invia una risposta *Ok (O)* a P;
   (b) Q inizia la sua procedura di elezione.
3. un processo Q riceve un messaggio *Elezione (E)* da un processo P con identificatore più alto:
   (a) Q ammette P come possibile nuovo coordinatore e interrompe la procedura di elezione.

Pseudocodice 2: Algoritmo Bully con blocchi di richieste

nuovo coordinatore. In totale abbiamo l'invio di $rk+n$ messaggi, ma nel caso peggiore $r = \frac{n}{k}$ quindi $n + n$ messaggi. Anche in questo caso vengono inviati $O(n)$ messaggi;

3. P è un generico processo ed esiste almeno un processo Q con identificatore maggiore attivo: P invia $r-1$ blocchi di richieste senza senza avere risposta. Solo con l'r-esimo blocco di richieste trova, nel caso peggiore, k processi attivi. Uno di questi sarà il processo attivo con identificatore più alto nella rete ricadendo nel caso 2. Gli altri $k-1$ processi del blocco invieranno a loro volta r blocchi di richieste fino ad avere risposta dal loro stesso blocco. Sintetizzando abbiamo quindi che

   - il processo P invia e riceve in totale $rk + k$ messaggi, dove rk è il numero di richieste di *Elezione* e k il numero di messaggi di *Ok* ricevuti;
   - il processo che sarà il nuovo coordinatore invia $2n$ messaggi e non ne riceve, come già descritto nel caso 2;
   - gli altri $k-1$ processi del blocco inviano e ricevono ognuno $rk + k$ messaggi come il processo che ha iniziato l'elezione.

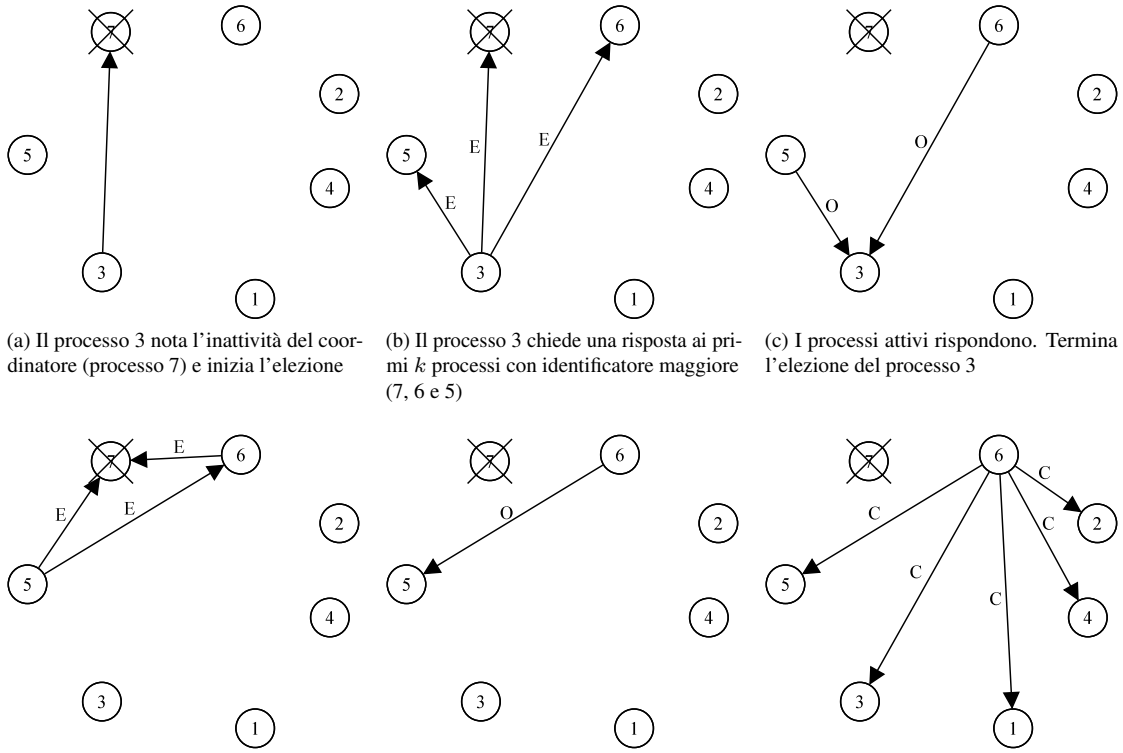
In totale la nostra procedura genera

$$rk + k + (k-1)(rk + k) + 2n$$

messaggi. Dato che nel caso peggiore $r = \frac{n}{k}$ otteniamo

$$n + k + (k-1)(n + k) + 2n$$
$$3n + k + kn + k^2 - n - k$$
$$2n + kn + k^2$$

ed anche in questo caso abbiamo un numero di messaggi lineare rispetto ad n.



(a) Il processo 3 nota l'inattività del coordinatore (processo 7) e inizia l'elezione

(b) Il processo 3 chiede una risposta ai primi k processi con identificatore maggiore (7, 6 e 5)

(c) I processi attivi rispondono. Termina l'elezione del processo 3

(d) I processi 5 e 6 richiedono la propria elezione

(e) Il processo 5 riceve risposta e termina l'elezione, il 6 invece non riceve risposte e può eleggersi coordinatore

(f) Il processo 6 comunica di essere il nuovo coordinatore

Figura 2: Esempio di elezione con l'algoritmo Bully con blocchi di richieste e $k = 3$

In tutti i casi l'algoritmo genera un numero di messaggi che cresce linearmente rispetto al numero di processi presenti nella rete. Possiamo quindi affermare che l'ABBR genera un numero di messaggi $O(n)$ ed è quindi migliore dell'AB che ne genera $O(n^2)$.

5 Analisi sperimentale

A supporto della dimostrazione teorica è stato effettuato un ulteriore confronto tra l'AB e l'ABBR implementando entrambi gli algoritmi in Java e contando i messaggi immessi nella rete dai processi durante l'elezione.

La simulazione ha preso in considerazione diverse configurazioni che divergono per numero di processi e valore del parametro k. Per ogni configurazione sono state effettuate 10 prove.

La procedura con la quale sono stati effettuati i test è la seguente:

1. viene creata la rete con n processi, ognuno con un proprio id;

2. ogni processo ha la probabilità del 20% di terminare prima dell'inizio dell'elezione. Questo passo è stato inserito per simulare casi reali in cui è possibile che vi siano processi inattivi;

3. il coordinatore viene terminato forzatamente ed i processi che se ne accorgono iniziano la propria procedura di elezione.

Nella tabella 1 sono riportate, per ogni configurazione, le medie del numero di messaggi circolati nella rete:

- in colonna 1 è riportato il numero dei processi della rete;
- in colonna 2 sono riportate le medie dei test effettuati con l'AB (utilizzando quindi $k = n$);
- nelle colonne 3, 4 e 5 sono riportate le medie dei test effettuati con l'ABBR rispettivamente con $k = 1$, $k = 2$ e $k = 3$.

Numero processi	**K = N**	**K = 1**	**K = 2**	**K = 3**
10	77	27	32	39
20	298	62	69	86
40	1143	141	129	189
60	2606	204	215	288
80	4477	251	355	362
100	6810	370	382	524

Tabella 1: Messaggi inviati nella rete

Come si può notare il numero di messaggi generati dall'AB è di molto superiore a quelli generati dall'ABBR ed in figura 3 si può notare meglio la differenza di crescita dei due algoritmi. In figura 4 si può invece notare la crescita di messaggi generati dall'ABBR al variare di k.

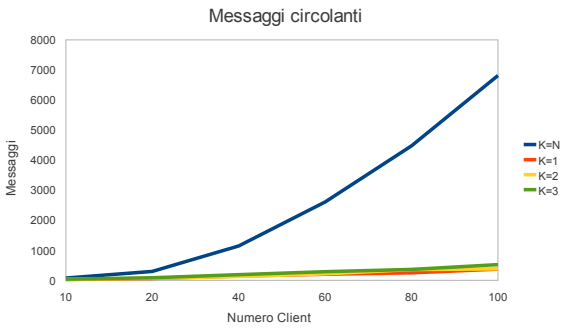


Figura 3: Messaggi circolanti nella rete al variare di k

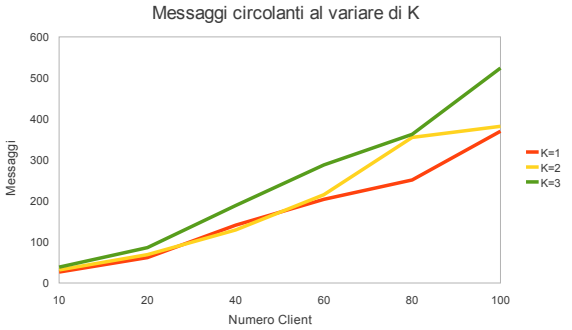


Figura 4: Messaggi circolanti nella rete al variare di k (focus sui valori dell'ABBR)

Oltre al numero di messaggi circolati nella rete è importante valutare il numero r di blocchi di richieste inviate dal processo che ha vinto l'elezione per diversi valori di k. Questo valore influenza il tempo necessario all'elezione di un nuovo coordinatore: più blocchi di richieste sono stati inviati e più alto è stato il tempo dell'elezione. In tabella 2 sono riportate le medie di r e il massimo valore di r tra i test effettuati:

- in colonna 1 è riportato il numero dei processi della rete;
- nelle colonne 2 e 3 sono riportate rispettivamente la media (A) e il numero massimo (M) di r per $k = 1$;
- nelle colonne 4 e 5 sono riportate rispettivamente la media (A) e il numero massimo (M) di r per $k = 2$;
- nelle colonne 6 e 7 sono riportate rispettivamente la media (A) e il numero massimo (M) di r per $k = 3$.

Come si può notare capita spesso con $k = 1$ di dover inviare più blocchi di richieste. Ad esempio con 100 processi si son dovuti inviare fino a 5 blocchi di richieste: sono solo 5 messaggi ma ciò vuol dire che l'algoritmo è rimasto in attesa di risposte per 5 volte e quindi il tempo necessario per l'elezione è stato quintuplicato rispetto al normale.

È quindi necessario calibrare con attenzione il valore di k: se troppo basso infatti costringe l'algoritmo ad

Numero processi	K = 1		K = 2		K = 3	
	A	M	A	M	A	M
10	2	3	1	2	1	1
20	2	3	1	2	1	2
40	2	4	1	1	1	2
60	2	4	1	2	1	2
80	2	3	2	2	1	1
100	2	5	1	2	1	2

Tabella 2: Blocchi di richieste inviati

effettuare più blocchi di richieste rimanendo in attesa e rallentando l'elezione, se troppo alto diminuisce il vantaggio rispetto all'AB e rischia di inviare un numero simile di messaggi.

6 Conclusioni

In questo documento è stato mostrato un nuovo algoritmo chiamato *algoritmo Bully con Blocchi di Richieste (ABBR)*. L'algoritmo consiste in una modifica applicata all'*algoritmo Bully (AB)* che separa il numero di richieste in blocchi così da limitare il numero di messaggi immessi nella rete. La dimostrazione presentata e i risultati dell'analisi sperimentale permettono di affermare che l'ABBR è migliore dell'AB sotto l'aspetto del numero di messaggi immessi nella rete. Un eventuale sviluppo futuro di questo algoritmo potrebbe portare all'integrazione con altri algoritmi già presenti in letteratura, come ad esempio l'*algoritmo Fast Bully*[3].

Riferimenti bibliografici

[1] Hector Garcia-Molina. Elections in a distributed computing system. *IEEE Transactions on Computers*, 31:48–59, 1982.

[2] M. S. Kordafshari, M. Gholipour, M. Mosakhani, A. T. Haghighat, and M. Dehghan. Modified bully election algorithm in distributed systems. In *Proceedings of the 9th WSEAS International Conference on Computers*, pages 10:1–10:6, Stevens Point, Wisconsin, USA, 2005. World Scientific and Engineering Academy and Society (WSEAS).

[3] Seok-Hyoung Lee and Hoon Choi. The fast bully algorithm: For electing a coordinator process in distributed systems. In Ilyoung Chong, editor, *Information Networking: Wireless Communications Technologies and Network Applications*, volume 2344 of *Lecture Notes in Computer Science*, pages 609–622. Springer Berlin / Heidelberg, 2002.

[4] Quazi Ehsanul Kabir Mamun, Salahuddin Mohammad Masum, and Mohammad Abdur Rahim Mustafa. Modified Bully algorithm for electing coordinator in distributed systems. *WEAS transaction in Computers*, 3(4):948–953, October 2004.

[5] Muhammad Mahbubur Rahman and Afroza Nahar. Modified bully algorithm using election commission. *CoRR*, abs/1010.1812, 2010.