Transactions on Information and Communications Technologies vol 15, © 1997 WIT Press, www.witpress.com, ISSN 1743-3517

Complexity analysis of distributed database algorithm

Alfred W.S. Loo[a], Chris Bloor[b] and David Grey[b]
[a]*Department of Computer Studies, Lingnan College, Tuen Mun, Hong Kong*
[b]*School of Computing and Information systems, University of Sunderland, St Peters Way, Sunderland ST6 0DD, United Kingdom*

Abstract

This paper presents the complexity analysis and empirical results of a distributed selection algorithm .

The algorithm uses the statistical properties of the data file. The objective of the algorithm is to minimize the number of communication messages required for the whole selection process.

The algorithm is designed to select the y th smallest key from a very large file which is physically distributed over many sites (stations). The size of the file is so large that it is not feasible or efficient to transfer all data to a single node as no node has sufficient memory space for internal sorting. The selection work will be shared by all sites involved and the load balancing is also ensured by the algorithm.

The complexity of the algorithm is O(log log N/P) for a network with P stations and a file with N records.

Introduction

Many organizations are using distributed systems nowadays due to the advancement of hardware and software technology. The distributed systems are providing many advantages which can not be offered by the traditional central computer systems.

Distributed databases operations becomes more important as it will affect the efficiency of distributed information system. Selection is a common and essential step in many scientific and engineering operations. Selection algorithms are also used in some sorting algorithms[2,4]. Thus it is worthwhile to improve the distributed sorting algorithms.

As the communication time is much longer than the internal processing time of each computer, most algorithms[1,2,3,4] for distributed systems aims to reduce the communication time by reducing the number of communication messages required in the process.

The algorithm discussed in this paper can be used in distributed system with broadcasting facilities or parallel computers with loosely coupled multiprocessors and a broadcasting channel. However, it can be modified so that it will be suitable for other distributed architectures.

The Distribution System Model

The model was presented in [4] and is described again for easy reference as follows :

- a large file is physically distributed in P sites(workstations).

- N records are approximately evenly distributed in P sites. (i.e. N/P records for each site.)

- no computer in any site has enough resource to collect all records and sort the file locally (or willing to use the resource as each computer will be working on other processes for other users at the same time.)

- a single channel is available for broadcast communication.

- each record consists of a single element (note: in order to simplify the illustration of the algorithm only and it is not a constraint of the model).

- we denote the elements in the file as X(i,j)
  where i - site number and $1 \leq i \leq P$
  j - index of records and $1 \leq j \leq N/P$

- X(i,j) elements are sorted so that the elements will be in the following order :

$X(1,1) < X(1,2) < \ldots\ldots < X(1,N/P)$
$X(2,1) < X(2,2) < \ldots\ldots < X(2,N/P)$
$X(3,1) < X(3,2) < \ldots\ldots < X(3,N/P)$

$X(P,1) < X(P,2) < \ldots\ldots < X(P,N/P)$

and $X(k,N/P) < X(k+1,1)$ for all k
where k is an integer from 1 to P-1

- the elements follows uniform distribution (the algorithm will also be suitable for other distribution by minor modification, uniform distribution will be used as an example in this paper.)

The objective is to find elements with ranks which are stored in an array Y(j).

Distributed Selection Algorithm

This algorithm was first presented in [4] as a sub-step in a distributed sorting algorithm.. It is fine-tuned as follows:

step 0:
(note: This is an optional step. If the maximum and minimum value are known , this step is not required.)

One processor will be selected as coordinator. It will calculate the initial D[j] values.
M : Maximum of the key values in the file
m : Minimum of the key values in the file

step 1
calculate
$D(j) = m + (M- m)*j /P$
where $1 <= j <= P-1$

step 2:
The coordinator broadcast the D(j) values to other processors.

step 3:
Each site select Pivot(i,j).
Pivot(i,j) is the biggest value which is less than or equal to D(j)

step 4
Other sites broadcast the rank R(i,j) of Pivot(i,j) in it's local elements (i.e. the numbers of local elements which is smaller than Pivot[j].

step 5
Each site recalculates Pivot(i,j) if $\sum_{i=1}^{P} R[i,j]$ does not equal Y(j).
target(j)= Pivot(i,j) if $\sum R[i,j]$ equal to Y(j)
Calculate a new Pivot(i,j)
Go back to step 4 until all target(j) satisfy this condition.

The new Pivot(j) will be calculated as follows :

After each probe, each local file will be splitted into two sub-files. The new pivot will be on the left or right sub-file depending on whether $\sum R[i,j]$ is less than Y(j) or greater than Y(j) respectively.

(note: the smallest key is on the left side and the largest key will be on the right side.)
New value = old pivot value +
(largest key of the sub-list - smallest key of the sub-list)* offset /
(position of old pivot - right side or left side index of the sub-list)

Complexity Analysis

Theorem 1 :

By definition, the number of records in each site is N/P

The expect number of probe to find the key or empty the segment is $LogLog\left(\frac{N}{P}\right)$

Proof:

The process is a reverse case of interpolation search. The search problem is to find the relative position of the record given the value of key. The selection problem is to find the value of key given the desired rank. According to Perl[5] and Gonnet[6], the average number of searches required to find a key using interpolation search of a file of size M is LogLog M. Perl[5] proved the LogLog performance by computing the average square distance between each probe position and the position of the desired key. Gonnet[6] indicated the same performance by approximating the probability of the success of each probe. Although the distributed selected is more complex, Theorem 1 can still be proved by modifying and extending the approach by Perl[5] as follow.

Denote :

i = index of the sites

j = index of target

k = index of the probe.

P = number of processors

$\alpha_{i,j,k}$ = probability of any key in the site i which is less than or equal to target (j) in the k*th* probe.

$F_{i,j,k}$ = sub-file in site i of the probe in the process of searching for target (j)

$L_{i,j,k}$ = the index of the smallest key in the sub-file of site i after the k_{th} probe in $F_{i,j,k}$

$H_{i,j,k}$ = the index of the largest key in the sub-file of site i after the k_{th} probe in $F_{i,j,k}$

$n_{i,j,k}$ = number of keys in the sub-file in $F_{i,j,k}$

$q_{i,k,k} = 1-\alpha_{i,j,k}$

$S_{i,j,k}$ = number of any key which is less than or equal to pivot (i,j,k) in $F_{i,j,k}$

pivot(i,j,k)= the key which is broadcasted in site i for the k_{th} probe in $F_{i,j,k}$

By definition, Target is the Y(j) th key in the file, thus

$$\alpha_{i,j,k} = \frac{\sum_{i=1}^{p} \theta_{i,j,k-1} - Y(j)}{p} \qquad \text{when } k > 1$$

and

$$\alpha_{i,j,k} = Y(j) / N \qquad \text{when } k = 1$$

If we define a key comparison is a success when it is less than the pivot key and otherwise a failure, then $S_{i,j,k}$ will follow a binomial distribution[7].

$$E(S_{i,j,k}) = \sum_{m=1}^{n_{i,j,k}} m(\alpha_{i,j,k})^m (1-\alpha_{i,j,k})^{n_{i,j,k}-m}$$

$$= n_{i,j,k} \ \alpha_{i,j,k}$$

The variance of $S_{i,j,k}$ is $n_{i,j,k} \ \alpha_{i,j,k}(1-\alpha_{i,j,k})$

Denote $\theta' = L_{i,j,k} + S_{i,j,k}$

$$\theta_{i,j,k} = E\left(\theta' \middle| F_{i,j,1}, F_{i,j,2}, \ldots\ldots, F_{i,j,k}\right) \qquad (1.1)$$

Define $V_{i,j,k} = \left|\theta_{i,j,k+1} - \theta_{i,j,k}\right|$ (1.2)

$$\theta_{i,j,k+1} = L_{i,j,k+1} + n_{i,j,k+1} * \alpha_{i,j,k+1} \qquad (1.3)$$

and

$$\theta_{i,j,k} = L_{i,j,k} \quad \text{if } \sum_{i=1}^{p} \theta_{i,j,k} < Y(j) \qquad (1.4)$$

$$\theta_{i,j,k} = H_{i,j,k} \quad \text{if } \sum_{i=1}^{p} \theta_{i,j,k} > Y(j) \qquad (1.5)$$

Using (1.3),(1.4) and (1.5), we have

$$V_{i,j,k} = n_{i,j,k+1}\alpha_{i,j,k+1} \quad \text{if } \sum_{i=1}^{p}\theta_{i,j,k} < Y(j)$$

or

$$V_{i,j,k} = n_{i,j,k+1}q_{i,j,k+1} \qquad \text{if } \sum_{i=1}^{p}\theta_{i,j,k} > Y(j)$$

Using (1.1), We have

$$\theta_{i,j,k+1} = E\left(\theta^1 \middle| F_{i,j,1} \ldots\ldots F_{i,j,k+1}\right) \tag{1.6}$$

$$V_{i,j,k} = \left|\theta_{i,j,k+1} - \theta_{i,j,k}\right|$$

$$= \left|E\left(\theta^1 - \theta_{i,j,k} \middle| F_{i,j,1,} \ldots\ldots F_{i,j,k+1}\right)\right| \quad \text{using (1.6)}$$

$$E\left(V^2{}_{i,j,k} \middle| F_{i,j,1}, \ldots\ldots F_{i,j,k}\right)$$

$$= E\left(\left(E\left(\theta' - \theta_{i,j,k} \middle| F_{i,j,1,} \ldots\ldots, F_{i,j,k}\right)\right)^2 F_{i,j,1,} \ldots\ldots F_{i,j,k}\right)$$

$$\le E\left(\left(E\left(\theta' - \theta_{i,j,k}\right)^2 \middle| F_{i,j,1,} \ldots\ldots F_{i,j,k}\right) \middle| F_{i,j,1,} \ldots\ldots F_{i,j,k}\right)$$

$$= E\left(\left(\theta' - \theta_{i,j,k}\right)^2 \middle| F_{i,j,1} \ldots\ldots F_{i,j,k}\right)$$

$$= n_{i,j,k}\alpha_{i,j,k}q_{i,j,k} \qquad \text{as } \left(\theta' - \theta_{i,j,k}\right)^2 \text{ is the variance of the distance}$$

$$E\left(V^2{}_{i,j,k} \middle| F_{i,j,1}, \ldots\ldots F_{i,j,k}\right) \le V_{i,j,k-1}$$

$$E\left(V^2{}_{i,j,1} \middle| F_{i,j,1}\right)$$

$$\le n_{i,j,1}\ \alpha_{i,j,1}\left(1 - \alpha_{i,j,1}\right)$$

$$E\left(V^2{}_{i,j,1} \middle| F_{i,j,1}\right) \qquad \le \frac{1}{4} n_{i,j,1}$$

$$\log\left(E\left(V_{i,j,k} \middle| F_{i,j,1} \ldots . F_{i,j,k}\right) \right) \le \log V_{i,j,k-1}$$

$$E\left(\log V_{i,j,k} \middle| F_{i,j,1} \ldots . F_{i,j,k}\right) \le \log V_{i,j,k-1}$$

Denote $\beta_{i,j,k} = \log V_{i,j,k}$,

$$E\left(2^k \beta_{i,j,k} \middle| F_{i,j,1} \ldots \ldots F_{i,j,k}\right)$$

$$\le 2^{k-1} \beta_{i,j,k-1}$$

Thus we have

$$E(T) < \log\log \frac{N}{P}$$

Simulation Result

Simulation were conducted by writing c++ programs. Uniform distributed data files were generated. One hundred experiments were repeated for record size of 1,000,000 . The average numbers of iteration to find the k th elements were presented in table 1-2. The theoretical prediction of the iterations were also listed with the experimental results in the tables.

We compare the theoretical and experimental results in Figure 1 to 2. As we can see from these figures, the experimental results follows our prediction.

Size = 1,000,000

Experiment Result				Prediction
Proc.	Best	Worst	SD. Iterations	Iterations
3	2	7	1.06 3.88	4.20
4	2	7	1.02 3.81	4.16
5	1	7	1 3.59	4.14
6	1	8	1.06 3.46	4.12
7	1	7	1.01 3.42	4.10
8	1	6	0.96 3.35	4.08
9	1	7	0.98 3.29	4.07

Table 1

Size = 1,000,000

Experiment Result					Prediction
Proc.	Best	Worst	SD.	Iterations	Iterations
10	1	7	0.96	3.26	4.05
20	1	7	0.88	2.97	3.96
30	1	6	0.87	2.82	3.91
40	1	6	0.82	2.7	3.87
50	1	6	0.82	2.62	3.84
60	1	6	0.81	2.58	3.81
70	1	6	0.81	2.54	3.79
80	1	6	0.81	2.49	3.77
90	1	6	0.81	2.47	3.75
100	1	6	0.82	2.43	3.73
110	1	6	0.81	2.41	3.72
120	1	6	0.8	2.39	3.70

Table 2

Average number of iterations required in 100 experiment

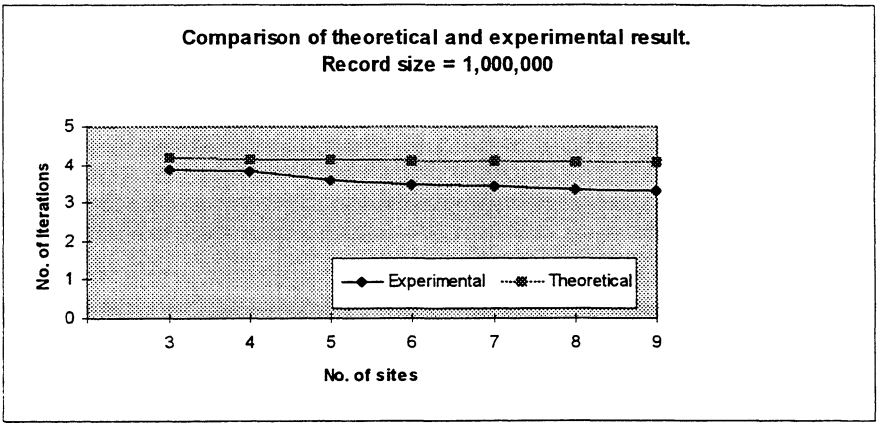

Fig. 1

Average number of iterations required in 100 experiment

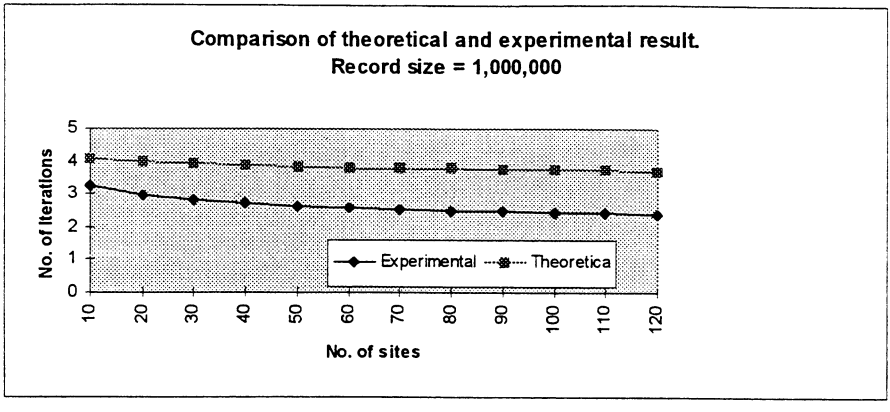

Fig. 2

Reference

1. Dechter, R.; Kleinrock, L.;Broad Communications and Distributed Algorithms; IEEE Transactions Computers, vol c- 35, no.3, 1986.

2. Huang,J.H.; Kleinrock,L.; Distributed Selectsort Sorting Algorithms on Broadcast Communication networks; Parallel Computing 1990.

3. Wegner,L.M.; Sorting a Distributed file in a Network; Proc. of Conf. Information Sci System, Princeton, New Jersey; 1982.

4 Loo, Alfred W.S.; Jim M. Ng; Distributed Statistical Sorting Algorithms; Proceeding of Sicon; Singapore ;1991.

5. Y. Perl, Interpolation Search - A LogLog n Search ,Comm ACM 21 (1978) 550-553.

6. Gonet, H; Roger, D; George, A; An Algorithmic and Complexity Analysis of Interpolation Search, Acta Information, 1980.

7. James Higging, Concepts in Probability and Stochastic Modeling (p.122), Duxbury Press , 1995.