Message Complexity of Simple Ring-based Election Algorithms – an Empirical Analysis

(extended abstract)

Friedemann Mattern

Department of Computer Science, University of Kaiserslautern,

P.O. Box 3049, D 6750 Kaiserslautern, West-Germany

Abstract

Several variants of the simple and well-known Chang-Roberts algorithm have been simulated. The empirical analysis shows that the algorithms compare very favorably with other ring-based election algorithms. For various ring sizes and number of concurrent starters the average message complexity, its distribution, and its standard deviation are studied.

1. Introduction

The *election problem* is a fundamental problem of distributed computing. It can roughly be characterized as distributed mutual exclusion without fairness requirements. The election problem arises whenever several uniquely identified but otherwise "identical" processes have to agree on a (temporary) *leader*. It is usually assumed that the process identifications are linearly ordered but that initially no process knows the identities of all other processes. Without loss of generality, the leader to be found is the process with the largest identity.

The election principle is often used as a mechanism for *breaking symmetry* in other (symmetric) distributed algorithms (e.g. [MAT89]). Other applications are concurrency control, regeneration of a lost (unique) token, and system initialization and recovery by electing a new coordinator after a crash or coordinator failure. The problem is to design an efficient distributed algorithm which can be initiated by an arbitrary process independently of any other process.

The election problem has been studied mainly for circular configurations. The ring-based solutions presented in the literature can be partitioned into two classes: In the first class it is assumed that messages can only be sent in one direction (*unidirectional* ring), in the second class messages can be sent to both directions (*bidirectional* ring). The complexity of the solution is usually measured in terms of the total number of messages exchanged during the execution of the algorithm.

Because for most algorithms the exact number of messages depends on the specific arrangement of process identifications on the ring, the *worst case* and the *average case* message complexity are of interest. Averaging is done by considering all $(n-1)!$ different arrangement of n processes on the ring equally likely. Furthermore, for *complexity analysis* it is assumed that all starting processes initiate the algorithm at the same moment and that all messages take the same time traversing a link between two processes. But obviously an election algorithm should also work correctly without these assumptions.

Table 1 reports the most significant results, an overview on other ring-based election algorithms may be found in [BOL86, BOD87, LAV88]. Notice that $\log n \approx H_n$ (the n-th harmonic number) and that all logarithms are to the base e.

Besides message complexity, the simplicity of a solution is also important. The *Chang-Roberts algorithm* [CHR79] and its bidirectional variants are very simple and have a very good *average case complexity*, despite their $O(n^2)$ worst case complexity. In [RKS87] Rotem et al. have shown that these algorithms require $O(n \log n)$ messages with very high probability, and recently Lavault has shown that the asymptotic message complexity of the bidirectional variant is $\frac{1}{2}\sqrt{2}\, n \log n$ [LAV88]. Therefore, the Chang-Roberts algorithm compares very favorably with other election algorithms.

Our empirical analysis of the Chang-Roberts algorithm and its bidirectional variants complements the previous theoretical analysis. We analyze the average message complexity, its distribution, and its standard deviation for several typical and extreme situations. The results not only confirm the observations by Rotem et al., but show that the algorithm is far better than the rather conservative mathematical estimations indicate.

2. The Chang-Roberts Algorithm

The unidirectional Chang-Roberts algorithm [CHR79] (which we call UNI) is based on the principle of selective message extinction. We present a variant of the original algorithm where a *subset* of the processes participate in the election. Any process which is not yet engaged in the election may initiate the algorithm independently of any other process. It starts the algorithm by becoming "active" and sending a message with its own identification to its neighboring process. A process receiving a message compares the number on the message with the highest number it has seen so far (including its own identification if it is a starter) stored in a local variable M. If M is larger, the process simply throws the message away. If M is smaller, it updates M and passes

CH2706-0/89/0000/0094$01.00 © 1989 IEEE

Algorithm	Average Case	Worst Case
Unidirectional Algorithms		
Chang-Roberts [CHR79]	nH_n (average case optimal)	$\frac{n(n+1)}{2}$
Petersen [PET82]	$1.360n\log n$ (empirical result [EVE84])	$2.078n\log n + O(n)$ (empirical result [EVE84])
Petersen (variant) [DKR82]	$1.395n\log n$ (empirical result [EVE84])	$1.956n\log n + O(n)$ (empirical result [EVE84])
lower bound	nH_n [PKR84]	
Bidirectional Algorithms		
Bidirectional variant of Chang-Roberts (Rotem et al. [RKS87], Bodlaender and van Leeuwen [BOL86])	$0.7071nH_n + O(n)$ [LAV88]	$\frac{n(n+1)}{2}$ (probabilistic) ~ $0.25n^2$ (deterministic) [BOL86]
Moran et al. [MSZ86], van Leeuwen and Tan [LET87]		$2.078n\log n + O(n)$
lower bound	$0.5nH_n$ [BOD87, BOD88]	

Table 1. Significant election algorithms.

the message on. A process has won the election if it gets back a message with its own identification having made a full round.

Without loss of generality we assume that each of the n processes is uniquely identified by an integer 1,...,n. The local variable M is initialized to 0. A process may start the algorithm as long as M=0. (If a process has not initiated the algorithm by the time a message reaches it, then it does not participate in the current election round). Each process i executes the same local algorithm:

```
Algorithm UNI for process i:
    Start: M := i;
           send <i> to neighbor;
    Upon receiving a message <j>:
           if M<j then M:=j;
                       send <j> to neighbor;
                  elseif M=j then leader;
           endif;
```

The correctness of the algorithm is easily verified: A message is eliminated by the first larger active process encountered. Thus all messages except the message with the highest value are eliminated on their way around the ring. At the end of the algorithm every process knows the identity of the leader and the leader knows that it has won the election. Notice that the FIFO property is not required.

2.1 Message Complexity

Clearly, the worst case message complexity for k starting processes arises when the participating processes are ordered in decreasing sequence and a message initiated by an active process must traverse all smaller processes and all passive processes. Therefore, the worst case message complexity $\hat{m}$ of Algorithm UNI is

$$\hat{m} = nk - \frac{k(k-1)}{2}.$$

If the processes are randomly ordered, a message initiated by the i-th highest active process traverses $\frac{n}{i}$ links on the average before being discarded by a higher process. Because

$$\frac{n}{1} + \frac{n}{2} + \cdots \frac{n}{k} = n\sum_{i=1}^{k}\frac{1}{i} = nH_k.$$

the average message complexity is nH_k.

Pachl et al. [PKR84] show that (for $k=n$) nH_n is a lower bound for the message complexity of the unidirectional election problem. Therefore, algorithm UNI is average case optimal. This suggests that the quadratic message complexity is an exception and that in *most* cases (i.e. for most configurations) the message complexity is about nH_k. In fact, Rotem et al. [RKS87] proved that (for $k=n$) the algorithm uses $O(nH_n)$ messages with probability tending to one. An interesting problem is therefore the quantification of the "average deviation" from the mean message complexity.

In order to assess the algorithm, the distribution and the standard deviation (or the variance) of the expected number of messages are of interest. To be able to compare the algorithm to other algorithms and to different bidirectional variants, we use the coefficient c of $m=cnH_k$ as a unit of measure. That is, for given ring size n, number of starters k, and number of messages m we define the coefficient c as

$$c = \frac{m}{nH_k}.$$

From previous considerations it follows that the average value $\bar{c}$ for Algorithm UNI is $\bar{c}=1$, and the maximal value $\hat{c}$ for Algorithm UNI is

$$\hat{c} = \frac{k}{H_k} - \frac{k(k-1)}{2nH_k}.$$

In order to estimate the deviation from $\bar{c}=1$ we are interested in the *distribution* of c. Figure 1 illustrates the situation we expect. The results of 100000 simulation experiments for $n=100$ and $k=100$ are shown in Table 2 (column UNI).

Fig. 1. An illustration of the distribution.

	UNI	RND	MIN
0% (=0) exceed	1.637	1.401	1.388
0.01% (=10) exceed	1.583	1.313	1.207
0.1% (=100) exceed	1.471	1.197	1.095
1% (=1000) exceed	1.317	1.070	0.995
5% (=5000) exceed	1.201	0.978	0.900
10% (=10000) exceed	1.143	0.933	0.858
30% (=30000) exceed	1.045	0.854	0.783
50% (=50000) exceed	0.987	0.814	0.746
75% (=75000) exceed	0.923	0.771	0.709
90% (=90000) exceed	0.877	0.742	0.682
100% (=100000) exceed	0.729	0.648	0.600
Average value	1.000	0.828	0.760

Table 2. Cumulative distribution of coefficient c for n=k=100.

This demonstrates that in general c is rather close to the average value $\bar{c}=1$. The observed maximal value of c was only 1.637, although theoretically it could be as large as $\hat{c} \approx 9.735$ for $n=k=100$. Table 3 shows the values for coefficient c which were not exceeded in 1% of all cases for other values of n and k. Each entry of the table is based on at least 5000 simulation runs. The figures confirm the observation that large deviations from $\bar{c}=1$ are highly improbable.

	n = 30	n = 100	n = 300	n = 1000
k = 2	1.31	1.33	1.33	1.33
k = 0.1n	1.51	1.50	1.40	1.33
k = 0.5n	1.48	1.37	1.27	1.24
k = n	1.40	1.32	1.17	1.27

Table 3. Upper bounds of the 99% interval.

2.2 The Standard Deviation

A measure of the extent to which the possible values of c are dispersed around the mean $\bar{c}=1$ is the standard deviation σ or the variance σ^2. In [RKS87] it is shown that σ^2 is the sum of hypergeometric distributed random variables. A similar derivation whose details are omitted here yields the general formula

$$\sigma = \frac{1}{H_k}\sqrt{\frac{1}{n}\sum_{j=1}^{k}\frac{(n-j)(j-1)}{j^2(j+1)}}.$$

For small values of k, it is easy to deduce specific and simpler formulas for the standard deviation σ from the general formula. Some of these formulas are listed in Table 4 together with their limit for $n \to \infty$.

The following estimation (which is analogue to a similar estimation in [RKS87]) demonstrates that σ can be bounded ($k>1$):

$$\sigma = \frac{1}{H_k}\sqrt{\frac{1}{n}\sum_{j=1}^{k}\frac{(n-j)(j-1)}{j^2(j+1)}} < \frac{1}{H_k}\sqrt{\frac{1}{n}\sum_{j=2}^{\infty}\frac{n}{(j+1)^2}} =$$

$$\frac{1}{H_k}\sqrt{\sum_{j=3}^{\infty}\frac{1}{j^2}} = \frac{1}{H_k}\sqrt{\frac{\pi^2}{6}-\frac{5}{4}} < \frac{0.629}{H_k} < \frac{0.629}{\log k}$$

Hence,

$$\sigma < \frac{0.629}{\log k} \qquad (A)$$

k	σ	$\lim_{n\to\infty}\sigma$
1	0	0
2	$\sqrt{\frac{n-2}{27n}}$	$\frac{\sqrt{3}}{9} \approx 0.19245$
3	$\frac{1}{11}\sqrt{5-\frac{12}{n}}$	$\frac{\sqrt{5}}{11} \approx 0.20328$
4	$\frac{1}{25}\sqrt{\frac{127n-348}{5n}}$	$\frac{1}{25}\sqrt{\frac{127}{5}} \approx 0.20159$

Table 4. Formulas for σ (Algorithm UNI).

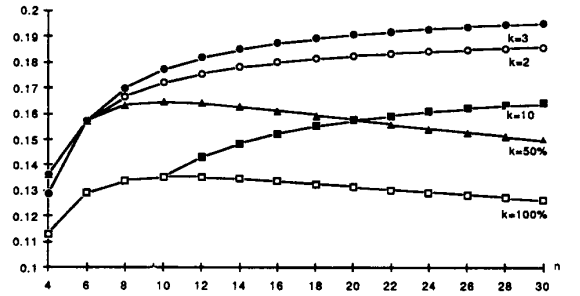

Fig. 2. The standard deviation σ for small rings.

In figures 2 and 3 σ is analyzed in more detail, they show σ plotted against the ring size n for different values of k. In accordance with (A), σ approaches 0 with increasing ring size if k is a constant fraction of n. It converges to some value > 0, however, if the number of starters remains fixed. k=3 leads to the highest asymptotic standard deviation, therefore its limit $\frac{\sqrt{5}}{11} \approx 0.20328$ is an upper bound for *all* values n, k:

$$\sigma < 0.2033 \qquad (B)$$

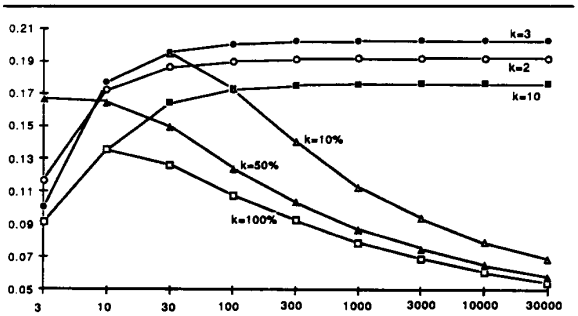

Fig. 3. The standard deviation σ for larger rings.

The probability that the values of c lie within a certain distance of $\bar{c}=1$ can be estimated by using the *Chebychev Inequality*. In a form which is convenient for our purposes it states that for any $\varepsilon>0$

$$\text{prob}(c>1+\varepsilon) \le \frac{\sigma^2}{\varepsilon^2}. \qquad \text{(C)}$$

By substituting (B) in (C) we get

$$\text{prob}(c>1+\varepsilon) < \frac{0.042}{\varepsilon^2}. \qquad \text{(D)}$$

Based on estimations similar to (C) and (D), Rotem et al. state that "the algorithm compares very favorably with the best-known ones and should be considered in many cases to be the algorithm of choice because of its simplicity..." [RKS87]. Our empirical analysis of the cumulative distribution and the analysis of the standard deviation underline this statement – the algorithm is even far better than the analytical estimations based on the rather conservative Chebychev Inequality indicate: For $\varepsilon=1$ we get prob(c>2)<0.042, whereas Table 2 shows that for $n=k=100$ the value $c=1.637$ was never exceeded in 100000 experiments!

3. Bidirectional Election Algorithms

3.1 A Probabilistic Algorithm

On bidirectional rings elections require less messages. An obvious bidirectional version of algorithm UNI is the following probabilistic algorithm [RKS87, BOL86, LAV88]. Only the starting phase is different from Algorithm UNI. Because the local actions are assumed to be atomic taking no time, we can assume that messages are not received simultaneously from both neighbors.

Algorithm RND for process i:
  Start: M := i;
    send <i> **to** *one of the two neighbors*;
  Upon receiving a message <j>:
    if M<j **then** M:=j;
      send <j> **to** next;
    elseif M=j **then** *leader*;
    endif;

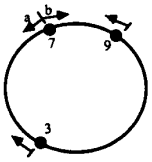

Fig. 4. Half way eliminations.

Clearly, the worst case message complexity is the same as in algorithm UNI. However, a message will be eliminated earlier if it runs into a larger message traveling in the opposite direction. Figure 4 illustrates the situation. If on the *unidirectional* ring the message initiated by process 3 is eliminated by process 7, then on a *bidirectional* ring we can expect that the two messages meet "half way" if process 7 decides to send its message in the opposite direction (case a). We assume that a process choses a starting direction with equal probability ½ among its two neighbors. Therefore, with probability ½ message 3 travels only half the distance which should save 25% of the total number of messages on the average. However, even if process 7 sends its message in the *same* direction as process 3 (case b), there is a positive probability that another process (process 9 in Figure 4) meets message 3 before it reaches process 7. Ignoring the effect of these "higher order eliminators", and assuming that messages traveling in opposite directions meet half way (i.e., they are started at the same time and travel at the same speed), the average total number of messages $\bar{m}$ can be estimated by

$$\bar{m} \le n+\frac{1}{2}\alpha+\frac{1}{2}\beta$$

The first term n comes from the highest message which travels n links in any case. α is the accumulated distance of all messages which are *not* stopped by a message traveling in the opposite direction, hence

$$\alpha = \frac{1}{2}n+\frac{1}{3}n+\cdots+\frac{1}{k}n,$$

and β is the accumulated distance of messages being eliminated half way

$$\beta = (\frac{1}{2}\frac{1}{2}n+\frac{1}{2})+\cdots+(\frac{1}{2}\frac{1}{k}n+\frac{1}{2})$$
$$= \frac{1}{4}(\frac{1}{2}n+\frac{1}{3}n+\cdots+\frac{1}{k}n)+\frac{1}{4}(k-1).$$

This yields

$$\bar{m} \le \frac{3}{4}nH_k+\frac{n+k-1}{4} \quad \text{and}$$
$$\bar{c} \le \frac{3}{4}+\frac{n+k-1}{4nH_k}. \qquad \text{(E)}$$

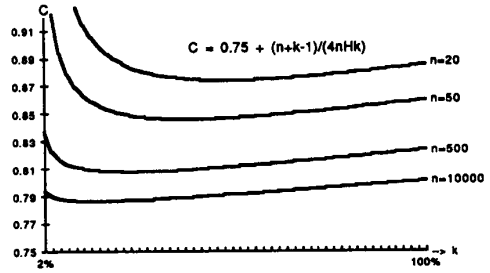

Fig. 5. Upper bounds.

Figure 5 shows the upper bound (E) for different values of n and k. In [BOL86] Bodlaender and van Leeuwen give a more precise analysis of the algorithm by taking into account the effect of some higher order eliminators. Recently, Lavault presented an exact analysis of the asymptotic complexity [LAV88].

3.2 Deterministic Variants

If the identities of the neighboring processes are known, deterministic variants are conceivable which are more efficient than Algorithm RND. By comparing its own

identification with the identification of the two neighbors an initiator can act according to one of several possible strategies. Figure 6 enumerates some of these strategies. They differ only in the starting phase.

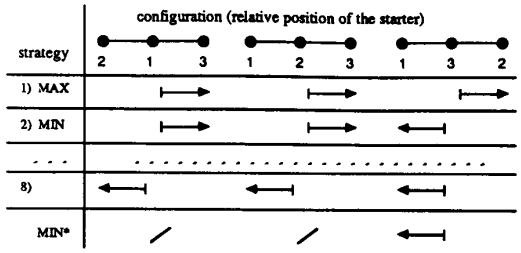

Fig. 6. Starting strategies for bidirectional rings with neighborhood knowledge.

In general ($n \geq 3$) there are 3 cases: The starting node can be smaller than both neighbors, it can be larger than both and it can be between them. For all three cases, the strategy determines the neighbor to which the initial message will be directed. Not all strategies make sense. Note that in general ($k < n$) a starter does not know whether a neighbor is also a starter. The two deterministic strategies named "MAX" and "MIN" have been analyzed in detail. MIN is a variant of Bodlaender and van Leeuwen's Algorithm-D [BOL86] which we have named "MIN*". MIN* assumes that *all* processes are starters. Using this strategy a process does not start the algorithm or propagate a message if it knows that one of its neighbor's identity is greater than its own. MIN* uses less messages than other strategies but it does only work if $k = n$.

3.3 Experimental Analysis of Coefficient c

Table 5 shows the experimentally determined average values of coefficient c for algorithms RND, MAX, MIN, and MIN* for $k = n$. The last line tells the number of experiments that have been conducted for ring size n.

Figure 7 and Figure 8 show the information of Table 5 in a graphical form. Obviously, the deterministic variants are much more efficient than Algorithm RND. The figures show that the asymptotic value 0.7071... of coefficient c as derived by Lavault [LAV88] is approximated very slowly – the asymptotic value is merely of theoretical interest.

Several series of experiments have been conducted to study the behavior of coefficient c for $k < n$. Figure 9 shows $\bar{c}$ for different values of k. Each sampling point has been

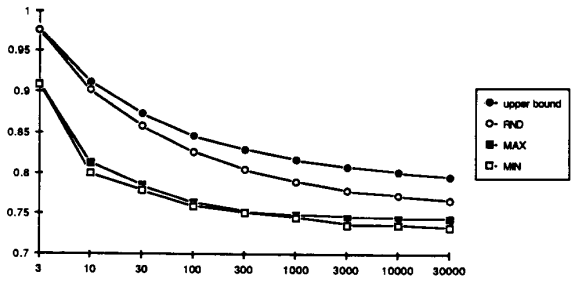

Fig. 7. Coefficient c for bidirectional algorithms.

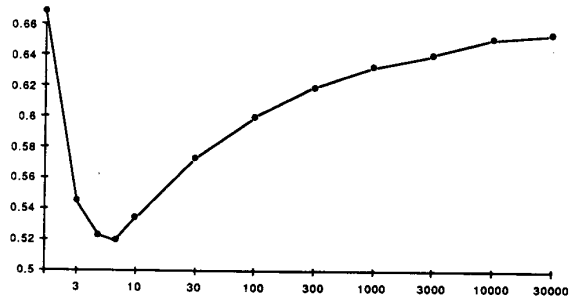

Fig. 8. Coefficient c for algorithm MIN*.

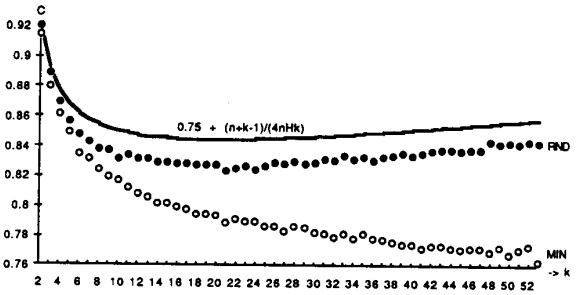

Fig. 9. Coefficient c for algorithms RND and MIN for different numbers of starters.

obtained by averaging over 10000 experiments. Whereas $\bar{c}_{MIN}$ is monotonically decreasing, $\bar{c}_{RND}$ is more or less constant over a large range.

n	3	10	30	100	300	1000	3000	10000	25000
RND	0.977	0.902	0.858	0.828	0.804	0.790	0.778	0.772	0.767
coeff. of var.	*9.00%*	*11.31%*	*10.87%*	*9.18%*	*8.36%*	*6.99%*	*6.08%*	*5.25%*	*5.19%*
MAX	0.909	0.813	0.784	0.764	0.753	0.749	0.746	0.744	0.744
coeff. of var.	*0.00%*	*8.23%*	*9.67%*	*9.41%*	*8.40%*	*7.57%*	*6.79%*	*6.38%*	*5.77%*
MIN	0.909	0.800	0.779	0.760	0.752	0.745	0.736	0.736	0.732
coeff. of var.	*0.00%*	*8.55%*	*9.87%*	*9.32%*	*8.19%*	*7.62%*	*6.31%*	*5.84%*	*5.38%*
MIN*	0.546	0.528	0.573	0.600	0.619	0.632	0.641	0.651	0.655
coeff. of var.	*0.00%*	*13.63%*	*13.54%*	*11.86%*	*9.97%*	*8.81%*	*7.73%*	*6.60%*	*6.02%*
N	9900	9900	9900	9900	9900	9900	2500	1000	120

Table 5. Coefficient c for different bidirectional algorithms.

n	10	30	100	300	1000	3000	10000	25000
RND (k=0.1n)	1.000	0.891	0.825	0.796	0.774	0.764	0.754	0.746
coeff. of var.	*0.00%*	*18.38%*	*15.78%*	*12.89%*	*10.45%*	*8.90%*	*7.42%*	*5.95%*
MIN (k=0.1n)	1.000	0.882	0.816	0.785	0.769	0.758	0.747	0.747
coeff. of var.	*0.00%*	*17.68%*	*15.50%*	*12.42%*	*10.42%*	*8.86%*	*7.41%*	*7.75%*
RND (k=3)	0.911	0.889	0.884	0.881	0.879	0.878		
coeff. of var.	*16.31%*	*18.42%*	*19.33%*	*19.36%*	*19.76%*	*19.64%*		
MIN (k=3)	0.879	0.877	0.879	0.883	0.881			
coeff. of var.	*14.23%*	*17.49%*	*18.72%*	*19.50%*	*19.23%*			
N	9900	9900	9900	9900	9900	2500	1000	120

Table 6. Coefficient c for a smaller number of starters.

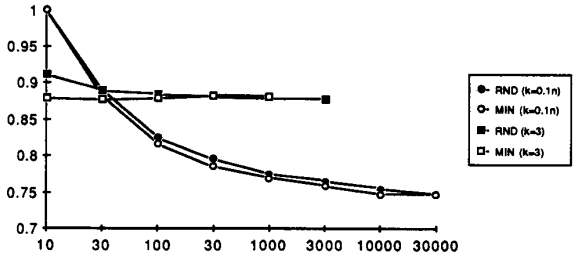

Fig. 10. Coefficient c for a smaller number of starters.

Table 6 lists the experimental results for the same series of experiments as shown in Table 5 and Figure 7 but for $k=3$ and $k=0.1n$ starters. Figure 10 displays the information in graphical form. By comparing figures 10 and 7 one sees that the graphs for $k=n$ and $k=0.1n$ are rather similar. For $k=3$ the two strategies MIN and RND behave almost identical for large rings. This is to be expected because the knowledge of the neighbor's identities does not convey much information if the probability for a neighbor being a starter is low.

The experimentally obtained cumulative distribution of c_{RND} and c_{MIN} for $k=n=100$ is shown in Table 2 together with the cumulative distribution of c_{UNI}. It again shows the superiority of Algorithm MIN over Algorithm RND, but qualitatively the three algorithms are very similar: In over 50% of all cases the values are below $\overline{c}$, and $2\overline{c}$ was never reached or exceeded in 100000 experiments.

In order to be able to compare the standard deviation of different algorithms, we use the *relative* standard deviation (the so-called *coefficient of variation*). The coefficient of variation *cvar* of a random variable X with mean μ and standard deviation σ is defined by $cvar=\frac{\sigma}{\mu}$. Notice that since for algorithm UNI $\overline{c}=1$, the standard deviation and the coefficient of variation are identical. Tables 5 and 6 list the experimentally obtained coefficient of variation for algorithms RND and MIN. The values are shown in a graphical form in Figure 11.

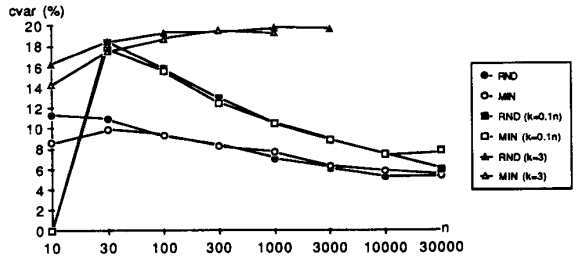

Fig. 11. Coefficient of variation for algorithms RND and MIN.

A comparison of this figure to Figure 3 shows that the coefficient of variation for unidirectional and bidirectional algorithms are very similar.

4. Conclusions

We have presented an empirical analysis of the Chang-Roberts election algorithm and several bidirectional variants. While a mathematical analysis of its behavior requires powerful and elaborate methods from the theory of combinatorial enumeration [LAV88], our simulations demonstrate that the estimations are often rather conservative. The simulations show that despite its simplicity the algorithm compares very favorably with other known election algorithms.

The distribution and the standard deviation for the bidirectional variant have not been analyzed before. However, our simulations show that the unidirectional and bidirectional variants behave very similar. By using the coefficient c as a unit of measure we were able to compare different variants. In particular, we analyzed the performance of the algorithm for a small number of starters which has not been done before but which is a more realistic situation in practice.

References

BOD87 BODLAENDER H.L. (1987) *New Lower Bounds for Distributed Leader Finding in Asynchronous Rings of Processors.* In: Proc. GI – 17. Jahrestagung (M. Paul, ed.), Springer-Verlag IFB 156, pp. 82-88

BOD88 BODLAENDER H.L. (1988) *A Better Lower Bound for Distributed Leader Finding in Bidirectional Asynchronous Rings of Processors.* Information Processing Letters 27, pp. 287-290

BOL86 BODLAENDER H.L., VAN LEEUWEN J. (1986) *New Upperbounds for Decentralized Extrema-Finding in a Ring of Processors.* In: Proc. STACS 86, Springer-Verlag LNCS 210, pp. 119-129

CHR79 CHANG E., ROBERTS R. (1979) *An Improved Algorithm for Decentralized Extrema-Finding in Circular Configurations of Processes.* Comm. of the ACM 22:5, pp. 281-283

DKR82 DOLEV D., KLAWE M., RODEH M. (1982) *An O(n log_2 n) Unidirectional Distributed Algorithm for Extrema-Finding in a Circle.* J. Algorithms 3, pp. 245-260

EVE84 EVERHARDT P. (1984) *Average Case Behavior of Distributed Extrema-Finding Algorithms.* Technical Report ACT-49/T-147, Coordinated Science Laboratory, University of Illinois at Urbana-Champaign, USA

LAV88 LAVAULT C. (1988) *Average Number of Messages for Distributed Leader Finding in Rings of Processors.* Internal Report 840, INRIA, France

LET87 VAN LEEUWEN J., TAN R. (1987) *An Improved Upperbound for Distributed Election in Bidirectional Rings of Processors.* Distributed Computing 2, pp. 149-160

MAT89 MATTERN F. (1989) *Asynchronous Distributed Termination – Parallel and Symmetric Solutions with Echo Algorithms.* Algorithmica (to appear)

MSZ86 MORAN S., SHALOM M., ZAKS S. (1986) *An 1.44...NlogN Algorithm for Distributed Leader Finding in Bidirectional Rings of Processors.* Technical Report RC 11933, IBM Th. J. Watson Research Center, Yorktown Heights, New York

PET82 PETERSON G.L. (1982) *An O(nlogn) Unidirectional Algorithm for the Circular Extrema Problem.* ACM Transactions on Programming Languages and Systems 4:4, pp. 758-762

PKR84 PACHL J., KORACH E., ROTEM D. (1984) *Lower Bounds for Distributed Maximum-Finding Algorithms.* Journal of the ACM 31:4, pp. 905-918

RKS87 ROTEM D., KORACH E., SANTORO N., (1987) *Analysis of a Distributed Algorithm for Extrema Finding in a Ring.* Journal of Parallel and Distributed Computing 4, pp. 575-591

The full version of the paper is available from the author.

email: mattern@uklirb.uucp