LETTER

On the Time Complexity of Dijkstra's Three-State Mutual Exclusion Algorithm

Masahiro KIMOTO[†a)], ***Nonmember***, **Tatsuhiro TSUCHIYA**[†], ***Member***, ***and*** **Tohru KIKUNO**[†], ***Fellow***

SUMMARY In this letter we give a lower bound on the worst-case time complexity of Dijkstra's three-state mutual exclusion algorithm by specifying a concrete behavior of the algorithm. We also show that our result is more accurate than the known best bound.

key words: *analysis of algorithms, distributed computing, self-stabilization, stabilization time*

1. Introduction

Dijkstra's three-state mutual exclusion algorithm is one of the first self-stabilizing algorithms [1]. Although more than 30 years have passed since its invention, the exact worst-case time complexity of this algorithm is still unknown. In this letter we give a lower bound on the worst-case time complexity, which matches the known best bound $1\frac{5}{6}n^2 - O(n)$ [2] but is more accurate. For the reason explained later, we conjecture that the new bound is the exact worst-case time complexity.

2. The Algorithm

We consider a system consisting of n processors $p_0, p_1, \cdots, p_{n-1}$ that are arranged in a ring. Processor $p_i, (0 \le i \le n-1)$ is adjacent to $p_{(i-1) \bmod n}$ and $p_{(i+1) \bmod n}$. Processor p_i has a local state $x_i \in \{0, 1, 2\}$ and can read the state of its adjacent processors. A *configuration* is an n-tuple of process states $(x_0, x_1, \cdots, x_n)(\in \{0, 1, 2\}^n)$. Dijkstra's three-state mutual exclusion algorithm is described as follows (addition and subtraction are modulo 3):

Processor p_0:
  if $x_0 + 1 = x_1$ **then** $x_0 := x_0 + 2$
Processor $p_i, 1 \le i \le n-2$:
  if $x_{i-1} - 1 = x_i$ or $x_i = x_{i+1} - 1$ **then** $x_i := x_i + 1$
Processor p_{n-1}:
  if $x_{n-2} = x_{n-1} = x_0$ or $x_{n-2} = x_{n-1} + 1 = x_0$
  then $x_{n-1} := x_{n-2} + 1$

A processor is *enabled* if the **if** condition is true. The algorithm runs in steps. In each step, exactly one enabled processor executes the statement of the algorithm, resulting in a new configuration. We write $C \to C'$ if configuration C can move to another configuration C' in a step. An *execution* is a sequence of configurations $C_0C_1 \cdots C_l$ where $C_i \to C_{i+1}$ for any $i, 0 \le i < l$. We also write $C \rightsquigarrow C'$ if there is an execution that starts with C and leads to C'. The *length* of an execution $C_0C_1 \cdots C_l$ is l. Given an execution $C_0C_1 \cdots C_l$, a *schedule* is a sequence of processors $P_1P_2 \cdots P_l$ such that for any $i, 1 \le i \le l$, P_i is enabled in C_{i-1} and the execution of the statement by P_i in C_{i-1} yields C_i.

Since this algorithm is intended to ensure mutual exclusion, a configuration is *legitimate* if exactly one processor is enabled. A configuration is *illegitimate* if it is not legitimate.

Proposition 1: [3] Dijkstra's three-state mutual exclusion algorithm is self-stabilizing; that is, (i) a legitimate configuration occurs in any execution starting with any configuration, and (ii) if a configuration C is legitimate, then any configuration C' such that $C \rightsquigarrow C'$ is legitimate.

The worst-case time complexity (or *stabilization time* in some literature) of the algorithm is the maximum number of steps executed until a legitimate state is reached. Formally, the worst-case time complexity is the length of the longest execution $C_0C_1 \cdots C_l$ such that C_i is illegitimate for any $i, 0 \le i \le l-1$ and C_l is legitimate. Let $T(n)$ denote the worst-case time complexity of the algorithm. When n is fixed, a number $LB(n)$ is a lower bound on the worst-case time complexity if $LB(n) \le T(n)$.

3. Lower Bound

Our proof of a lower bound is rather direct: We show some very long executions where only the very last configuration is legitimate. Then we obtain the length of these executions. By definition, the worst-case time complexity is greater than or at least equal to that length; thus the length of these executions is a lower bound on the worst-case time complexity.

Our results apply when $n \ge 9$. There are three cases to consider: (1) $n = 3m$; (2) $n = 3m+1$; and (3) $n = 3m+2$. For each of these cases, we provide a long execution that consists of three parts. First we show the results for Case (1) and then proceed to the other two cases.

To make the proofs concise, we use the same notations as [2]. Notation $x_{i-1} < x_i$ means $x_i = (x_{i-1} + 1) \bmod 3$, while $x_{i-1} > x_i$ means $x_i = (x_{i-1} - 1) \bmod 3$. For example, configuration $(1, 1, 0, 1, 2, 2, 0)$ is represented as $1 = 1 > 0 < 1 < 2 = 2 < 0$. With these notations, the algorithm is represented as a collection of eight types of moves (types 0 to 7), as shown in Table 1. Regular expressions over $\{<, >, =\}$ are used to denote configurations. For example, $[=><^2=<]$ is a possible notation for $(1, 1, 0, 1, 2, 2, 0)$.

Lemma 1: When $n = 3m, n \ge 6$, there is an execution of

Manuscript received April 7, 2009.
[†]The authors are with Osaka University, Suita-shi, 565–0871 Japan.
a) E-mail: m-kimoto@ist.osaka-u.ac.jp
DOI: 10.1587/transinf.E92.D.1570

Table 1 The algorithm in a tabular form (C' is the next configuration to C, i.e., $C \to C'$).

Type	Processor	C	C'
0	p_0	$x_0 < x_1$	$x_0 > x_1$
1	p_i	$x_{i-1} > x_i = x_{i+1}$	$x_{i-1} = x_i > x_{i+1}$
2	p_i	$x_{i-1} = x_i < x_{i+1}$	$x_{i-1} < x_i = x_{i+1}$
3	p_i	$x_{i-1} > x_i < x_{i+1}$	$x_{i-1} = x_i = x_{i+1}$
4	p_i	$x_{i-1} > x_i > x_{i+1}$	$x_{i-1} = x_i < x_{i+1}$
5	p_i	$x_{i-1} < x_i < x_{i+1}$	$x_{i-1} > x_i = x_{i+1}$
6	p_{n-1}	$x_{n-2} > x_{n-1} < x_0$	$x_{n-2} < x_{n-1} > x_0$
7	p_{n-1}	$x_{n-2} = x_{n-1} = x_0$	$x_{n-2} < x_{n-1} > x_0$

length $n+3$ from $[<><^{n-3}]$ to $[==<^{n-3}]$.

Proof We show the existence of schedule $\underbrace{p_3}_{type\ 5}\ \underbrace{p_4 p_5 \cdots p_{n-2}}_{2}\ \underbrace{p_{n-1}}_{7}\ \underbrace{p_0}_{0}\ \underbrace{p_1}_{4}\ \underbrace{p_1}_{2}\ \underbrace{p_0}_{0}\ \underbrace{p_1}_{1}\ \underbrace{p_2}_{4}$.

$[<><^{n-3}]$,
$[<>>=<^{n-5}]$, after 1 step of type 5:
$[<>><^{n-5}=]$ after $n-5$ steps of type 2:
$[<>><^{n-4}]$, after 1 step of type 7 (note that $x_{n-2} = x_{n-1} = x_0$ in the previous configuration since $n = 3m$):
$[>>><^{n-4}]$, after 1 step of type 0:
$[=<><^{n-4}]$, after 1 step of type 4:
$[<=><^{n-4}]$, after 1 step of type 2:
$[>=><^{n-4}]$, after 1 step of type 0:
$[=>><^{n-4}]$, after 1 step of type 1:
$[==<^{n-3}]$, after 1 step of type 4. □

Lemma 2: When $n \geq 9$, $2 \leq k \leq n-6$, and $(n-k-1) \bmod 3 = 0$, there is an execution of length $n + 9k + 10$ from $[=^k<^{n-k-1}]$ to $[=^{k+3}<^{n-k-4}]$.

Proof We show the existence of schedule $\underbrace{p_k p_{k+1} \cdots p_{n-2}}_{type\ 2}\ \underbrace{p_{n-1}}_{7}\ \underbrace{p_{k-1} p_{k-2} \cdots p_1}_{2}\ \underbrace{p_k p_{k-1} \cdots p_2}_{2}\ \underbrace{p_{k+1} p_k \cdots p_3}_{2}\ \underbrace{p_0}_{0}\ \underbrace{p_2}_{5}\ \underbrace{p_2 p_3 \cdots p_{k+1}}_{1}\ \underbrace{p_1 p_2 \cdots p_k}_{1}\ \underbrace{p_{k+1}}_{4}\ \underbrace{p_{k+1} p_k \cdots p_1}_{2}\ \underbrace{p_{k+2} p_{k+1} \cdots p_2}_{2}\ \underbrace{p_{k+3} p_{k+2} \cdots p_3}_{2}\ \underbrace{p_0}_{0}\ \underbrace{p_2}_{5}\ \underbrace{p_2 p_3 \cdots p_{k+3}}_{1}\ \underbrace{p_1 p_2 \cdots p_{k+2}}_{1}\ \underbrace{p_{k+3}}_{4}$.

$[=^k<^{n-k-1}]$,
$[=^{k-1}<^{n-k-1}=]$, after $n-k-1$ steps of type 2:
$[=^{k-1}<^{n-k}]$, after 1 step of type 7 (note that $x_{n-2} = x_{n-1} = x_0$ in the previous configuration since $(n-k-1) \bmod 3 = 0$):
$[<=^{k-1}<^{n-k-1}]$, after $k-1$ steps of type 2:
$[<<=^{k-1}<^{n-k-2}]$, after $k-1$ steps of type 2:
$[<<<=^{k-1}<^{n-k-3}]$, after $k-1$ steps of type 2:
$[><<=^{k-1}<^{n-k-3}]$, after 1 step of type 0:
$[>>=^k<^{n-k-3}]$, after 1 step of type 5:
$[>=^k><^{n-k-3}]$, after k steps of type 1:
$[=^k>><^{n-k-3}]$, after k steps of type 1:
$[=^{k+1}<^{n-k-2}]$, after 1 step of type 4:
$[<=^{k+1}<^{n-k-3}]$, after $k+1$ steps of type 2:
$[<<=^{k+1}<^{n-k-4}]$, after $k+1$ steps of type 2:
$[<<<=^{k+1}<^{n-k-5}]$, after $k+1$ steps of type 2:
$[><<=^{k+1}<^{n-k-5}]$, after 1 step of type 0:
$[>>=^{k+2}<^{n-k-5}]$, after 1 step of type 5:
$[>=^{k+2}><^{n-k-5}]$, after $k+2$ steps of type 1:
$[=^{k+2}>><^{n-k-5}]$, after $k+2$ steps of type 1:
$[=^{k+3}<^{n-k-4}]$, after 1 step of type 4. □

Lemma 3: When $n \geq 6$, there is an execution of length $10n - 30$ from $[=^{n-4}<<<]$ to $[=^{n-4}>=>]$.

Proof We show the existence of schedule $\underbrace{p_{n-4} p_{n-3} p_{n-2}}_{type\ 2}\ \underbrace{p_{n-1}}_{7}\ \underbrace{p_{n-5} p_{n-6} \cdots p_1}_{2}\ \underbrace{p_{n-4} p_{n-5} \cdots p_2}_{2}\ \underbrace{p_{n-3} p_{n-4} \cdots p_3}_{2}\ \underbrace{p_0}_{0}\ \underbrace{p_2}_{5}\ \underbrace{p_2 p_3 \cdots p_{n-3}}_{1}\ \underbrace{p_1 p_3 \cdots p_{n-4}}_{1}\ \underbrace{p_{n-3}}_{4}\ \underbrace{p_{n-3} p_{n-4} \cdots p_1}_{2}\ \underbrace{p_{n-4} p_{n-5} \cdots p_2}_{2}\ \underbrace{p_0}_{0}\ \underbrace{p_{n-1}}_{7}\ \underbrace{p_{n-2} p_{n-3} \cdots p_3}_{2}\ \underbrace{p_2}_{5}\ \underbrace{p_2 p_3 \cdots p_{n-3}}_{1}\ \underbrace{p_1 p_2 \cdots p_{n-4}}_{1}\ \underbrace{p_{n-2}}_{1}$.

$[=^{n-4}<<<]$,
$[=^{n-5}<<<=]$, after 3 steps of type 2:
$[=^{n-5}<<<<]$, after 1 step of type 7:
$[<=^{n-5}<<<]$, after $n-5$ steps of type 2:
$[<<=^{n-5}<<]$, after $n-5$ steps of type 2:
$[<<<=^{n-5}<]$, after $n-5$ steps of type 2:
$[><<=^{n-5}<]$, after 1 step of type 0:
$[>>=^{n-4}<]$, after 1 step of type 5:
$[>=^{n-4}><]$, after $n-4$ steps of type 1:
$[=^{n-4}>><]$, after $n-4$ steps of type 1:
$[=^{n-3}<<]$, after 1 step of type 4:
$[<=^{n-3}<]$, after $n-3$ step of type 2:
$[<<=^{n-3}]$, after $n-3$ step of type 2:
$[><=^{n-3}]$, after 1 step of type 0:
$[><=^{n-4}<]$, after 1 step of type 7:
$[><<=^{n-4}]$, after $n-4$ steps of type 2:
$[>>=^{n-3}]$, after 1 step of type 5:
$[>=^{n-4}>=]$, after $n-4$ steps of type 1:
$[=^{n-4}>>=]$, after $n-4$ steps of type 1:
$[=^{n-4}>=>]$, after one step of type 1. □

Theorem 1: When $n = 3m \geq 9$, we have:

$$1\frac{5}{6}n^2 - 4\frac{1}{6}n - 2 \leq T(n)$$

Proof By Lemmas 1, 2 and 3, there is an execution such that:

(i) it is represented as: $[<><^{n-3}] \rightsquigarrow [=^2<^{n-3}] \rightsquigarrow [=^5<^{n-6}] \rightsquigarrow \cdots \rightsquigarrow [=^{n-4}<^3] \rightsquigarrow [=^{n-4}>>=] \to [=^{n-4}>=>]$, and

(ii) the length is:

$$n + 3 + \sum_{i=1}^{m-2}(n + 9(3i-1) + 10) + 10n - 30$$
$$= 1\frac{5}{6}n^2 - 4\frac{1}{6}n - 2$$

The final configuration of the execution, that is, $[=^{n-4}>=>]$, is legitimate, because only p_{n-3} is enabled. Now consider the immediate predecessor configuration to the final configuration, that is, the $(1\frac{5}{6}n^2 - 4\frac{1}{6}n - 3)$–th configuration. This configuration, represented as $[=^{n-4}>>=]$, is not a legitimate configuration, since p_{n-3} and p_{n-2} are both enabled.

From (ii) of Proposition 1, if a legitimate configuration occurs in an execution, then all successor configurations in the execution must be legitimate. Since the $(1\frac{5}{6}n^2 - 4\frac{1}{6}n - 3)$–th configuration is illegitimate, every configuration in the execution, except the final configuration, is illegitimate. Therefore the worst-case time complexity is greater than or at least equal to the length of the execution. □

For the case $n = 3m + 1$ (Case (2)) the case $n = 3m + 2$ (Case (3)), a bound is obtained in almost the same manner, except that Lemma 1 is replaced with Lemma 4 and Lemma 5, respectively.

Lemma 4: When $n = 3m + 1 \geq 7$, there is an execution of length $n + 10$ from $[<>>><^{n-5}]$ to $[===<^{n-4}]$.

Proof One such execution is $[<>>><^{n-5}] \rightarrow [<>=<^{n-4}] \rightsquigarrow [<><^{n-4}=] \rightarrow [<><^{n-3}] \rightarrow [>><^{n-3}] \rightarrow [=<^{n-2}] \rightarrow [<=<^{n-3}] \rightarrow [>=<^{n-3}] \rightsquigarrow [><<=<^{n-5}] \rightarrow [>>==<^{n-5}] \rightsquigarrow [>==><^{n-5}] \rightsquigarrow [==>><^{n-5}] \rightarrow [===<^{n-4}]$. The corresponding schedule is $\underbrace{p_3}_{type\ 4}\ \underbrace{p_3p_4\cdots p_{n-2}}_{2}\ \underbrace{p_{n-1}}_{7}\ \underbrace{p_0}_{0}\ \underbrace{p_1}_{4}\ \underbrace{p_1}_{2}\ \underbrace{p_0}_{0}\ \underbrace{p_2p_3}_{2}\ \underbrace{p_2}_{5}\ \underbrace{p_2p_3}_{1}\ \underbrace{p_1p_2}_{1}\ \underbrace{p_3}_{4}$.

Lemma 5: When $n = 3m + 2 > 8$, there is an execution of length $2n + 11$ from $[<>><^{n-4}]$ to $[====<^{n-5}]$.

Proof One such execution is $[<>><^{n-4}] \rightarrow [<>>=<^{n-6}] \rightsquigarrow [<>>><^{n-6}=] \rightarrow [<>>><^{n-5}] \rightarrow [>>>><^{n-5}] \rightarrow [=<>><^{n-5}] \rightarrow [<=>><^{n-5}] \rightarrow [>=>><^{n-5}] \rightarrow [>=>>>=<^{n-7}] \rightsquigarrow [>=>>><^{n-7}=] \rightarrow [>=>>><^{n-6}] \rightarrow [=>>>><^{n-6}] \rightarrow [==<>><^{n-6}] \rightsquigarrow [<==>><^{n-6}] \rightarrow [>==>><^{n-6}] \rightsquigarrow [==>>><^{n-6}] \rightarrow [===<><^{n-6}] \rightsquigarrow [<===><^{n-6}] \rightarrow [>===><^{n-6}] \rightsquigarrow [===>><^{n-6}] \rightarrow [====<^{n-5}]$.
The corresponding schedule is $\underbrace{p_4}_{type\ 5}\ \underbrace{p_5p_6p_7\cdots p_{n-2}}_{2}\ \underbrace{p_{n-1}}_{7}\ \underbrace{p_0}_{0}\ \underbrace{p_1}_{4}\ \underbrace{p_1}_{2}\ \underbrace{p_0}_{0}\ \underbrace{p_5}_{5}\ \underbrace{p_6p_7p_8\cdots p_{n-2}}_{2}\ \underbrace{p_{n-1}}_{7}\ \underbrace{p_1}_{1}\ \underbrace{p_2}_{4}\ \underbrace{p_2p_1}_{2}\ \underbrace{p_0}_{0}\ \underbrace{p_1p_2}_{1}\ \underbrace{p_3}_{4}\ \underbrace{p_3p_2p_1}_{2}\ \underbrace{p_0}_{0}\ \underbrace{p_1p_2p_3}_{1}\ \underbrace{p_4}_{4}$.

Theorem 2: When $n = 3m + 1, n \geq 10$, we have:

$$1\frac{5}{6}n^2 - 4\frac{1}{2}n - 1\frac{1}{3} \leq T(n)$$

Proof Consider an execution $[<>>><^{n-5}] \rightsquigarrow [=^3<^{n-4}] \rightsquigarrow [=^6<^{n-7}] \rightsquigarrow \cdots \rightsquigarrow [=^{n-4}<^3] \rightsquigarrow [=^{n-4}>>=] \rightarrow [=^{n-4}>=>]$. By Lemmas 4, 2 and 3, this execution indeed exists and its length is:

$$\begin{aligned} n + 10 + \sum_{i=1}^{m-2}(n + 9 \cdot 3i + 10) + 10n - 30 \\ = 1\frac{5}{6}n^2 - 4\frac{1}{2}n - 1\frac{1}{3} \end{aligned}$$

The final configuration $[=^{n-4}>=>]$ is legitimate, because only p_{n-3} is enabled. On the other hand, its immediate predecessor configuration $[=^{n-4}>>=]$ is illegitimate, because p_{n-3} and p_{n-2} are both enabled. Hence, from (ii) of Proposition 1, every configuration in the execution, except the final configuration, is illegitimate. Therefore the worst-case time complexity is greater than or at least equal to the length of the execution. □

Theorem 3: When $n = 3m + 2, n \geq 11$, we have:

$$1\frac{5}{6}n^2 - 3\frac{5}{6}n - 9\frac{2}{3} \leq T(n)$$

Proof Consider an execution $[<>><^{n-4}] \rightsquigarrow [=^4<^{n-5}] \rightsquigarrow [=^7<^{n-8}] \rightsquigarrow \cdots \rightsquigarrow [=^{n-4}<^3] \rightsquigarrow [=^{n-4}>>=] \rightarrow [=^{n-4}>=>]$. By Lemmas 5, 2 and 3, this execution indeed exists and its length is:

$$\begin{aligned} 2n + 11 + \sum_{i=1}^{m-2}(n + 9(3i + 1) + 10) + 10n - 30 \\ = 1\frac{5}{6}n^2 - 3\frac{5}{6}n - 9\frac{2}{3} \end{aligned}$$

The final configuration $[=^{n-4}>=>]$ is legitimate, while its immediate predecessor configuration $[=^{n-4}>>=]$ is illegitimate. By the same argument as the proof of Theorems 1 and 2, every configuration in the execution, except the final configuration, is illegitimate. Thus the worst-case time complexity is greater than or at least equal to the length of the execution. □

4. Discussion

The known best lower bound on the worst-case time complexity was given by Chernoy, Shalon and Zaks [2]. They proved lower bound of $1\frac{5}{6}n^2 - O(n)$ by showing that there is a schedule of length $1\frac{5}{6}n^2 - 10\frac{1}{6}n + 14$ when $n = 3m$. Although our bound matches $1\frac{5}{6}n^2 - O(n)$, ours is tighter than $1\frac{5}{6}n^2 - 10\frac{1}{6}n + 14$ when $n = 3m$. When $n = 3m \geq 9$, we have:

$$\begin{aligned} \left(1\frac{5}{6}n^2 - 4\frac{1}{6}n - 2\right) - \left(1\frac{5}{6}n^2 - 10\frac{1}{6}n + 14\right) \\ = 6n - 16 > 0 \end{aligned}$$

Also our result applies when $n = 3m + 1$ and $n = 3m + 2$.

It should be noted that when the paper [2] appeared, our results had already been published in a technical report [4], which is a preliminary version of this letter. Thus this letter is not an incremental improvement on the result provided in [2].

Table 2 Exact worst-case time complexity. It perfectly coincides with our lower bound for $9 \leq n \leq 20$.

n	worst-case time complexity $T(n)$
9	109
10	137
11	170
12	212
13	250
14	296
15	348
16	396
17	455
18	517
19	575
20	647

So far we have assumed that exactly one enabled process executes the statement of the algorithm in each step. This model is often referred to as the centralized scheduler model. A different model could be that any subset of enabled processes can be selected in each step, which is called the distributed scheduler model. The three-state algorithm is correct in the latter model [5]. Clearly the proposed lower bound holds under the distributed scheduler, because any execution in the centralized scheduler model is possible in the distributed scheduler model.

We obtained the executions used in our proofs by analyzing the algorithm's behavior with the NuSMV model checking tool [6]. Model checking is a state exploration-based verification technique. The use of model checking for analyzing self-stabilizing algorithms was studied in [7], [8]. We used these studies with some modifications to derive the executions used.

Using NuSMV we also mechanically computed the exact worst-case time complexity for $9 \leq n \leq 20$. Interestingly the complexity exactly matches our lower bound. Table 2 shows the concrete figures for this range of n. Based on this finding, we conjecture that our lower bound is the exact worst-case time complexity when $n \geq 9$. If our conjecture is true, then it is also true under a distributed scheduler, because any single step under a distributed scheduler can be simulated by a sequence of steps under a centralized scheduler [5], [9].

References

[1] E.W. Dijkstra, "Self-stabilizing systems in spite of distributed control," Commun. ACM, vol.17, no.11, pp.643–644, Nov. 1974.

[2] V. Chernoy, M. Shalom, and S. Zaks, "A self-stabilizing algorithm with tight bounds for mutual exclusion on a ring," Proc. 22nd Int'l Symp. on Distributed Computing (DISC), Lecture Notes in Computer Science, vol.5218, pp.63–77, Springer, Sept. 2008.

[3] E.W. Dijkstra, "A belated proof of self-stabilization," Distributed Computing, vol.1, no.1, pp.5–6, Jan. 1986.

[4] M. Kimoto, T. Tsuchiya, and T. Kikuno, "The lower bound on the stabilization time of Dijkstra's three state mutual exclusion algorithm," IEICE Technical Report, COMP2008-7, April 2008.

[5] J. Burns, M. Gouda, and R. Miller, "On relaxing interleaving assumptions," Proc. MCC Workshop Self-Stabilizing Systems, MCC Technical Report, no.STP-379-89, 1989.

[6] A. Cimatti, E.M. Clarke, F. Giunchiglia, and M. Roveri, "NuSMV: A new symbolic model checker," Software Tools for Technology Transfer, vol.2, no.4, pp.410–425, 2000.

[7] T. Tsuchiya, S. Nagano, R.B. Paidi, and T. Kikuno, "Symbolic model checking for self-stabilizing algorithms," IEEE Trans. Parallel Distrib. Syst., vol.12, no.1, pp.81–95, Jan. 2001.

[8] T. Tsuchiya, Y. Tokuda, and T. Kikuno, "Computing the stabilization time of self-stabilizing systems," IEICE Trans. Fundamentals, vol.E83-A, no.11, pp.2245–2252, Nov. 2000.

[9] V. Chernoy, M. Shalom, and S. Zaks, "On the performance of Dijkstra's third self-stabilizing algorithm for mutual exclusion," 9th International Symposium on Stabilization, Safety, and Security of Distributed Systems (SSS), Lecture Notes in Computer Science, vol.4838, pp.114–123, Springer, Paris, Nov. 2007.