2007

The solution of traffic signal timing by using traffic intensity estimation and fuzzy logic

Paothai Vonglao
Edith Cowan University

THE SOLUTION OF TRAFFIC SIGNAL TIMING BY USING TRAFFIC INTENSITY ESTIMATION AND FUZZY LOGIC

By

Paothai Vonglao

A dissertation submitted to the School of Engineering and Mathematics, Faculty of Computing, Health & Science of Edith Cowan University, Western Australia,

In Partial Fulfilment of the Requirements for the degree

DOCTOR OF PHILOSOPHY

Faculty of Computing, Health and Science
Edith Cowan University

July 2007

USE OF THESIS

The Use of Thesis statement is not included in this version of the thesis.

ABSTRACT

This study aims at calculating the traffic signal timing that suits traffic intensity at intersections studied in the inner city of Ubon Rachathani Provice, Thailand. The mixed models between maximum likelihood estimation and Bayesian inference are presented to estimate traffic intensity. A queuing system is used to generate the performance of traffic flow. A fuzzy logic system is applied to calculate the optimal length of each phase of the cycle. The fortran language is used to produce the computer program for computation. The algorithm of the computer programming is based on EM algorithm, Markov Chain Monte Carlo algorithm, queuing generation and fuzzy logic. The output of traffic signal timing from the fuzzy controller are longer than the traffic signal timing from the conventional controller. Cost function is used to evaluate the efficiency of the traffic controller. The result of the evaluation shows that fuzzy controller is more efficient than a conventional controller.

DECLARATION

I certify that this thesis does not, to the best of my knowledge and belief:

(i) incorporate without acknowledgment any material previously submitted for a degree or diploma in any institution of higher education.

(ii) contain any material previously published or written by another person except where due reference is made in the text; or

(iii) contain any defamatory material.

I also grant permission for the Library at Edith Cowan University to make duplicate copies of my thesis as required.

Signature: Date 9 July 2007 :

Paothai Vonglao

ACKNOWLEDGEMENTS

First and foremost, I would like to express my gratitude to my supervisor,

Dr. David Mcdougall and associate supervisor, Dr Jim Cross for their considerable support and encouragement. Without their kind assistance and guidance, this study would have never been accomplished. My deepest thanks go to Rajabhat Ubonratchathani University, for providing a partial scholarship for my study. Special thanks go to Dr. Chompunoot Morachat for her proof reading and for correcting my English and to Ajarn Chamlong Wongprasert for his kind assistance on computer operation. Also I would like to thank my colleagues for their kind assistance and contributions and encouragement. Personally, I thank my family, my wife and my son for their love, patience, kind understanding, and cooperation which contribute most to the completion of my dissertation.

TABLE OF CONTENTS

Tables

Figures

Chapter 1
Introduction

1.1 The background of traffic signal timing

The changes to social structures resulting from technology in both positive and negative ways have brought us advantages and disadvantages at the same time. The negative aspect of the changes introduces an important problem, the traffic problem. The seriousness of the problem depends on the size of the community. That is, the larger the city, the more serious and complex a problem we will face. Moreover, the longer we let the problem go unsolved for longer and longer, the problem will become more and more serious.

Part of the traffic problem is congestion at intersections that is caused by various factors. One important factor that impacts on traffic at intersections is the length of each phase in the cycle of the traffic signal. It may not be appropriate and may not be suitable for traffic pattern parameters such as volume of vehicles, queue length, delay, speed and so on. It is a worldwide problem. Rice Square in Worcester is one example (Kotsopoulos, 1999). Moreover, there is poor timing on traffic signals in cities such as Atlanta (Ledford, 2002). On the other hand if the traffic flow is saturated, the optimal signal length based on Webster's formulation is not available (Lan, 2004). Finally the example of the congestion at intersections in Bangkok is well known. The modern Bus Rapid Transit (BRT) alone cannot solve the traffic problems in Bangkok (Jaiimsin,2004). To try to improve the situation, road transport will be integrated with other modes of transport, including the conventional bus network, skytrain, subway, rail and ferries in 2006, according to the transit plan.

As mentioned above, one reason for traffic jams is that traffic signal timing is often not suitable for traffic control at the intersection in real time. So the concerned traffic office needs to optimize traffic signal timing to solve the traffic congestion at intersections. Engineers behind the federally funded Traffic Signal System Improvement Program in Denver (Hsiao-Ching & Denver, 1998) have worked over the past 10 years to ease metro-area traffic congestion by coordinating and adjusting the timing of traffic signals on major streets. There are many papers that propose methods to improve traffic signal timing. All of the methods use a similar process, based on observed traffic data input at intersections, such as volume, pattern of traffic, number of cars going straight or turning right, delay, queue length, speed, density, and so on. The data input is used according to the individual method. The results from the method can be used to control traffic at intersections. The relevant papers are considered below.

Traffic Adaptive Control is a useful method (Jaqannathan & Khan, 2001) that could optimize traffic signal timing to fit with traffic volume. The results from the method consist of three components, cycle length, phase length and offset that can be used to efficiently control traffic. The software that is used to find traffic signal timing is SYNCHRO. It is composed of capacity analysis, coordination, and actuated signal modeling. This software provides a detailed summary report on capacity, level of service, volumes, timing, queue length, blocking problems, delay, fuel consumption and emission level.

Dynamic Intersection Signal Control Optimization is an another method (Lo & Chow, 2004) that can be used to control traffic flow at intersections. It is based on the entire fundamental diagram of traffic flow. The input data consists of time-variant traffic patterns and the method derives a dynamic timing plan, useful to decrease delays at intersections.

Traffic Signal Retiming is another process that can optimize traffic signal length at intersections (Sunkari, 2004). This includes development of new signal timing parameters, phasing sequence and traffic control strategy improvements.

In addition to the three papers above, many authors propose methodology to improve traffic signal timing and traffic control at intersections. Lan (2004) proposes a new formulation to find the optimal traffic signal length when traffic flows become saturated. Leonard et al. (1998) suggest traffic signal timing based on five basic signal timing policies: minimizing delay, minimizing stopping, minimizing fuel consumption, maximizing coordination, and baseline.

Mathematical methods have often been used. Schutter (2002) looks at the mathematical programming problem of designing optimal switching schemes and an optimal switching sequence for signal controlled intersections. The results decrease queue and waiting time. Yi, Xin & Zhao (2001) implement a general speed-density relationship in a dynamic queue length estimation model, leading to the development of a general mathematical formulation for intersection queue length studies that can be used to control traffic at intersections. On the other hand Lee, Messer, Oh, & Lee (2004) propose a rule to control traffic at intersections, based on allowing the green light to show if any individual vehicle, pedestrian or cyclist queue, measured at regular intervals and averaged over the peak hour, is at least four, or if the sum of the individual vehicle, pedestrian and cyclist queues, measured anywhere within the intersection, exceeds six. And finally a similar idea is proposed by Rakha & Zhang (2004) as evaluation of Transit Signal Priority(TSP). In general TSP provides benefits to transit vehicles that receive priority, but TSP has a marginal system wide impact for low traffic demand. On the other hand the system wide impact of TSP is directly proportional to the frequency of transit vehicles.

All of the above show that there are global concerns about traffic signal timing, and the output of the studies are useful in controlling traffic at intersections. Although there are many methods to improve traffic signal timing as previously mentioned, the lack of coordination could result in inefficient traffic flow. (Hsiao-Ching & Denver, 1998)

1.2 The background of the traffic problem in Ubon Rachathani

Ubon Rachathani, as the big city in the northeast of Thailand, has the 5th rank in area and the 4th in population in Thailand. It is now one of the traffic jam problem cities as well. The problem is not as serious as in Bangkok. However, if there is no attempt to solve the traffic problem, Ubon Rachathani will be soon face the same problems as Bangkok. The traffic jam problem in Ubon Rachthani is caused by the increasing number of cars (Engineering Faculty of Songkhla Nakarin University: 1999) and the lack of observance of traffic regulations. Parking at prohibited spots, double parking and other infringements are common. In addition, part of the traffic problem is that traffic congestion at intersections is caused by the design of traffic signals.
Control at intersections is pre-timed or fixed time, and the length of each phase in cycles is not suitable for the traffic intensity.(Ubon Rachathani Municipality, 2001)

1.3 The traffic control and traffic signal timing in Ubon Rachathani Municipality

In the Ubon Ratchathani Municipality, there are 48 intersections and 5 crossroads with signals. The traffic signals at each intersection are controlled in isolation by setting the pre-timed or fixed-time cycle. However a traffic policeman can adjust the timing to suit traffic intensity. On the other hand the office that is responsible for traffic control in the municipality has set the length of each phase in the cycle, or the length of green light in the cycles, to control traffic at intersections as follow: (Ubon Rachathani Municipality, 2004).

1) The traffic signal timing at intersections of the main road, Chayangkoon Road, that bears heavy traffic in the rush hour:

1.1) From 05.30 – 06.30 in the morning:

The length of the green light (phase length) on the main road is 20 seconds.

The length of the green light (phase length) on the sub road is 15 seconds.

1.2) From 0 6.30 – 09.30 in the morning

The length of the green light (phase length) on the main road is 25 seconds.

The length of the green light (phase length) on the sub road is 20 seconds.

1.3) From 0 9.30 – 15.30 in the afternoon:

The length of the green light (phase length) on the main road is 20 seconds.

The length of the green light (phase length) on the sub road is 15 seconds.

1.4) From 15.30 – 17.30 in the afternoon

The length of the green light (phase length) on the main road is 25 seconds.

The length of the green light (phase length) on the sub road is 20 seconds.

1.5) From 17.30 – 23.00 in the evening

The length of the green light (phase length) on the main road is 20 seconds.

The length of the green light (phase length) on the sub road is 15 seconds.

1.6) From 23.00 – 05.30 in the morning

The length of the green light (phase length) on the main road is 20 seconds.

The length of the green light (phase length) on the sub road is 15 seconds.

2) The traffic signal timing at intersections on the subroads, except Chayangkoon Road.

   2.1) From 05.30 – 22.00 in the afternoon

   The length of the green light (phase length) on the main road is 20 seconds.

   The length of the green light (phase length) on the sub road is 15 seconds.

   2.2) From 22.00 – 05.30 in the morning:

   The amber blink is provided on the main road.

   The red blink is provided on the sub road.

1.4 The background for estimation of traffic signal timing

Traffic signal controllers at intersections are divided into four types, based on their potential, as follows:

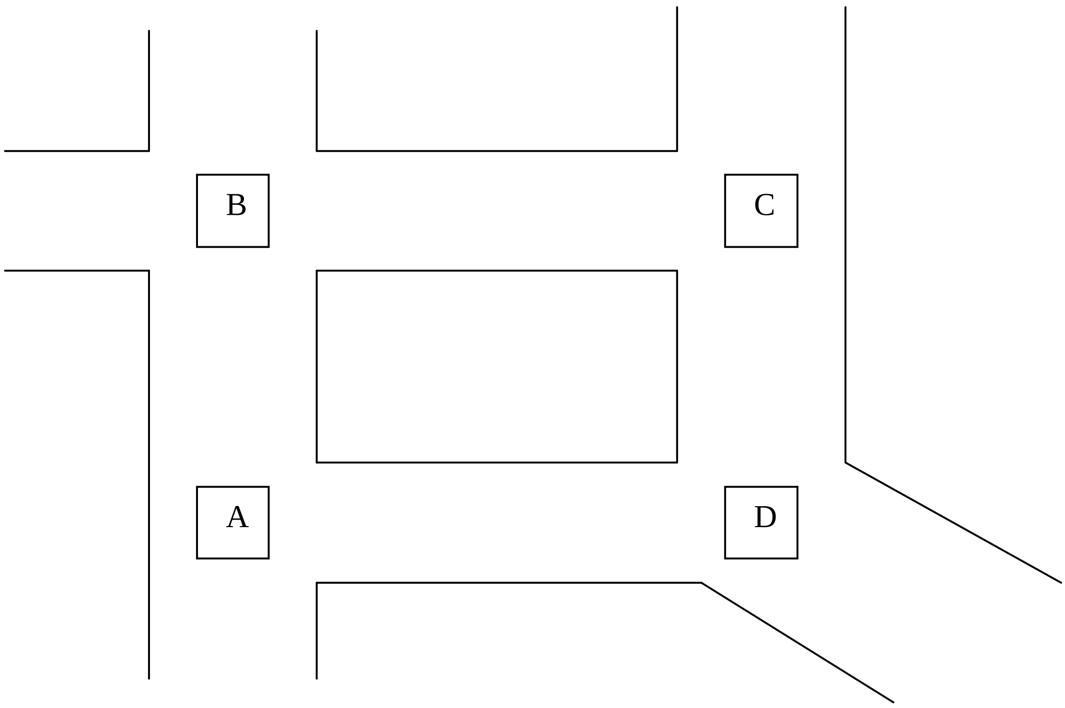
1) Pre-timed or fixed time traffic signal control. They offer fixed length for each phase of a cycle.
2) Semi-actuated traffic signals control. They offer flexible length for each phase in cycles, to match the number of cars from the sub road by using a detector. Whenever there are lots of cars on the main road, the controller will let the cars run, while the cars in the sub road have to wait until the numbers of waiting cars reach a specified number and then they will be allowed to go.
3) Fully-actuated traffic signals control, These allow all vehicles from any direction to pass the intersection by choosing a cycle length that is appropriate for the number of cars, by using a detector.
4) Volume density traffic signals control. They count the number of cars by using the detector and then the information is sent to the central computer in order to control the traffic flow of the whole traffic network. Moreover, the control gives priority to emergency vehicles, such as ambulances.

However, Ubon Rachathani Municipality still uses the old technology of pre-timed traffic signal control to control traffic flow at intersections. Based on the limitation of the control, one way to improve the efficiency of traffic signal control is to improve traffic signal timing identification in each phase of the cycle.

This study proposes an alternative method to calculate suitable lengths for each phase in the cycle for a given traffic intensity. The statistical and mathematical methodology is used to identify the optimal length of each phase, to decrease delay and queue of traffic flow at the intersections studied.

1.5 The actual intersections studied

This study focuses on the main traffic network in the inner city of Ubon Rachathani that is composed of four intersections: Uboncharearnsri Intersection, Clock Hall Intersection, Chonlaprathan Intersection and Airport Intersection. The study will be limited to part of the rush hour, namely 8.00-8.30 am. A diagram of the traffic network is given below:



A : Uboncharearnsri Intersection   B : Airport Intersecrtion

C : Chonlaprathan Intersection   D : Clock Hall Intersection

Figure 1.1 Diagram of traffic network studied

1.6 The outcomes and the organisation of the thesis

The outcomes of the thesis will give advice to traffic policeman to adjust the suitable signal time for controlling the traffic at the studied intersection. The organisation of the thesis is composed of six principal components as follows:

1. Introduction
2. Discussing the theory background
3. Research methodology
4. Input and analysis
5. Result of the study
6. Conclusion and discussion

1.7 Objectives

To calculate the optimal traffic signal timing during the given period (08.00-08.30 am)around intersections in Uboncharearnsri, Airport, Chonlaprathan, and Clock Hall of Ubon Ratchathani metropolitan area.

1.8 The Expected Outcomes

1) To derive a method to calculate the traffic signal timing at targeted intersections during rush hour.
2) To get to know the traffic signal timing that is relevant to the number of vehicles at the targeted intersections.

The statistical estimation, maximum likelihood and Bayesian inferrence, and the fuzzy logic system were used to find the expected outcomes.

Chapter 2
Background to Research

2.1 Fuzzy logic systems

2.1.1 General background

Fuzzy logic was first developed in 1965 by Lotfi A. Zadeh, Professor Emeritus, Computer Science Division, University of California-Berkeley. Fuzzy logic uses three primary elements: fuzzy sets, the membership function and production rules. Applications of fuzzy logic occur in three primary categories: consumer products, industrial/commercial systems and decision support systems.(Glenn, 1994)

David (1992) describes the components of a fuzzy controller. Toshinori & Yashvant (1994) present a fuzzy system composed of fuzzy set, logic, algorithms, and control. Implementation of the fuzzy control is suitable for a problem that is described in approximate form and that requires a complicated mathematical model to explain the behavior of the model. A fuzzy system can be applied to various subjects.

2.1.2 Applications of fuzzy systems

In Soud & Kazemian (2004), Usage Parameter Control (UPC) is the

process that provides support for quality of service across a heterogeneous system. They propose a novel form of the Usage Parameter Control(UPC) by using a Fuzzy Logic Controller (FLC) to measure the rate of individual network flow to actively manage link utilization. The results obtained significantly improve upon the best service of the system.

Abdel-Aty & Abdelwahab (2004) study the effectiveness of methods to predict driver injury severity as a result of a crash. Fuzzy adaptive resonance is one method. The study shows a fuzzy adaptive resonance has an accuracy of 70.6 percent.

Masalonis & Parasuraman (2003) apply fuzzy signal detection techniques, which combine fuzzy logic and conventional signal detection theory, to empirical data. The object of the application is to detect aircraft incidents in air traffic control. The results illustrate the potential of fuzzy signal detection theory to provide a more complete picture of performance in aircraft incident detection.

Adeli & Jiang (2003) study zone capacity, which cannot be described by any mathematical function because it is a complicated function of a large number of interacting variables. They propose a novel adaptive neuro-fuzzy logic model for estimation of the freeway work zone capacity. Comparisons with two empirical equations demonstrate that the new model in general provides a more accurate estimate of the work zone capacity, especially when the data for factors impacting the work zone capacity are only partially available.

Kikuchi & Tanaka (2003) use a fuzzy rule based on a simulation process to examine how the presence of vehicles equipped with an Adaptive Cruise Control System (ACCS) affects stability and safety of a flow consisting of both ACCS and non-ACCS vehicles.

Ramasamy & Selladurai (2004) propose the use of fuzzy logic in quality function deployment. The deployment is a proven tool used to develop process and product, and translates the voice of a customer into engineering characteristics and then prioritises the characteristic based on a customer’s requirements. Fuzzy logic is useful to define the relationship between the characteristic and customer attributes.

Kirawanich & O'Connell (2004) describe a system that uses fuzzy logic to control the semiconductor switches in the switch-mode of active power line conditioners. The simulations and measurements show that the system can significantly improve line current total harmonic distortion and power factor during both steady-state and transient operating conditions.

Fisher (2004) mentions the importance of fuzzy logic used to improve the potential of computers to think like fuzzy-thinking people, instead of like purely logical machines.

In addition the article, claims fuzzy logic has been used to control subway trains, elevators, washing machines, microwave ovens, and cars. Another really important use for fuzzy logic is in robots.

Stewart, Cheraghi, & Malzahn (2004) use fuzzy Bayesian methodology in a fuzzy defect avoidance system. The system is used to reduce the amount of scrap and rework activity in a product process in industry. This method can be used to provide continuous opportunities for defect avoidance.

Harb & Smadi (2004) present the idea of using the fuzzy logic concept for controlling chaotic behavior in systems. The fuzzy control is useful because there is no mathematical model available for the system and the control can produce nonlinear control that can be developed empirically.

Beynon, Pee, & Tang (2004) point out that fuzzy set theory has evolved into a valuable addition to traditional techniques, such as regression and decision tree models, for decision analysis conducted under conditions of vagueness and ambiguity. They apply a fuzzy decision tree approach to a problem involving typical accounting data. The results show that fuzzy logic enables a decision-maker to gain additional insights into the relationship between firm characteristics and audit fees, through human subjective judgment expressed in linguistic terms.

Zhang & Tam (2004) present an incorporation of discrete-event simulation and fuzzy logic to model uncertainties in a construction process. The fuzzy set is used to model the uncertain demand in linguistic terms. The fuzzy rule base is built to control the activities. The activity duration is generated through the fuzzy logic reasoning. Through the application of the fuzzy construction simulation system, an illustrative example is presented to demonstrate the effect of considering these uncertainties on the productivity.

Cho & Yi (2004) propose the use of a fuzzy logic controller in vehicle dynamics to control the vehicle trajectory when the driver suddenly depresses the brake pedal under critical conditions. The function of the fuzzy controller is to control each brake and works to compensate for the trajectory error on the split - road conditions to maintain the desired trajectory.

2.1.3 Applications of fuzzy logic for traffic control.

The previous section illustrates the wide use of fuzzy logic for control and decision in any system. This section concentrate on the use of fuzzy logic for traffic control. There are seven relevant papers.

Zhenyang (2004) discusses a model to control traffic flow at intersections by using fuzzy logic control. The model is designed with a four-level fuzzy logic controller to estimate relative traffic intensities in competing approaches to intersections. The estimator is then used to determine whether a leading or lagging signal phase should be selected or terminated for each approach. On the other hand the researcher creates a dynamic traffic signal left-turn phase control system, and implements the four-level fuzzy logic control model to optimize signal operations at intersections. The resulting system is on efficient tool for reducing intersection traffic delay.

Ande (1996) creates a model to control traffic flow at intersections by using fuzzy logic. The model is adaptive, using actual traffic intensities by means of standard input traffic flow parameters, which are measured by a loop detector. The results of the study show the model is more efficient than the conventional traffic controls such as pre-timed controllers or even semi-actuated controllers based on heavy traffic conditions.

Enid (1999) designs a fuzzy logic based traffic controller for an arterial street. The controller can adjust the timing parameters on-line based on the current traffic conditions. The strategy of fuzzy logic based control consists of a local controller at each intersection and a global controller that communicates with all the local controllers. The object of these controllers is to optimize traffic signal timing at coordinate intersections. . Simulations showed significant improvements on the average time in queue, the average queue length, and the average travel time, when compared to coordinated pre-timed and semi-actuated controllers.

Seongho (1994) developed the Advanced Traffic Management Systems (ATMS) to improve traffic signal control at intersections. Fuzzy logic is used for a real-time traffic adaptive signal control scheme in the systems. The results of the study show that the ATMS framework will lead to real-time adjustment of the traffic control signals, resulting in significant reduction in traffic congestion.

Adeli & Karim (2000) presents fuzzy logic in a new multi-paradigm intelligent system approach to solve traffic problems that are disrupted by traffic incidents. The approach uses advanced signal processing, pattern recognition and classification techniques.

Lee, Krammes&Yen (1998) use a fuzzy logic based incident detection algorithm for a traffic network. The model is used to detect traffic incidents, any problems on the street surface that require the attention of an operator or result in an operator formulating a response, (such as lane blockages). The algorithm feeds an incident report such as the time, location, and severity of the incident to the system's optimization manager, which uses that information to determine the appropriate traffic signal control strategy.

Cabrera & Ivan (2000) create a methodology to design traffic signal controls based on fuzzy logic control. There are many applications that use fuzzy logic to control traffic flow at intersections but there is no uniform design procedure. So they propose the design to help people, not familiar with fuzzy logic control, to apply the method for traffic signal control. The designed fuzzy controller uses existing traffic detectors to measure the number of vehicles at the intersection and decides how to change the traffic signals in order to minimize the average delay of vehicles. Simulation results show that traffic controllers developed with the proposed methodology reduce a average delay of vehicles at intersections compared with conventional traffic control strategies.

2.1.4 The concept of a fuzzy logic system

Wang (1994) presents the common concept of a fuzzy logic system. The system consists of fuzzy concepts and fuzzy logic. The fuzzy concepts involve fuzzy sets , linguistic variables and so on. The fuzzy logic is the process that is used to infer the parameter of a system based on incorporated numerical information and an expert's knowledge. For most engineering systems, there are two important information sources: a sensor which provides numerical measurements of variables, and human experts who provide linguistic instructions and descriptions about the system. The information from sensors is called numerical information and the information from human experts is called linguistic information. To apply information to a variety of control, signal processing, and communication problems and to analyse their performance, it is necessary to develop a collection of methods which can effectively combine numerical and linguistic information into the engineering systems. An adaptive fuzzy logic system is such a tool. The system is defined as a fuzzy logic system that is constructed from a set of fuzzy IF-THEN rules using fuzzy logic principles, and a training algorithm that adjusts the parameters of the fuzzy logic system based on numerical information. In other words adaptive fuzzy systems can be viewed as fuzzy logic systems whose rules are automatically generated through training. The strategy of an adaptive fuzzy logic system for combining numerical and linguistic information is based on the construction of an initial fuzzy logic system by using linguistic information. Then the parameters of the system are adjusted based on numerical information. An additional strategy is to use numerical information and linguistic information to construct two separate fuzzy logic systems. Then the final fuzzy logic system is the average of the two systems.

Definition 1 Linguistic variable (intuitive) : A linguistic variable is a variable that can take either a word in natural language (for example small, fast and so on) or a number as its values.

Definition 2 Linguistic variable (formal): A Linguistic variable is characterized by a quintuple $(x, T(x), U, G, S)$ in which x is the name of variable; $T(x)$ is the term set of x, that is, the set of names of linguistic values of x with each value being a fuzzy set defined on U; G is a syntactic rule for generating the name of values of x; and S is a semantic rule for associating each value with its meaning.

Definition 3 Fuzzy set: Let U be a collection of objects, for example, $U = R^n$, usually called the universe of discourse. A fuzzy set F in U is characterized by a membership function $m_F : U \rightarrow [0,1]$, with $m_F(u)$ representing the grade of membership of $u \in U$ in the fuzzy set F. A fuzzy set may viewed as a generalization of the concept of an ordinary set whose membership function only takes two values $\{0,1\}$.

The most, popular fuzzy logic systems may be classified into three types: pure fuzzy logic systems, Takagi and Sugeno's fuzzy systems, and fuzzy logic systems with fuzzifier and defuzzifier. These are briefly described in the next three subsections.

1) Pure fuzzy logic system

The pure fuzzy logic system is conceptualised as two components, a fuzzy rule base and a fuzzy inference engine. The fuzzy rule base consists of a collection of fuzzy IF-THEN rules, and the fuzzy inference engine is used to determine a mapping from a fuzzy set in the input universe of discourse $U \subset R^n$ to fuzzy sets in the output universe of discourse $V \subset R$, which is based on fuzzy logic principles. The fuzzy rule base is composed of M rules, of the following form:

$$L^{(j)} \ : \ \text{IF } x_1 \text{ is } F_1^j \ \text{ and } \ldots \text{ and } \ x_n \text{ is } F_n^j \ \text{ THEN } \ y \ \text{ is } G^j$$

Here F_i^j and G^j are fuzzy sets, $\underline{x} = (x_1, x_2, \ldots, x_n) \in U$ and $y \in V$ are input and output linguistic variables, respectively, and $j = 1, 2, \ldots, M$. $i = 1, 2, \ldots, n$ .These fuzzy IF-THEN rules provide a convenient framework to incorporate a human expert's knowledge. In other words each fuzzy rule, $R^{(j)}$ is fuzzy set $F_1^j \times F_2^j \times \ldots \times F_n^j \rightarrow G^j$ in the product space $U \times V$. The most commonly used fuzzy logic principle in fuzzy inference engines is the so-called sup-star composition. Specifically, let A' be an arbitrary fuzzy set in U ; that is, A' is the input to the pure fuzzy logic system. Then the output determined by each fuzzy rule, $R^{(j)}$, is a fuzzy set $A' \circ R^{(j)}$ in V whose membership function is

$$\mu_{A' \circ R^{(j)}}(y) = \sup_{\underline{x} \in U} [\mu_{A'}(\underline{x}) * \mu_{F_1^j \times \ldots \times F_n^j \rightarrow G^j}(\underline{x}, y)]$$

where the "$*$" operator is "min" or "product" and μ_A represents the membership function of the fuzzy set A. The final output of the pure fuzzy logic system is the fuzzy set $A' \circ (R^{(1)}, R^{(2)}, \ldots, R^{(M)})$ in V which has membership function:

$$\mu_{A' \circ (R^{(1)}, R^{(2)}, \ldots, R^{(M)})}(y) = Max[\mu_{A^1 \circ R^{(1)}}(y), \mu_{A' \circ R^{(2)}} \ldots, \mu_{A' \circ R^{(M)}}(y)]$$

2) Takagi and Sugeno's fuzzy system

Takagi and Sugeno use the following fuzzy rule :

$$L^{(j)} : \text{ IF } x_1 \text{ is } F_1^j \ \text{ and } \ldots \text{ and } x_n \text{ is } F_n^j \ \text{ THEN } \ y^j = c_0^j + c_1^j x_1 + c_2^j x_2 \ldots + c_n^j x_n$$

Here F_i^j are fuzzy sets; c_i^j are real-value of parameters; y^j is the system output due to rule $L^{(j)}$, and $j = 1, 2, \ldots, M$. For a real-value input vector $\underline{x} = (x_1, x_2, \ldots, x_n)$, the output $y(\underline{x})$ of Takagi and Sugeno's fuzzy system is a weighted average of the $y^{j,}s$:

$$y(\underline{x}) = \frac{\sum_{j=1}^{M} w^j y^j}{\sum_{j=1}^{M} w^j}$$

where the weight w^j of rule $L^{(j)}$ for the input is calculated as

$$w^j = \prod_{i=1}^{n} m_{F_i^l}(x_i)$$

3) Fuzzy logic systems with Fuzzifier and Defuzzifier

A fuzzy logic system with fuzzifier and defuzzifier is a pure fuzzy logic system which adds a fuzzifier to the input and a defuzzifier to the output. The fuzzifier maps crisp points(numeric values) in U to fuzzy sets in U , and the defuzzifier maps fuzzy sets in V to crisp points(numeric values) in V . The fuzzy inference engines are the same as those in pure fuzzy logic systems. Such a fuzzy logic system consists of four components.

3.1) Fuzzifier

The fuzzifier performs a mapping from a crisp point $\underline{x} = (x_1, x_2, \mathbf{K}, x_n)$ into a fuzzy set $A^/$ in U . The mapping is commonly called a membership function. A membership function is a curve that defines how each crisp point in the input space is mapped to a membership value between 0 and 1. The membership function is usually one of the following:

1) Singleton fuzzifier: $A^/$ is a fuzzy singleton with support $\underline{x}$, that is, $m_{A^/}(\underline{x^/}) = 1$ for $\underline{x}^/ = \underline{x}$ and $m_{A^1}(\underline{x}^/) = 0$ for all other $\underline{x}^/ \in U$ with $\underline{x}^/ \neq \underline{x}$

2) Nonsingleton fuzzifier: $m_{A^/}(\underline{x}) = 1$ and $m_{A^/}(\underline{x}^/)$ decreases from 1 as $\underline{x}^/$ moves away from $\underline{x}$. The nonsingleton fuzzifier may be useful if the inputs are corrupted by noise. The function itself can be an arbitrary curve whose shape suits the expert from the point of view of simplicity, convenience, speed, and efficiency.

The fuzzy logic toolbox includes 11 built-in membership function types. These 11 functions are, in turn, built from several basic functions: piecewise linear functions, the

Gaussian distribution function, the sigmoid curve; quadratic and cubic polynomial curves. The most commonly used functional forms are triangular, trapezoid and Gaussian which are ways to determine the parameters in $m_{A'}(\underline{x}')$ based on measured data. The simplest membership functions are formed using straight lines. Of these, the simplest is the triangular membership function; it is nothing more than a collection of three points forming a triangle. The trapezoidal membership function has a flat top and really is just a truncated triangular curve. These straight line membership functions have the advantage of simplicity. Two membership functions can be built on the Gaussian distribution curve: a simple Gaussian curve and a two-sided composite of two different Gaussian curves. Gaussian membership function have the advantage of being smooth and nonzero at all point.

Figure 3.1 shows the membership functions of three fuzzy sets, namely, “slow”, “medium”, and “fast” for the linguistic variable “the speed of the car”. In this example, the universe of discourse is all possible speeds of the car; that is $U = [0, V_{\max}]$, where $V_{\max}$ is the maximum speed of the car.

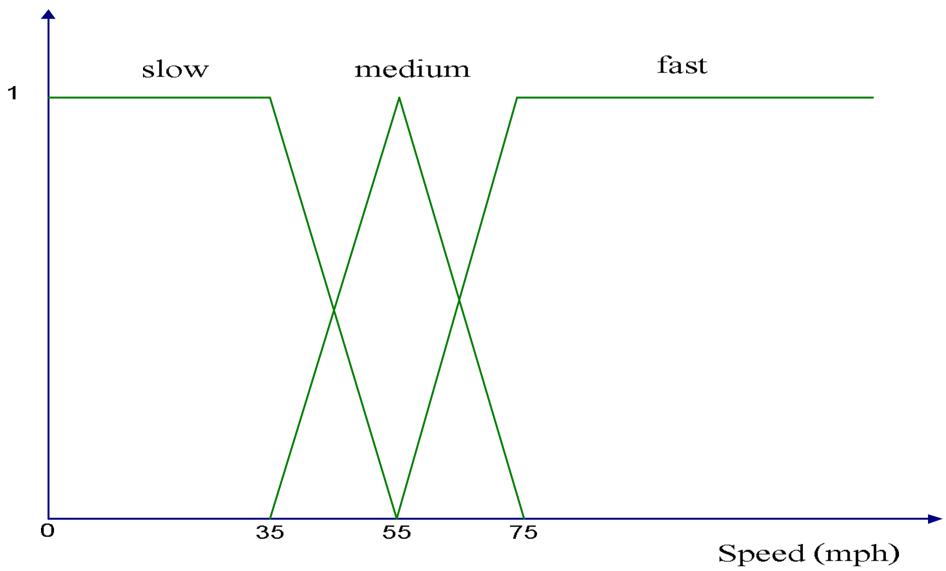


Figure 2.1 Membership functions of three fuzzy sets, namely, “slow”, “medium”, and “fast” for the speed of the car (Wang, 1994, p. 10)

3.2) Fuzzy rule base

A fuzzy rule base consists of a collection of fuzzy IF-THEN rules in the following form:

$$L^{(j)} \ : \text{ IF } \ x_1 \text{ is } F_1^j \text{ and } \ldots \text{ and } x_n \text{ is } F_n^j \text{ , THEN } \ y \text{ is } G^j$$

Here F_i^j and G^j are fuzzy sets in $U_i \subset R$ and $V \subset R$, respectively, and $\underline{x} = (x_1, x_2, ..., x_n)$, $\underline{x} \in U_1 \times U_2 \times \ldots \times U_n$ and $y \in V$ are linguistic variables. Let n be the number of fuzzy set F_i^j; that is, $i = 1,2,\ldots,n$ and M be the number of fuzzy IF-THEN rules in the rule base; that is, $j = 1,2,\ldots,M$. $\underline{x}$ and y are the input and output of the fuzzy logic system, respectively. The fuzzy rule is derived from asking human experts and using training algorithms based on measured data. The membership functions for the fuzzy sets are determined in two ways depending upon where the rules come from. If the rules are provided by human experts, then the membership functions should be specified by the experts because these functions are an integrated part of the expert's knowledge. If the rules are determined by numerical data, then the first task is to determine the functional forms for $\mu_{F_i'}$ and $\mu_{G'}$ .The most commonly used functional forms are Gaussian, triangular, and trapezoid.

3.3) Fuzzy inference engine

In a fuzzy inference engine, fuzzy logic principles are used to combine the fuzzy IF-THEN rules in the fuzzy rule base in a mapping from fuzzy sets in $U = U_1 \times U_2 \times \ldots \times U_n$ to a fuzzy set in V . The fuzzy IF-THEN rule can be interpreted in a number of ways. For simplicity, we denote $F_1^j \times F_2^j \ldots \times F_n^j = A$ and $G' = B$, and the rule is denoted by $A \to B$. Some commonly used interpretations for the fuzzy IF-THEN rule are as follows:

1) Mini-operation rule of fuzzy implication:

$$\mu_{A\to B}(\underline{x}, y) = \min\{\mu_A(\underline{x}), \mu_B(y)\}$$

2) Product-operation rule of fuzzy implication:

$$\mu_{A\to B}(\underline{x}, y) = \mu_A(\underline{x})\mu_B(y)$$

3) Arithmetic rule of fuzzy implication:

$$\mu_{A\to B}(\underline{x}, y) = \min\{1, 1 - \mu_A(\underline{x}) + \mu_B(y)\}$$

4) Maxmin rule of fuzzy implication:

$$m_{A\to B}(\underline{x},y)=\max\{\min[m_A(\underline{x}),m_B(y)],1-m_A(\underline{x})\}$$

5) Boolean rule of fuzzy implication:

$$m_{A\to B}(\underline{x},y)=\max\{1-m_A(\underline{x}),m_B(y)\}$$

6) Goguen's rule of fuzzy implication:

$$m_{A\to B}(\underline{x},y)=\begin{cases}1 & ,m_A(\underline{x})\le m_B(y)\\ \dfrac{m_B(y)}{m_A(\underline{x})} & ,m_A(\underline{x})>m_B(y)\end{cases}$$

where $m_A(\underline{x})=m_{F_1^j\times\mathbf{K}\times F_n^j}(\underline{x})$ is defined either according to the min-operation rule:

$$m_{F_1^j\times F_2^j\times\mathbf{K}\times F_n^j}(\underline{x})=\min\{m_{F_1^j}(x_1),m_{F_2^j}\,\mathbf{K},m_{F_n^j}(x_n)\}$$

or according to the product-operation rule:

$$m_{F_1^j\times F_2^j\times\mathbf{K}\times F_n^j}(\underline{x})=m_{F_1^j}(x_1)\cdot m_{F_2^j}(x_2)\cdot\mathbf{K}\cdot m_{F_n^j}(x_n)$$

3.4) Defuzzifier

The defuzzifier performs a mapping from fuzzy sets in V to crisp points $y\in V$. There are three common choices of this mapping:

1) The maximum defuzzifier, defined as

$y=\arg\sup_{y\in V}(m_{B'}(y))$;

2) The center average defuzzifier, defined as

$$y=\frac{\sum_{j=1}^{M}y^{-j}(m_{B^j}(y^{-j}))}{\sum_{j=1}^{M}(m_{B^j}(y^{-j}))}$$, y^{-j} is the center of the fuzzy set G^j and

3) The modified center average defuzzifier, defined as

$$y=\frac{\sum_{j=1}^{M}y^{-j}(m_{B^l}(y^{-j})/d^j)}{\sum_{j=1}^{M}(m_{B^l}(y^{-j})/d^j)}$$, d^j is a parameter characterizing the shape of $m_{G^j}(y)$

2.1.5 Method of fuzzy logic control

Kandel & Langholz, (1994) present at least two methods of fuzzy logic control as follows:

1) Min-Max-Gravity method

The fuzzy logic controllers are based on the fuzzy reasoning method called "min-max-gravity method " by Mamdani (1977). The rules used for this method are as follows:

Rule 1: IF x_1 is F_1^1 and $\cdots$ and x_n is F_n^1 THEN y is G^1,

Rule 2: IF x_1 is F_1^2 and $\cdots$ and x_n is F_n^2 THEN y is G^2,

$\vdots$ $\qquad\qquad\qquad\qquad\qquad\qquad$ $\vdots$

Rule M: IF x_1 is F_1^M and $\cdots$ and x_n is F_n^M THEN y is G^M,

Fact : $\qquad x_1', x_2', \ldots, x_n'$

Consequence: $\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad G'$

Here F_i^j is a fuzzy set in set U , $U \subset R$: and G^j is fuzzy set in V , $V \subset R$: and $(x_1', x_2', \ldots, x_n') \in R^n$. $i = 1,2,\ldots n;$ $\quad j = 1,2,\ldots, M$

For simplicity, we let $F_1^j \times F_2^j \ldots \times F_n^j = A$ and $G^j = B$, and each rule is then denoted as $A \to B$. This is defined by

$$m_{A\to B}(x_1, x_2, \ldots x_n) = \min[m_{F_1^j}(x_1), \ldots, m_{F_n^j}(x_n), m_{G^j}(y)]$$

The inference result G_j' infered from the fact of $x_1', x_2', \ldots, x_n'$ and fuzzy rule $A \to B$ is given by

$$m_{G_j'}(y) = \min[m_{F_1^j}(x_1'), m_{F_2^j}\left(x_2'\right)\ldots, m_{F_n^j}(x_n'), m_{G^j}(y)].$$

The final consequence G' is defined by

$$m_{G'}(y) = \max[m_{G_1'}(y), \ldots m_{G_n'}(y)].$$

The representative point y' for the resulting fuzzy set G' is obtained as the center of gravity of G' , that is

$$y' = \frac{\int y m_{G'}(y)dy}{\int m_{G'}(y)dy}$$, where $\int m_{G'}(y)dy$ is the area of fuzzy set G' .

The process of the Min-Max-gravity method can be shown by figure 3.2 as follows:

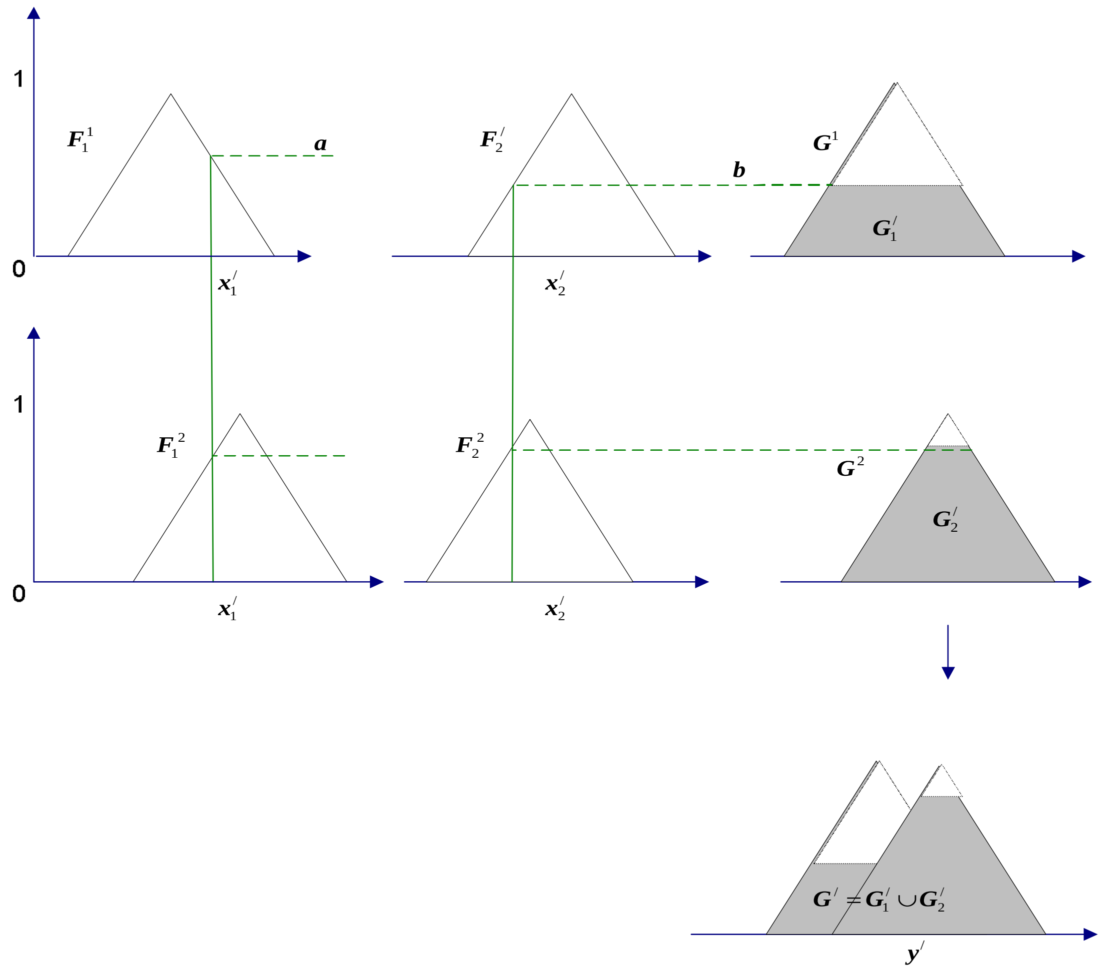


Figure 2.2 Min-Max-Gravity method (Kandel & Langholz, 1994, p.277)

2) Product-sum-gravity method

This section outlines the method of fuzzy reasoning called the product-sum-gravity method, which replaces min by the algebraic product, and max by the sum in the max-min-gravity method. The consequence G' for the product-sum-gravity method is obtained as follows.

Firstly, consider the multiple fuzzy reasoning form.

Rule 1: IF x_1 is F_1^1 and $\wedge$ and x_n is F_n^1 THEN y is G^1,

Rule 2: IF x_1 is F_1^2 and $\wedge$ and x_n is F_n^2 THEN y is G^2,

$\vdots$ $\vdots$

Rule M: IF x_1 is F_1^M and $\wedge$ and x_n is F_n^M THEN y is G^M,

Fact : $x_1', x_2', \ldots, x_n'$

Consequence: G'

The inference result G_j' from the fact $x_1', x_2', \ldots, x_n'$ and the fuzzy rule j is given by:

$$m_{G_j'}(y) = m_{F_1^l}(x_1') \cdot m_{F_2^l}(x_2') \ldots m_{F_n'}(x_n') \cdot m_{G_l}(y).$$

The consequence G' is defined by

$$m_{G'}(y) = m_{G_1'}(y) + \ldots + m_{G_M'}(y).$$

The representative point y' of G' is obtained by using the centre of gravity method. The centre of gravity y' of G' is described below.

Let y_j be the centre of gravity of the inference result G_j' and S_j be the area of G_j' in Figure 3.3 Then y_j is defined as :

$$y_i = \frac{\int y \cdot m_{G_j'}(y)dy}{\int m_{G_j'}(y)dy} = \frac{\int y \cdot m_{G_j'}(y)dy}{S_j}.$$

The centre of gravity y' of the final consequence G_j' is given by

$$y' = \frac{\int y \cdot m_{G'}(y)dy}{\int m_{G'}(y)dy} = \frac{\int y[m_{G_1'}(y) + \ldots + m_{G_M'}(y)]dy}{\int [m_{G_1'}(y) + \ldots + m_{G_M'}(y)]dy} = \frac{\sum_{j=1}^{M} S_j \cdot y_j}{\sum_{j=1}^{M} S_j}$$

The product-sum-gravity method is illustrated in figure 2.3

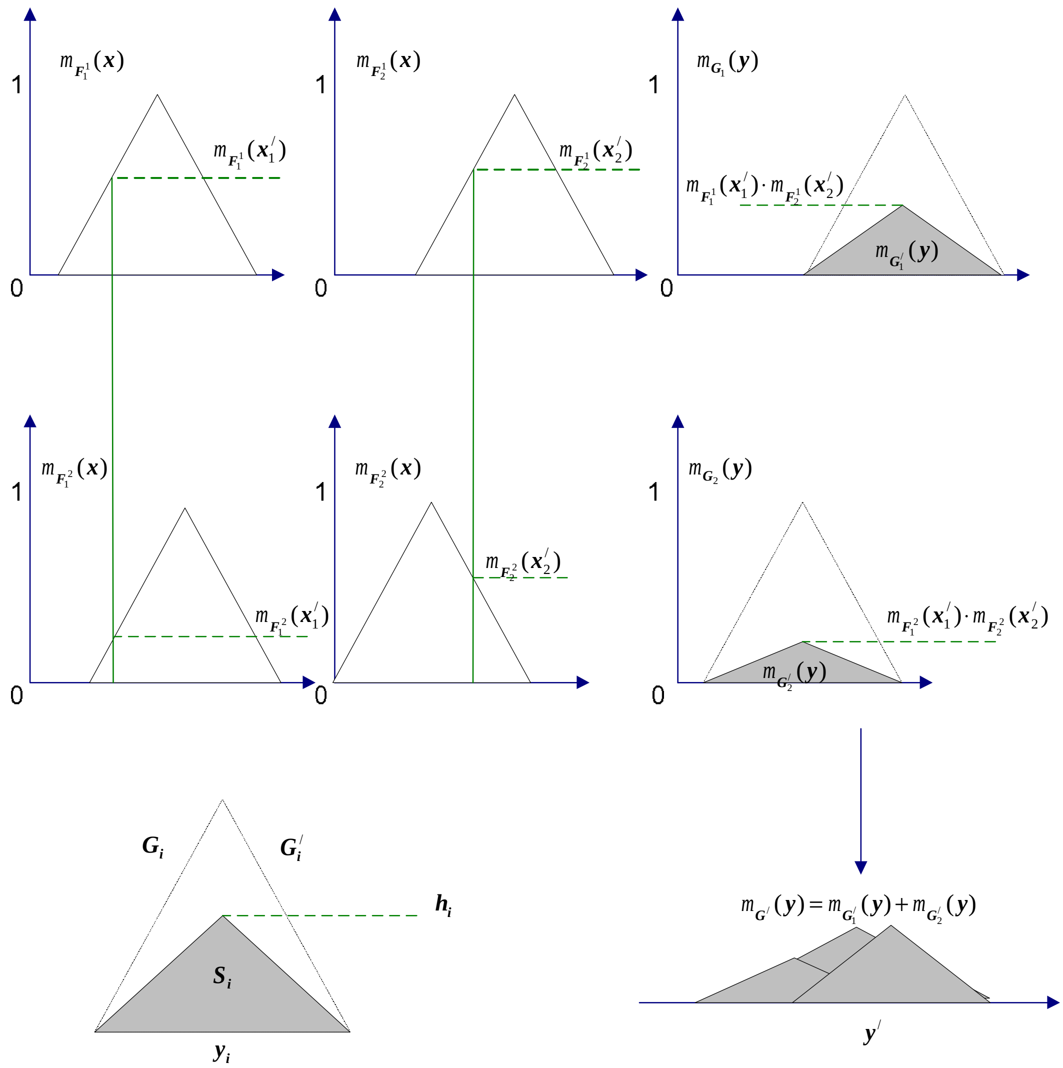


Figure 2.3 Product-sum-gravity method (Kandel & Langholz,1994, p.281-282)

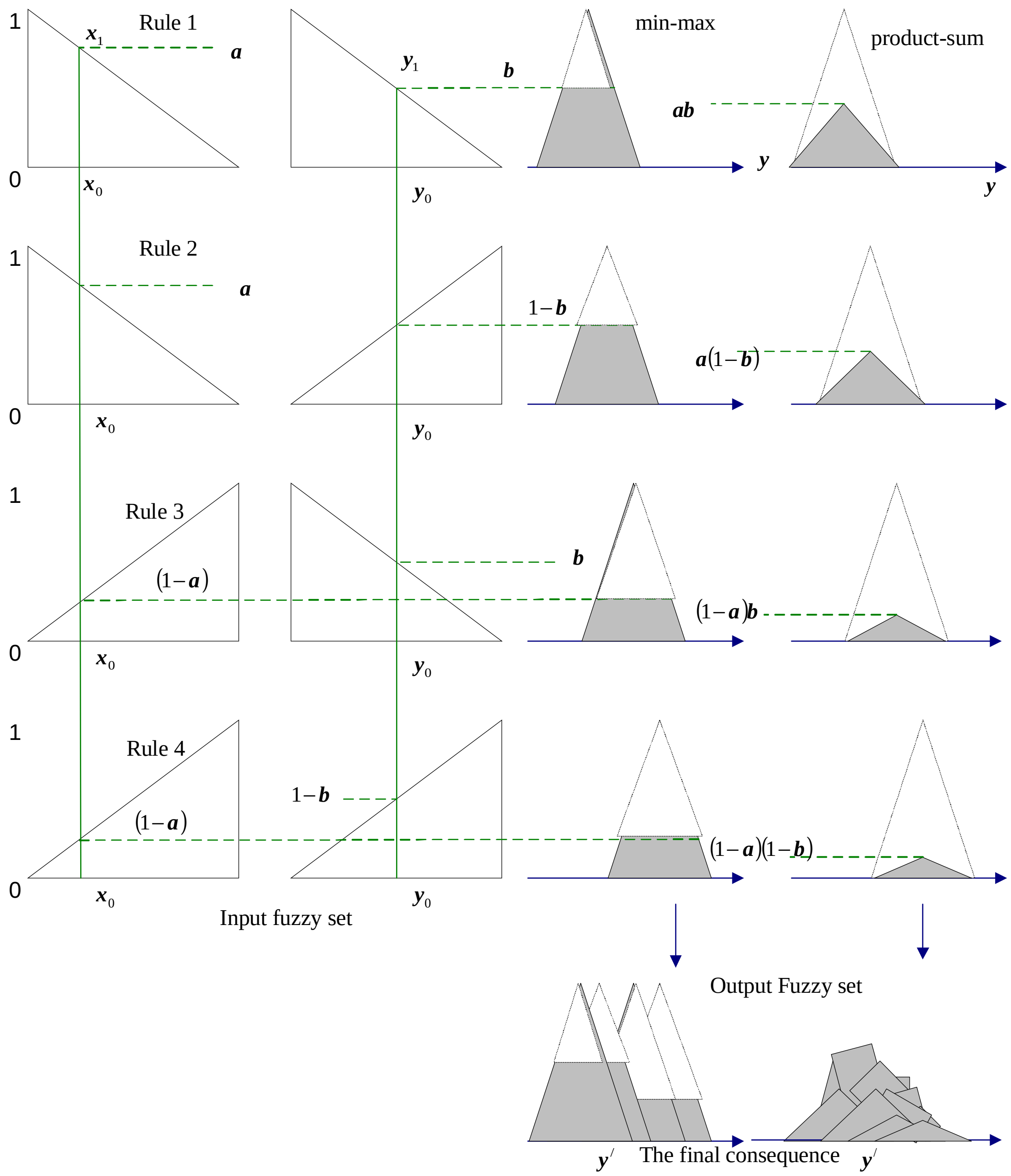


Figure 2.4 Comparison of result from the min-max and product-sum method (Kendel & Langholz, 1994, p. 283)

As indicated by Teodorovic & Vukadinovic (1998), Pappis & Mamdani (1977) attempted to solve the problem of controlling an isolated signalized intersection by using a fuzzy logic system. They introduce four fuzzy (linguistic) variables.

T : The time that has lapsed since the last light changed at the intersection,

A : The number of vehicles from the priority direction that have passed through the green light during the considered time period,

Q : The number of vehicles waiting in line on the one-way street that does not have priority, and

E : The length of time to the next light change.

Fuzzy variables T , A, and Q are input variables whose values determine the value of output variable E. Fuzzy variable A could be assigned the value “many” vehicles, “more than several” vehicles, “few” vehicles, and so on. Fuzzy variable Q could be assigned similar values. Variables T and E are assigned as “very short”, “short”, “medium” time and so on. Pappis and Mamdani also use fuzzy sets such as “any” number of vehicles, “more than” and “less than”. The grade of membership of every element belonging to fuzzy “any” equals 1.

In addition Pappis & Mamdani (1977) propose triangle and trapezoidal forms for the curve of the membership function of a fuzzy set. They also describe an approach to find membership values. For example, consider a fuzzy set M, where a element x^* of set M has the largest grade of membership in set M. Let G be the fuzzy set “greater than M”. Let L be fuzzy set “less than M”. The membership functions of fuzzy sets L and G can be defined as follow:

$$m_G(x) = \begin{cases} 0 & , x \le x^* \\ 1 - m_M(x) & , x > x^* \end{cases}$$

$$m_L(x) = \begin{cases} 1 - m_M(x) & , x < x^* \\ 0 & , x \ge x^* \end{cases}$$

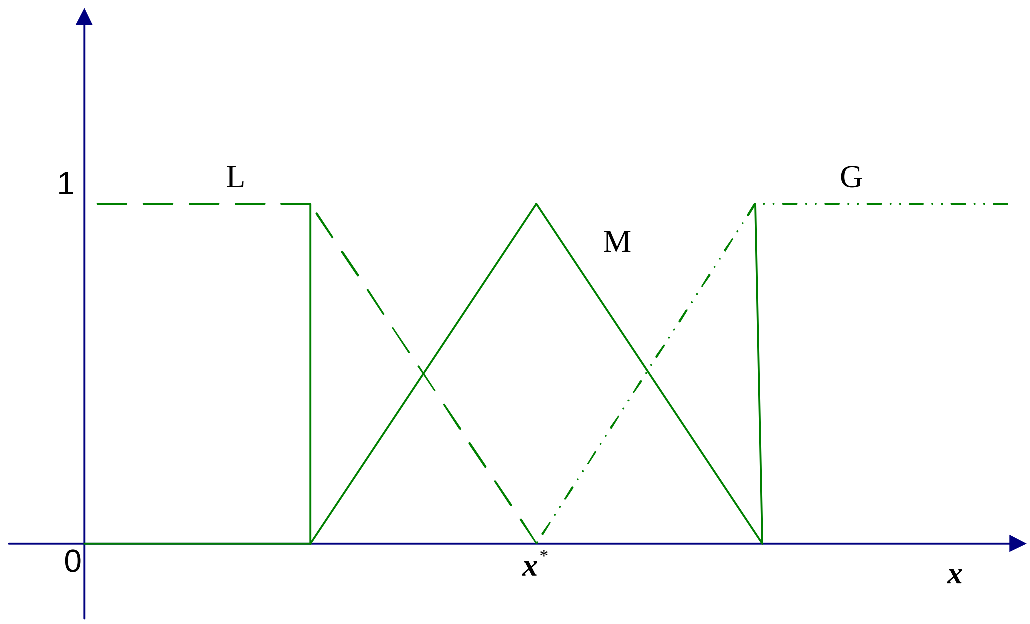


Figure 2.5 Membership function of fuzzy sets M,G and L
(Teodorovic & Vukadinovic, 1998, p. 97)

The algorithm to control traffic at an isolated intersection proposed by Pappis and Mamdani (1977) uses rules of the following type:

Rule 1: IF T is very short and A is greater than none and Q is any, THEN E is very short.
Rule 2: IF T is short and A is greater than few and Q is less than very small, THEN E is short.
Rule 3: IF T is medium and A is greater than few and Q is less than very small, THEN E is medium.
Rule 4: IF T is long and A is greater than medium and Q is less than very small, THEN E is long.
Rule 5: IF T is very long and A is greater than many and Q is less than very small, THEN E is very long.

The values of fuzzy variable E represent the extension of time to allow a vehicle to pass the intersection. The extensions given to the system were between 1 and 10 seconds. Every 10 seconds a different set of five rules is used to make the decision on the length of time to the next light change at the intersection. The min-max-gravity method is used to find the value of the fuzzy variable E based on numerical values t, a, and q for the input variables T, A, and Q respectively.

Kelsey & Bisset (1993) present the simulation of traffic flow and control by using Takagi and Sugeno's fuzzy system. Simulation output can be compared with the output from conventional methods. There are four fuzzy variables in the fuzzy controller.

G : The average density of traffic behind the green light,

R : The average density of traffic behind the red light,

L : The length of the current cycle time, and

C : The index to decide whether to change the state of the light or remain in the same state.

Fuzzy variables G , R, and L are input variables whose values determine the value of output variable C. Fuzzy variable G could be assigned the values "Zero" vehicle, "Low" vehicles, "Medium" vehicles, and "High" vehicles. Fuzzy variable R could be assigned similar values. Variables L could be assigned values "Short" time, "Medium" time and "Long" time. There are four membership functions describing the densities of traffic at green and red lights, and three membership functions describing the length of the current cycle time.

The membership functions are shown in Figure 2.6-2.8 as follows:

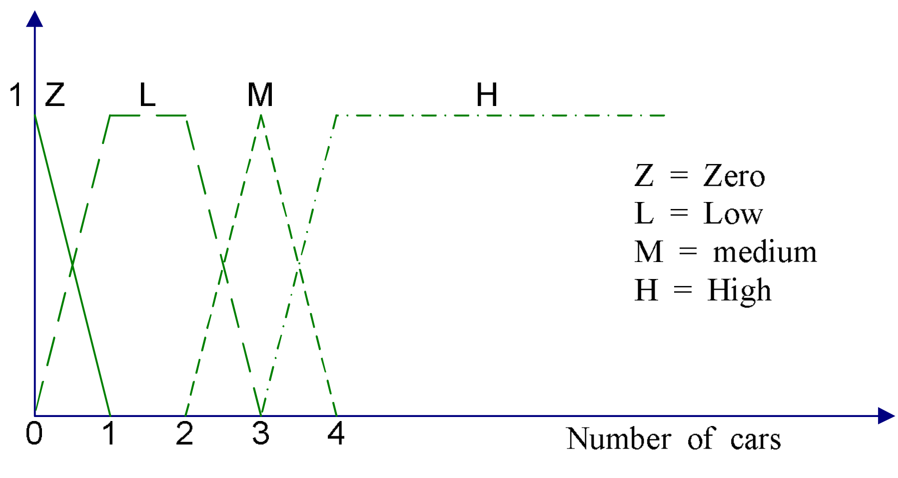


Figure 2.6 Membership function of number of cars behind green light (Kelsey & Bisset, 1993, p. 266)

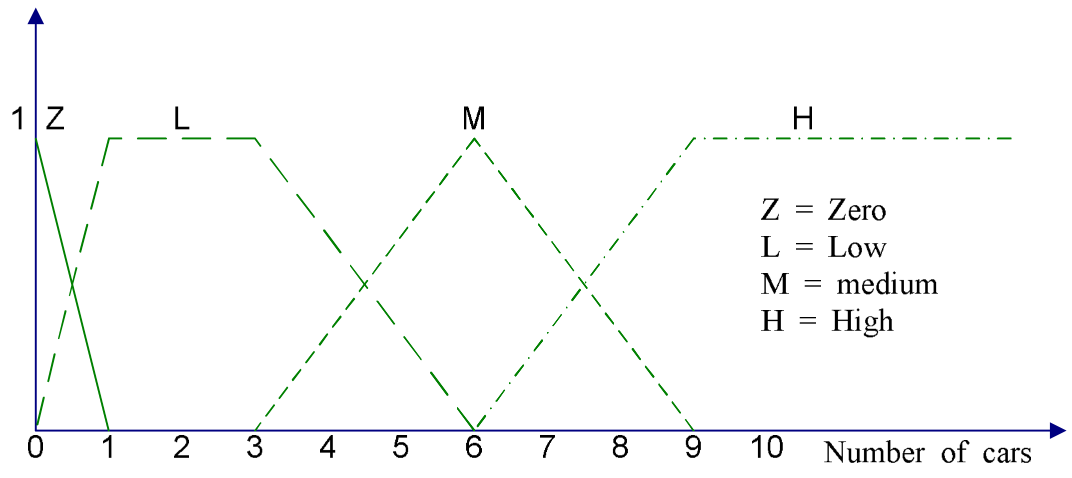


Figure 2.7 Membership function of number of cars behind red light

(Kelsey & Bisset,1993, p. 267)

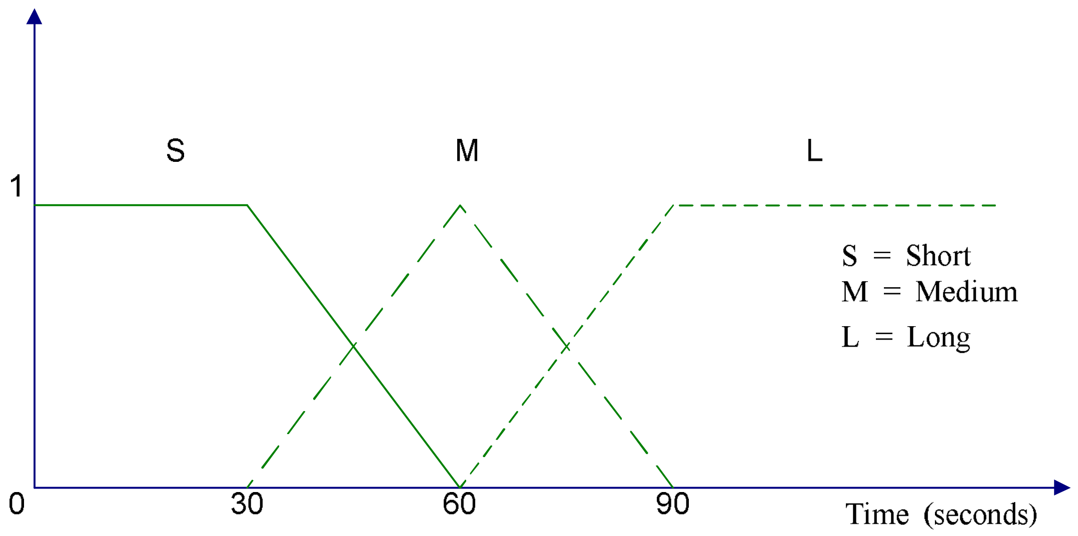


Figure 2.8 Membership function of length of current cycle

(Kelsey & Bisset, 1993, p. 267)

Fuzzy variable C is the output variable whose values are “No change”, “Probably no change”, “Maybe change”, “Probably yes change” and “Change”. The membership function of the values represents a degree of a binary value, 1 being yes and 0 being no, as shown in Figure 2.9

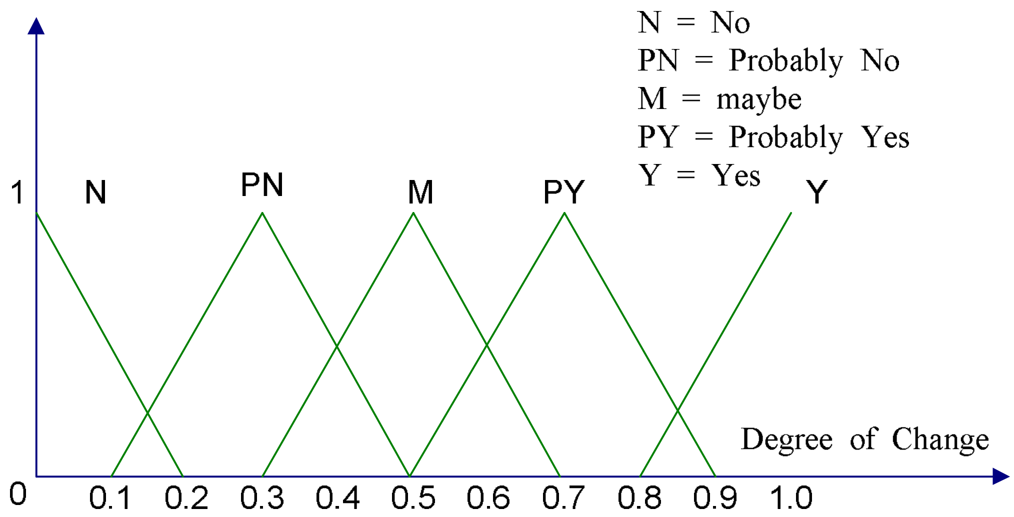


Figure 2.9 Membership function of change (Kelsey & Bisset, 1993, p. 267)

Kelsey & Bisset (1993) also present the fuzzy rule which maps the combination of the inputs to the output to decide whether to change the light. The fuzzy controller presented uses 26 different fuzzy rules as follows:

1. IF green is zero and red is zero THEN change is no.
2. IF green is zero and red is low THEN change is yes.
3. IF green is zero and red is medium THEN change is yes.
4. IF green is zero and red is high THEN change is yes.
5. IF red is zero THEN change is no.
6. IF green is low and red is low THEN change is no.
7. IF green is medium and red is medium THEN change is no.
8. IF green is high and red is high THEN change is no.
9. IF green is low and red is medium and time is short THEN change is maybe.
10. IF green is low and red is medium and time is medium THEN change is probably yes.
11. IF green is low and red is medium and time is long THEN change is yes.
12. IF green is low and red is high and time is short THEN change probably no.

13. IF green is low and red is high and time is medium THEN change is may be.
14. IF green is low and red is high and time is long THEN change is probably yes.
15. IF green is medium and red is low and time is short THEN change is probably no.
16. IF green is medium and red is low and time is medium THEN change is probably no.
17. IF green is medium and red is low and time is long THEN change is maybe.
18. IF green is medium and red is high and time is short THEN change is maybe.
19. IF green is medium and red is high and time is medium THEN change probably Yes.
20. IF green is medium and red is high and time is long THEN change is yes.
21. IF green is high and red is low and time is short THEN change is maybe.
22. IF green is high and red is low and time is medium THEN change probably yes.
23. IF green is high and red is low and time is long THEN change is yes.
24. IF green is high and red is medium and time is short THEN change is probably no.
25. IF green is high and red is medium and time is medium THEN change is probably no.
26. IF green is high and red is medium and time is long THEN change is maybe.

2.2 The traffic intensities estimation based on the maximum likelihood estimation

2.2.1 Maximum likelihood estimation

Maximum likelihood estimation is a method that is used to estimate the parameters of a distribution, or estimate performance of a model . Bera & Bilias (2002) states that the statistical expert who provided the analytical foundation of maximum likelihood estimation is Fisher (1922). He also studied the efficiency of maximum likelihood estimation relative to moment estimation proposed by Karl Pearson's (1894) moment estimation.

Abutaled & Papaioannou (2000) propose maximum likelihood estimation to estimate time-varying parameters in time series models. The result of this approach is then applied to the Athens Stock Exchange Index. Chan & McAleer (2002) use maximum likelihood estimation to investigate the properties of two models of time series, the Smooth Transition Autoregressive (STAR) model and the Smooth Transition Autoregressive Generalized Autoregressive Conditional Heteroscedasticity (STAR-GARCH) model based on finite samples. These numerical results are used as a guide in empirical research, with an application to Standard and Poor's Composite 500 Index returns for alternative STAR-GARCH models.

The likelihood function of a continuous-time diffusion is observed only at discrete dates, and is not computable. Ait-Sahalia (2002) explicitly constructs a sequence of closed-form functions that converges to the true likelihood function, and the estimator also converges to the true maximum likelihood. Eqorov, Li, & Xu (2003) extend the same method to the time-inhomogeneous case, and prove that this approximation converges to the true likelihood function and yields consistent parameter estimates.

Maximum likelihood estimation can be applied in business management and econometrics, for estimation of default correlations between variables in management of loan portfolios (Demey, Jean-Frederic, Roget, & Poncalli, 2004) for example. The estimation overcomes problems such as scarce data and small sample biases. Deschamps (1998) uses full maximum likelihood estimation to estimate parameters in a dynamic demand model. Durtham, Gallant, Ait-Sahalia, & Brandt (2002) propose maximum likelihood estimation to provide a convenient way to describe the dynamics of economic and financial data. O'Loughlin & Coenders (2004) present the maximum likelihood approach as advantageous over the partial least square method in estimation of customer performance. Porter (2002) mentions the use of maximum likelihood in econometrics model estimation, the conditional information matrix variance estimator is usually avoided in choosing a method for estimating the variance of the estimator. The author proposes a simulation method to estimate the variance. Swann (2002) demonstrates a method that can be used to examine a more complicated econometric model.

Frehlich & Sharman (2005) use maximum likelihood estimation to estimate the performance of pulsed coherent Doppler radar in estimating aircraft trailing wake vortices. The estimation provides accurate detection and tracking of the key vortex parameters for a simple vortex model.

Fridman & Harris (1998) develop maximum likelihood estimation to analyze stochastic volatility models. The study shows that the method matches the performance of the best estimation tools currently in use.

Ghitany & Al-Awadhi (2002) propose maximum likelihood to estimate the parameters of Burr XII distribution. The study shown that the estimators are strongly consistent with the true values of parameters.

Gill (2004) uses maximum likelihood estimation to estimate the canonical parameter of an exponential family that gradually begins to drift from its initial value at an unknown change point.

Herring & Ibrahim (2002) introduce maximum likelihood estimation to estimate a random effects cure rate model based on development of the Expectation Maximization (EM) algorithm, and efficient Gibbs sampling. The EM algorithm is also applied by Karlis (2001) to estimate the performance of mixed Poisson regression models based on a real data set concerning crime data from Greece.

Karlis (2003) describe an EM algorithm for maximum likelihood estimation to estimate parameters of the multivariate Poisson distribution model Kim & Taylor (1995) develop a modification of the restricted EM algorithm to estimate linear restriction parameters. Ning-Zhong, Zneng (2005) extend the restricted EM algorithm to estimate the inequality restrictions parameter. Hunter & Lange (2004) claim the EM algorithm is the most effective algorithm for maximum likelihood estimation. In biomedical research, maximum likelihood is used by Lee & Shi (2001) to estimate the performance of the latent variable model. However every EM algorithm is a special case of the more general class of Method of Moment (MM) optimization algorithms,as is shown by Hunter & Lange (2004). The paper explains the principle of MM algorithms and includes numerous examples to illustrate the concept of the algorithm.

Hsiao, Pesaran, & Tahmiscioqlu (2002) apply a transformed likelihood estimation to estimate fixed effects dynamic panel data models. The study shows that the properties of maximum likelihood estimation are better than the linear generalized method of moment estimation.

Jewell (2004) uses maximum likelihood estimation to estimate a series of ordered multinomial parameters. The results are then applied to estimation of a survival distribution. Jonker (2003) proposes maximum likelihood estimation to estimate the life length of people who were born in the seventeenth or eighteenth century in England. Chen & Ibrahim (2001) propose maximum likelihood estimation to estimate the parameter for a novel class of semi-parametric survival models.

Keats, Lawrence, & Wang (1997) present a Fortran program based on point and interval maximum likelihood estimation to estimate the parameters of the Weibull distribution. Kotz, Kozubowski, & Podqorski (2002) use maximum likelihood to estimate the parameters of a univariate asymmetric Laplace distribution for all situations.

Lynch, Nkouka, Huebschmann, & Guldin (2003) use maximum likelihood estimation to estimate parameters for a range of specified probability densities in a logistic equation, where traditional estimation techniques for logistic models cannot be used. On the other hand Horton & Laird (2001) present a new method for maximum likelihood estimation of logistic regression models with incomplete covariate data where auxiliary information is available.

Milescu, Akk, & Sachs (2005) describe maximum likelihood estimation to estimate parameters of rate constants from macroscopic ion channel data for a kinetic model.

Milligan (2003) use maximum likelihood estimation to quantify the statistical performance of the traditional maximum likelihood estimator in relatedness between individuals in genetics and population biology.

Miranda & Rui (1997) introduce an efficient numerical algorithm for computing the full information maximum likelihood estimators of the nonlinear rational expectations asset pricing model. The study show that the maximum likelihood estimator is more efficient than the method of moments estimator.

Rous, Jewell, & Brown (2004) use a full information maximum likelihood estimation procedure to estimate the relationship between birth-weight and prenatal care. The data is collected from the state of Texas, and the result shows the effect of mothers with less healthy fetuses making more prenatal care visits, known as adverse selection in prenatal care.

Scheike & Martinussen (2004) present maximum likelihood to estimate the parameters of interest for case-cohort sampling that aims to reducing the data sampling and costs of large cohort studies. The estimation is found by a simple EM algorithm that is easy to implement.

Yu & Wong (2005) propose a special modification of maximum likelihood estimation to estimate parameters in a linear regression model when the error distribution is unknown. The study shows that the special estimation is consistent, and can be applied to engineering data.

Ellson (1993) suggests that maximum likelihood estimation is one method to learn about the parameters of a population based on the characteristics of a sample. The parameter estimator that we find by maximum likelihood estimation maximizes the joint probability function of a sample we obtain from random sampling. The details of maximum likelihood estimation procedure are as follows.

Let X be a random variable which has a normal distribution with known parameter, s^2 (variance of population) and unknown parameter, m (mean of population). Our goal is to estimate the population mean by maximum likelihood estimation. First we need take a random sample with n size. Let $\left(X_1, X_2, \mathbf{K}, X_n\right)$ be the sample. Random sampling produces independent identically distributed (iid) random variables with joint probability density as follows:

$$
\begin{aligned}
L(x_1, x_2, ..., x_n, m) \quad &= \quad \prod_{i=1}^{n} f(x_i) \\
&= \quad \prod_{i=1}^{n} \frac{1}{\sqrt{2ps^2}} \exp\left(\frac{-(x_i - m)^2}{2s^2}\right)
\end{aligned}
$$

Taking the natural logarithm of both sides of the equation we get the loglikelihood function :

$$\ln L(x_1, x_2, \ldots, x_n, m) = \ln \prod_{i=1}^{n} \frac{1}{\sqrt{2ps^2}} \exp\left(\frac{-(x_i - m)^2}{2s^2}\right)$$

$$= \sum_{i=1}^{n} \ln\left[\frac{1}{\sqrt{2ps^2}} \exp\left(\frac{-(x_i - m)^2}{2s^2}\right)\right]$$

$$= -n(\ln\sqrt{2ps^2}) - \frac{1}{2s^2}\sum_{i=1}^{n}(x_i - m)^2$$

Note that the value of loglikelihood function is dependent only on the term $-\sum_{i=1}^{n}(x_i - m)^2$. We ignore all the constants in the equation because they are not needed to maximize the function. So the estimator of μ that maximizes the likelihood function is computed by calculus as follows:

$$\frac{\partial}{\partial m}\left[-\sum_{i=1}^{n}(x_i - m)^2\right] = 2\sum_{i=1}^{n} x_i - 2nm$$

Let $$2\sum_{i=1}^{n} x_i - 2nm = 0$$

So the maximum likelihood estimate $m = \dfrac{\sum_{i=1}^{n} x_i}{n}$

In the same way let $\boldsymbol{X}$ be a vector of c-dimensional Poisson random variable.

$$\boldsymbol{X} = (X_1, X_2, \mathbf{K}, X_c)$$

$$X_j \sim Poisson(l_j) \text{ ; j = 1,2,…,c. independent.}$$

$$l = (l_1, l_2, \ldots, l_c)$$

$$\boldsymbol{X} \sim Poisson(l)$$

Let us estimate l by using maximum likelihood estimation. First we need to take a random sample with n size. Let $(\boldsymbol{X}^{(1)}, \boldsymbol{X}^{(2)}, \mathbf{K}, \boldsymbol{X}^{(n)})$ be the sample. As before these are independent identically distributed (iid) random variables.

Here $$\boldsymbol{X}^{(1)} = (X_1^{(1)}, X_2^{(1)}, \mathbf{K}, X_c^{(1)})$$

$$\boldsymbol{X}^{(2)} = (X_1^{(2)}, X_2^{(2)}, \mathbf{K}, X_c^{2})$$

$$\text{M} \qquad\qquad \text{M}$$

$$\boldsymbol{X}^{(n)} = (X_1^{(n)}, X_2^{(n)}, \mathbf{K}, X_c^{(n)})$$

$$f(x_j^{(i)}) = \frac{\exp(-l_j) \cdot l_j^{x_j^{(i)}}}{x_j^{(i)}!}$$

The joint probability distribution of $\left(\boldsymbol{X}^{(1)}, \boldsymbol{X}^{(2)}, \mathbf{K}, \boldsymbol{X}^{(n)}\right)$ is as follow:

$$L\left(\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \mathbf{K}, \boldsymbol{x}^{(n)}\right) = \prod_{i=1}^{n} f\left(\boldsymbol{x}^{(i)}\right)$$

$$= \prod_{\boldsymbol{i}=1}^{\boldsymbol{n}} \frac{\exp(-l) \cdot l^{\,\boldsymbol{x}^{(i)}}}{\boldsymbol{x}^{(\boldsymbol{i})}!}$$

$$\ln L\ (\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(\boldsymbol{n})}) \quad = \quad \ln \prod_{\boldsymbol{i}=1}^{\boldsymbol{n}} \frac{\exp(-l) \cdot l^{\,\boldsymbol{x}^{(i)}}}{\boldsymbol{x}^{(\boldsymbol{i})}!}$$

$$= \; -n\ l + \ln(l) \sum_{\boldsymbol{i}=1}^{\boldsymbol{n}} \boldsymbol{x}^{(\boldsymbol{i})} - \sum_{\boldsymbol{i}=1}^{\boldsymbol{n}} \ln \boldsymbol{x}^{(\boldsymbol{i})}!$$

$$\therefore \qquad \frac{\partial \ln \boldsymbol{L}(\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(\boldsymbol{n})})}{\partial l} \qquad = -n + \frac{1}{l} \sum_{\boldsymbol{i}=1}^{\boldsymbol{n}} \boldsymbol{x}^{(\boldsymbol{i})}$$

Let $\dfrac{\partial \ln \boldsymbol{L}(\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(\boldsymbol{n})})}{\partial l} \quad = \quad 0$

$$-n + \frac{1}{l} \sum_{\boldsymbol{i}=1}^{\boldsymbol{n}} \boldsymbol{x}^{(\boldsymbol{i})} = 0$$

So the solution for maximum likelihood estimation is $\quad l \;=\; \dfrac{1}{n} \displaystyle\sum_{\boldsymbol{i}=1}^{\boldsymbol{n}} \boldsymbol{x}^{(\boldsymbol{i})}$

or $\quad l \;=\; \left(\dfrac{1}{n} \displaystyle\sum_{i=1}^{n} x_1^{(i)}, \dfrac{1}{n} \sum_{i=1}^{n} x_2^{(i)}, \mathbf{K}, \dfrac{1}{n} \sum_{i=1}^{n} x_c^{(i)} \right)$

Let $\qquad \boldsymbol{X}$ be a vector of c-dimensional random variable

$\boldsymbol{Y}$ be a vector of r-dimensional random variable

$$\boldsymbol{X} \;=\; \left(X_1, X_2, \mathbf{K}, X_c\right)$$

$$\boldsymbol{Y} \;=\left(Y_1, Y_2, \mathbf{K}, Y_r\right)$$

$$\boldsymbol{X}^{(\boldsymbol{i})} \;=\; \left(X_1^{(i)}, X_2^{(i)}, \mathbf{K}, X_c^{(i)}\right)'$$

$$\boldsymbol{Y}^{(\boldsymbol{i})} \;=\left(Y_1^{(i)}, Y_2^{(i)}, \mathbf{K}, Y_r^{(i)}\right)' \qquad ; \qquad i = 1, 2, \mathbf{K}, n$$

$\boldsymbol{A}$: rxc matrix

Using maximum likelihood estimation as in the previous section, we can estimate l given $\boldsymbol{Y}^{(i)} = \boldsymbol{A}\boldsymbol{X}^{(i)}$ with the likelihood equation in vector notation that can be expressed :

$$l \;=\; \frac{1}{n} \sum_{i=1}^{n} E\ [\boldsymbol{X}^{(i)} \,|\, \boldsymbol{Y}^{(i)} = \boldsymbol{A}\boldsymbol{X}^{(i)}]$$

2.2.2 EM algorithm for maximum likelihood estimation

Kim & Taylor (1995) suggest that the EM algorithm is one of the most powerful algorithms for maximum likelihood estimation in an incomplete data problem. In the EM algorithm it is usually necessary to find the conditional distribution in the E step, then use standard maximum likelihood estimation for the complete data problem in the M step. Let $\boldsymbol{x} = (x_1, x_2, \mathbf{K}, x_n)$ be an observation vector and l be a cx1 parameter vector of interest. Let $f\,(\boldsymbol{x}|l\,)$ be the known probability density of $\boldsymbol{x}$ indexed by the unknown parameter l . Denote the log-likelihood of n observations by $l\,(l\,|\boldsymbol{x})$. If there are no restrictions on the parameter, a fast and popular algorithm for maximizing $\boldsymbol{l}\,(l\,|\boldsymbol{x})$ is the Newton-Raphson algorithm. The score function and the information matrix for the Newton-Raphson algorithm are given by

$$\boldsymbol{S}_U = \frac{\partial \boldsymbol{l}(\,l\mid \boldsymbol{x})}{\partial\, l} \quad \text{and} \quad \boldsymbol{I}_U = -\frac{\partial^2 \boldsymbol{l}(\,l\mid \boldsymbol{x})}{\partial^2\, l}$$

where $\boldsymbol{I}_U$ is assumed to be positive definite. So an unrestricted maximum likelihood estimate of l is a solution of a set of iterations given by

U1. $i \leftarrow 0$; choose a starting value for l , denoted by $l_{\boldsymbol{U}(0)}$.

U2. $l_{\boldsymbol{U}(\boldsymbol{i}+1)}[\,l_{\boldsymbol{U}(\boldsymbol{i})}] \leftarrow l_{\boldsymbol{U}(\boldsymbol{i})} + \boldsymbol{I}_{\boldsymbol{U}}{}^{-1}\boldsymbol{S}_{\boldsymbol{U}}$, where $\boldsymbol{S}_U$ and $\boldsymbol{I}_U$ are evaluated at $\mathrm{l}_{\mathbf{U(I)}}$. Stop if $l_{\boldsymbol{U}(\boldsymbol{i})}$ has converged.

U3. $l_{\boldsymbol{U}(\boldsymbol{i}+1)} \leftarrow l_{\boldsymbol{U}(\boldsymbol{i}+1)}[\,l_{\boldsymbol{U}(\boldsymbol{i})}]$; $i \leftarrow i+1$ go to U2.

In U2, $l_{U(i+1)}[\,l_{U(i)}]$ denotes the $(i+1)$ th term in the Newton-Raphson sequence for the unrestricted problem obtained by taking one Newton-Raphson step from $l_{\boldsymbol{U}(\boldsymbol{i})}$. Now suppose there are r linearly independent restrictions on the parameter λ, such as

$$\boldsymbol{Y} = \boldsymbol{A}\,l$$

Here $\boldsymbol{A}$ is the known rxc matrix defining the restrictions, with rank$(\boldsymbol{A})$ = r < c ; and $\boldsymbol{Y}$ is a known rx1 vector. We use the Lagrange multiplier method to derive an algorithm to find the restricted maximum likelihood estimation. When the Lagrange multiplier method is used to incorporate the restrictions, the restricted log-likelihood is given by

$$l\,(l\,|\boldsymbol{x}, q\,) = l\,(l\,|\boldsymbol{x}) - q\,'(\boldsymbol{Y} - \boldsymbol{A}\,l\,),$$

where $q = (q_1, q_2, ..., q_r)$ are the Lagrange multipliers. When q is given , the procedure for maximization of the restricted log-likelihood $\boldsymbol{l}\left(l \,|\, \boldsymbol{x}, q\right)$ is the same as the unrestricted maximization in U1-U3. A simple adaptation of the Newton-Raphson iteration scheme leads to the restricted solution. The score function and the information matrix for the restricted log-likelihood can be expressed as

$$\boldsymbol{S_R} = \frac{\partial \boldsymbol{l}(l \mid \boldsymbol{x}, q)}{\partial l} = \boldsymbol{S_U} + \boldsymbol{A}' q \quad \text{and} \quad \boldsymbol{I_R} = -\frac{\partial^2 \boldsymbol{l}(l \mid \boldsymbol{x}, q)}{\partial^2 l} = \boldsymbol{I_U}.$$

From the relationship of the score functions and information matrices between the unrestricted and restricted problems, we can easily verify that the Lagrange multiplier is a function of the unrestricted solution and the unrestricted information matrix. A sequence $l_{R(0)}, l_{R(1)}, l_{R(2)}, ...$ for the restricted problem is obtained by the following algorithm:

R1. $i \leftarrow 0$ choose a starting value , $l_{R(0)}$.

R2. Calculate $l_{U(i+1)}[l_{R(i)}]$ from U2 for the unrestricted problem.

R3. Calculate $l_{R(i+1)}$ for the restricted problem from the following equation:

$$l_{\boldsymbol{R}(i+1)} = l_{\boldsymbol{U}(i+1)}[l_{\boldsymbol{R}(i)}] + \boldsymbol{I_U}^{-1}\boldsymbol{A}'(\boldsymbol{A}\boldsymbol{I_U}^{-1}\boldsymbol{A}')^{-1}(\boldsymbol{Y} - \boldsymbol{A}\, l_{\boldsymbol{U}(i+1)}[l_{\boldsymbol{R}(i)}]),$$

where $\boldsymbol{I_U}$ are evaluated at $l_{\boldsymbol{R}(i)}$. Stop if $l_{\boldsymbol{R}(i)}$ has converged.

R4. $i \leftarrow i+1$, go to R2

From R3, it is clear that each member of the sequence for the restricted problem is easily obtained in each iteration by using the unrestricted solution and information matrix.

2.2.3 Estimating source-destination traffic intensity from link data

Vardi (1996) estimate source - destination traffic intensities from link data based on maximum likelihood estimation and the sample moments approach. The method is presented below.

Consider a network system that contains n nodes. Any two nodes are fixed; one as the source, and the other as the destination, and they are called source-destination pairs (SD), or direct routes. The target is a traffic intensity estimator between two nodes. This network system is composed of c = n (n-1) SD, and we call the direct route that has no nodes between source and destination a direct link. The number of direct links in this network system are r (r$\leq$ c).

Let $X_j^{(k)}$ be the number of vehicles for direct route j at measurement period k.

We assume that

$X_j^{(k)} \sim Poisson\left(l_j\right)$; $j = 1,2,\mathbf{K},c; k = 1,2,\mathbf{K},K$. independent.

$\boldsymbol{X}^{(\boldsymbol{k})}$ is the number of vehicles in vector form for the direct route.

$\boldsymbol{X}^{(\boldsymbol{k})} = \left(X_1^{(k)}, X_2^{(k)}, \mathbf{K}, X_c^{(k)}\right)^{/}$

$Y_i^{(k)}$ is the number of vehicles that are observed from direct link i at measurement period k.

$\boldsymbol{Y}^{(\boldsymbol{k})}$ is the number of vehicles in vector form for direct links.

$\boldsymbol{Y}^{(\boldsymbol{k})} = \left(Y_1^{(k)}, Y_2^{(k)}, \mathbf{K}, Y_r^{(k)}\right)^{/}$

Let $\boldsymbol{A}$ be the rxc routing matrix for this network. The matrix $\boldsymbol{A}$ is a zero-one matrix whose rows correspond to the direct link; its columns correspond to direct routes, and its entry, a_{ij} is 1 or 0 according to whether link i does or does not belong to the direct path of the SD pair j. So we derive the relation between $\boldsymbol{Y}^{(\boldsymbol{k})}$ and $\boldsymbol{X}^{(\boldsymbol{k})}$ in equation form as

$$\boldsymbol{Y}^{(\boldsymbol{k})} = \boldsymbol{A}\boldsymbol{X}^{(\boldsymbol{k})}; \qquad k = 1,2,\mathbf{K},K$$

Our goal is to estimate $l \equiv \left(l_1, l_2, \mathbf{K}, l_c\right)^{/}$ from $\boldsymbol{Y}^{(1)}, \boldsymbol{Y}^{(2)}, \mathbf{K}, \boldsymbol{Y}^{(\boldsymbol{k})}$ based on maximum likelihood estimation and sample moments.

The likelihood equations in vector notation can be expressed as

$$l = \frac{1}{K}\sum_{k=1}^{K} E\ [\boldsymbol{X}^{(\boldsymbol{k})} | \boldsymbol{Y}^{(\boldsymbol{k})} = \boldsymbol{A}\boldsymbol{X}^{(\boldsymbol{k})}] \qquad (2.1)$$

The EM algorithm can be used to search for the solution of equation (2.1), and the EM algorithm, in vector notation , is

$$l^{(n+1)} = E\ [\overline{\boldsymbol{X}} | \boldsymbol{Y}^{(1)}, \boldsymbol{Y}^{(2)}\ \mathbf{K}, \boldsymbol{Y}^{(\boldsymbol{k})}, l^{(\boldsymbol{n})}] \quad , n = 1,2,\mathbf{K}$$

($l^{(0)} > 0$, arbitrary). due to of the linearity of E and independence across k's ,

$$l^{(\boldsymbol{n}+1)} = \frac{1}{K}\sum_{\boldsymbol{k}=1}^{\boldsymbol{k}} E\left[\boldsymbol{X}^{(\boldsymbol{k})} \mid \boldsymbol{Y}^{(\boldsymbol{k})}, l^{(\boldsymbol{n})}\right] \quad , n = 1,2,\mathbf{K}$$

The trouble with this iteration formula is that the summands $E\left(\boldsymbol{X}|\boldsymbol{Y},\, l\right)$ (superscripts ignored for simplicity) are extremely hard to calculate as they require finding all the solutions in natural numbers of $\boldsymbol{Y} = \boldsymbol{AX}$.

Approximation of $\overline{\boldsymbol{Y}} = \dfrac{1}{K}\sum_{\boldsymbol{k}=1}^{\boldsymbol{K}} \boldsymbol{Y}^{(\boldsymbol{k})}$ is possible when k is large as $\overline{\boldsymbol{Y}}$ is approximates a multivariate normal distribution.

$$\overline{\boldsymbol{Y}} \sim \ \boldsymbol{N_r}(\boldsymbol{A}\,l\,,\ K^{-1}\,\boldsymbol{ALA}^{/})\ ,\quad L = diag\left(l\right) \qquad ,$$

So the log-likelihood of $\overline{\boldsymbol{Y}}$ is

$$l\left(l\right) = -\log\left|\boldsymbol{ALA}^{/}\right| - K\left(\overline{\boldsymbol{Y}} - \boldsymbol{A}\,l\right)^{/}\left(\boldsymbol{ALA}^{/}\right)^{-1}\left(\overline{\boldsymbol{Y}} - \boldsymbol{A}\,l\right) \tag{2.2}$$

The maximum likelihood estimation (MLE) based on this approximation would seek to maximize $l\left(l\right)$ subject to the constraints $l_i \geq 0$, i = 1, $\mathbf{K}$, c. When K is large, the second term is the dominant term in (2.2), and suggests

$\arg\min_{\lambda\geq 0}\ K\left(\overline{\boldsymbol{Y}} - \boldsymbol{A}\,l\right)^{/}\left(\boldsymbol{ALA}^{/}\right)^{-1}\left(\overline{\boldsymbol{Y}} - \boldsymbol{A}\,l\right)$ as a reasonable large-sample substitute for the MLE. Note that this is a weighted least square with positive constraints and with weighted values depending on l , which can be estimated by the sample covariance matrix of the Y's .

The approximate normal distribution of $\overline{\boldsymbol{Y}}$ is completely determined by the mean vector, $\boldsymbol{A}\,l$, and covariance matrix, $\boldsymbol{ALA}^{/}$, of $\boldsymbol{Y}$. Thus we can equate the sample's first and second moment to their theoretical values to obtain a linear (in l) system of the following estimating equations :

$$\hat{E}\,(Y_i) = \overline{Y}_i = \sum_{l=1}^{c} a_{ij}\, l_j \qquad , \quad \text{i} \ = \ 1\,,2\,,\ldots\,,\text{r}$$

and $$C\hat{O}V(Y_i, Y_{i^/}) = \frac{1}{K}\sum_k Y_i^{(k)} Y^{(k)}{}_{i'} - \overline{Y}_i \overline{Y}_{i'} = \sum_{j=1}^{c} a_{ij}\, a_{i'l}\, l_j\ ,\ \ 1 \leq \text{i} \leq \text{i}' \leq \text{r}\,.$$

These are $\dfrac{r(r+3)}{2}$ linear equations that can be written in vector notation as

$$\begin{bmatrix} \overline{\boldsymbol{Y}} \\ \boldsymbol{S} \end{bmatrix} = \begin{bmatrix} \boldsymbol{A} \\ \boldsymbol{B} \end{bmatrix} l \tag{2.3}$$

Here $\boldsymbol{S}$ is the sample covariance matrix stretched out as a vector of length $\dfrac{r(r+1)}{2}$

$$S_{ii^/} = \frac{1}{K}\sum_k Y_i^{(k)} Y_{i^/}^{(k)} - \overline{Y}_i \overline{Y}_{i^/}$$

$\boldsymbol{B}$ is an $\frac{r(r+1)}{2} \times c$ matrix with rows indexed by $(i, i^{/})$, $1 \le i \le i' \le r$, to match the indexing of $\boldsymbol{S}$, with the $(i, i^{/})$ th row of B as the element – wise product of row i and row $i^{/}$ of the matrix $\boldsymbol{A}$.

Here, suppose that all the constants on left side of (2.3) are strictly positive and that $\boldsymbol{B}$ has no rows of zeros. Then, because all of the entries of $\boldsymbol{A}$ and $\boldsymbol{B}$ are nonnegative and $(\overline{\boldsymbol{Y}}^{/}, \boldsymbol{S}^{/})^{/} > 0$ and l is constrained to be > 0, equation (2.3) is of the general from of a LININPOS (Linear inverse positive) problem, the EM algorithm will be used to "solve" it .The canonical form of the EM iteration for solving the LININPOS problem is that $\overline{\boldsymbol{Y}} = \boldsymbol{A} l$ is

$$l_j \leftarrow \frac{l_j}{\sum_{i=1}^{r} a_{ij}} \sum_{i=1}^{r} \frac{a_{ij} \overline{Y}_i}{\sum_{k=1}^{c} a_{ik} l_k} \quad ; \ j = 1, 2, \ldots, c \tag{2.4}$$

If the linear system is given in a block form as

$$\begin{bmatrix} \overline{\boldsymbol{Y}} \\ \boldsymbol{S} \end{bmatrix} = \begin{bmatrix} \boldsymbol{A} \\ \boldsymbol{B} \end{bmatrix} l \quad ,$$

where $\boldsymbol{A}$ is rxc , $\overline{\boldsymbol{Y}}$ is rx1 , $\boldsymbol{S}$ is mx1 (indexed as r+1 ,…, r+m) and $\boldsymbol{B}$ is mxc (rows indexed as r+1 ,…,r+m) , then (4) becomes

$$l_j \leftarrow \frac{l_j}{\sum_{i=1}^{r} a_{ij} + \sum_{i=r+1}^{r+m} b_{ij}} \left[\sum_{i=1}^{r} \frac{a_{ij} \overline{Y}_i}{\sum_{k=1}^{c} a_{ik} l_k} + \sum_{i=r+1}^{r+m} \frac{b_{ij} S_i}{\sum_{k=1}^{r+1} b_{ik} l_k} \right].$$

Also, Vardi (1996) presents the steps of the simulation process to estimate traffic intensity by using Maximum likelihood based on the EM algorithm as follows:

Step 1 Let $l = (l_1, l_2, \ldots, l_c)$ be 'daily transmission' rate;

Step 2 Generate daily data on direct links for k days :

$$\boldsymbol{Y}^{(1)} \equiv \left(Y_1^{(1)}, Y_2^{(1)}, \ldots, Y_r^{(1)}\right)$$

$$\boldsymbol{Y}^{(2)} \equiv \left(Y_1^{(2)}, Y_2^{(2)}, \ldots, Y_r^{(2)}\right)$$

$$\vdots \qquad \qquad \vdots$$

$$\boldsymbol{Y}^{(k)} \equiv \left(Y_1^{(k)}, Y_2^{(k)}, \ldots, Y_r^{(k)}\right)$$, then

Calculate $\quad \boldsymbol{\overline{Y}} = \dfrac{1}{K}\sum_{\boldsymbol{i}=1}^{\boldsymbol{k}} \boldsymbol{Y}^{(\boldsymbol{k})}$,

and sample covariance matrix, $\boldsymbol{S}$, where

$$S_{ii'} = \frac{1}{K}\sum_{k} Y_i^{(k)} Y_{i'}^{(k)} - \overline{Y}_i \, \overline{Y}_{i'} \quad ;$$

Step 3 Estimate $\hat{l} = (\hat{l}_1, \hat{l}_2, ..., \hat{l}_{\boldsymbol{c}})'$ based on applied algorithm

$$l_j \leftarrow \frac{l_j}{\sum_{i=1}^{r} a_{ij} + \sum_{i=r+1}^{r+m} b_{ij}} \left[\sum_{i=1}^{r} \frac{a_{ij}\overline{Y}_i}{\sum_{k=1}^{c} a_{ik}\, l_k} + \sum_{i=r+1}^{r+m} \frac{b_{ij} S_i}{\sum_{k=1}^{r+1} b_{ik}\, l_k} \right];$$

Step 4 Go to step 2 to estimate $\hat{l}$ m time to get $\hat{l}^{(1)}, \hat{l}^{(2)}, ..., \hat{l}^{(\boldsymbol{m})}$; and

Step 5 Calculate mean vector; $\overline{\hat{l}} = \dfrac{1}{m}\sum_{k=1}^{m} \hat{l}^{(k)}$ and covariance matrix based on m estimations then we get $\overline{\hat{l}}$ which is the unbiased estimator of l , route count.

2.3 The traffic intensities estimation based on Bayesian inference

2.3.1 The Bayesian approach

Moore (1997) agrees that Bayesian method are increasingly important to infer parameters. Bayesian inference is a process that can be used to infer interesting parameters. The main idea of the Bayesian approach according to VerevKa & Parasyuk (2002) consists of sequential calculations of a posterior probability distribution function of the parameter, based on some collection of associated evidence by using Bayes' theorem. Carin, Stern & Rubin (1995) present that posterior distribution is complicate model so it is difficult to directly sampling from the posterior distribution. The indirectly method to sampling when it is very hard to finding distribution function is Gibbs sampling. Casella & George (1992) support the meaning of Gibbs sampling as a technique for generating random variables from a distribution indirectly, without having to calculate the density distribution fuction. Additional Gibbs sampling is based only on elementary properties of Markov Chains.

Erkanli, Soyer, & Costello (1999) use Baysian inference and model selection for a prevalente estimation to estimate the interesting parameter. They generate random variable from distribution by using Markov Chain Monte Carlo method. Geweke (1989) develop the method for the systematic application of Monte Carlo integration to sampling for Bayesian inference in econometric model. In addition Jensen (2004) proposes Baysian inference to estimate the parameter for the integration model. He also uses Markov Chain Monte Carlo method to generate random variable from posterior distribution function for the tractional order of the integration model. According to Carin, Stern & Rubin (1995) two effective methods that can be used to generate random variables in Markov chain Monte Carlo method are Metropolis-Hasting and Gibb sampler. Liu & Sabatti (2000) comment that although Monte Carlo methods have frequently been applied with success in Bayesian inference, indiscriminate use of Markov chain Monte Carlo method leads to unsatisfactory performances in numerous applications. They propose a generalized version of the Gibbs sampler that is based on conditional moves along the traces of groups of transformations in the sample space.The sampler provides a framework encompassing a class of recently proposed tricks such as parameter expansion and reparameterisation.

Blackwell (2003) uses fully Bayesian inference based on hybrid Markov chain Monte Carlo methods, with a mixture of Gibbs sampler and the Metropolis-Hasting algorithm to infer a parameter of the certain radio-tracking model.

Haqqer, Janss, Kadarmideen & Stranzinger (2004)use Bayesian inference to study the parameters of a mixed inheritance model. The Gibbs sampler is used to sample values of the important random variables that are of concern in the inference model such as, body weight and average egg weight. The sequential sampling deliver a random walk that converges to its posterior distribution which helps understanding of the model. Fouqere & Kamionka (2003) use Bayesian inference procedures for the continuous time mover-stayer model. The Gibbs sampler algorithm is applied to estimate proportions of stayers and functions of these parameters.

Chen, Ibrahim & Lipsitz (2002) propose Bayesian inference for missing data with a novel class of semi parametric survival models. The study delivers an informative class of joint prior distributions for the regression coefficients and the parameters arising from the covariate distribution. It is useful in recovering information on the missing covariates. Chopin & Pelqrin (2004) use Bayesian inference on the switching regression model based on the hidden Markov method. The study delivers a joint estimation of the parameter and the number of regimes. Corander & Villani (2004) consider Bayesian inference for the dimensionality in the multivariate reduced rank regression framework. The inference deliver a closed form approximation to the posterior distribution of the dimensionality proven.

Dunson & Herring (2003) propose Bayesian inference for testing the predictor in a Cox model. The inference is use to test null hypothesis that present no difference between an ordered category predictor with an order restricted. The null hypothesis is versus alternative hypothesis that present a monotone increase across level of the predictor. On the other hand in biomedical studies, usually interest in assessing the association between one or more ordered categorical predictor and outcome variable. Duson & Neelon (2003) propose a general Bayesian approach for inference on order-constrained parameters in generalized linear models. The output from the Gibbs sampler is used for assessing ordered trends.

Geweke, Gowrisankaran, & Town (2003) develop the new economic method based on Bayesian inference to infer hospital quality in a model. A dependent variable in the model is mortality rates and an independent variable is hospital admission. The study finds the smallest and largest hospitals to be of the highest quality.

Huelsenbeck, Ronguist, Nielsen, & Bollback (2001) propose Bayesian inference for a phylogeny model. The study finds a new perspective to a number of outstanding issues in evolutionary biology, including the analysis of large phylogenetic trees and complex evolutionary models and the detection of the footprint of natural selection in DNA sequences.

Kleiberqen (2004) proposes Bayesian inference to explain a nested regression model. The study obtained the prior and posterior probability that can be used to represent the nested model. Odejar & McNulty (2001) develop Bayesian methods to estimate the parameter of a stochastic switching regression model. Markov Chain Monte Carlo methods, data augmentation, and Gibbs sampling are used to facilitate estimation of the posterior means. Paiqe & Butler (2001) develope and approximate marginal Bayesian inference for neural network models. The study describes the method in the context of two nonlinear datasets that involve univariate and multivariate nonlinear regression models.

Lazar (2003) compares empirical likelihood tests and Bayesian inference. The study shows that empirical likelihood tests have many of the same asymptotic properties as those derived from parametric likelihoods. This leads naturally to the possibility of using empirical likelihood as the basis for Bayesian inference. Nair, Tang & Xu,(2001) propose Bayesian inference for three important mixture problems in quality and reliability instead of the traditional, maximum likelihood approach in situations where the large-sample normal approximation is not adequate.

Liu & Lawrence (1999) propose full Bayesian inference to infer the parameter in the bioinformatics method. Bayesian inference is use to assign probabilities for all possible values of all unknown variables in a problem in the form of a posterior distribution. The study show that information from the posterior distribution can be achieved for most bioinformatics method that use dynamic programming.
Martin (2003) present an integrated set of Bayesian tools for heterogeneous event counts model, and compares the method with the traditional approach.

Rovers et al.(2005) focuses on the debate concerning Bayesian inference approach. The issue of the debate involves comparison the posterior distribution that is calculated from Bayes' theorem with the posterior distribution from empirically measure. Their trial was undertaken based on prior and posterior belief among surgeons. The results showed that the trial had a little or no impact on the beliefs of the surgeons, that is, the mean the posterior belief did not adjust to the extent that was expected according to Bayes' theorem.

Oh, Choi & Kim (2003) apply Bayesian inference to the latent class model. The study consists of parameter estimation and selection of an appropriate number of classes. The Gibbs sampler is used to generate the random variable from a posterior distribution of unknown parameters. Output from the Gibbs sampler is used to estimate the parameter and select an appropriate number of classes.

Pasquale, Barone, Sebstiani & Stander (2004) develop Bayesian inference, by means of Markov chain Monte Carlo algorithms, for dynamic magnetic resonance images of the breast. The results show the potential of the methodology to extract useful information from acquired dynamic magnetic resonance imaging data about tumour morphology and internal pathophysiological features.

Blackwell (2001) proposes Bayesian inference for an inhomogeneous Poisson point process. The Markov chain Monte Carlo approach is applied in the point of observation process. The results of the study can be applied to modeling the territories of clans of badgers. Roberts, Papaspiliopoulos, & Dellaportas (2004) develop Markov chain Monte Carlo methodology for Bayesian inference for non-Gaussian Ornstein-Uhlenbeck stochastic volatility processes. The Metropolis-Hastings algorithms is used to generate the point process and model parameter.

Piles, Gianola, Varona & Blasco (2003) present Bayesian implementation via Markov chain Monte Carlo method for a cross-sectional trait model. The study contains a hierarchical model and a cross-sectional assessment. The hierarchical model is used to infer the parameters of joint distribution fucntion that provides distribution of a longitudinal trait. Basu, Banerjee, & Sen (2000) apply Markov chain Monte Carlo method in Bayesian inference to infer Cohen's kappa coefficient, a widely popular measure for chance-corrected nominal scale agreement between two rates.

Carey, Baker, & Platt (2001) use the Gibbs sampler for Bayesian inference to infer the minimum protective antibody concentration, a quantity of great interest in the study of immune responses to infectious pathogens. Wang, He & Sun (2005) presents capture-recapture methods using Bayesian inference. The method is used to estimate the total number of people with a certain disease in a certain research area. Several lists with information about patients are used as input and the results are useful in epidemiology. Waqner & Gill (2005) point out that the classical statistical inference approach in public administration is defective and should be replaced. They support Bayesian inference as better suited for structuring scientific research into administrative questions due to overt assumptions, flexible parametric forms, systematic inclusion of prior knowledge, and rigorous sensitivity analysis.

Carlin & Louis (1996) state that inferential statistics used with Bayesian approach on the basis of targeted population parameters estimation can be applicable to the observed data which is $\boldsymbol{y} = (y_1, y_2, \mathbf{K}, y_n)$. This application can be done by taking the likelihood function of $\boldsymbol{y}$ when specifying the vector of unknown parameter $q = (q_1, \mathbf{K}, q_k)$. Such likelihood function is actually represented by $f\ (\boldsymbol{y}|q)$. For Bayesian approach q refers to a random vector with the prior distribution function as $p\ (q|h)$, when h is a vector of hyperparameters(the parameter of q). This allows the application of distribution function of q to be more appropriate expressed as

$$p\ (q|\boldsymbol{y},h)\ =\ \frac{\boldsymbol{p}(\boldsymbol{y},q|h)}{\boldsymbol{p}(\boldsymbol{y}|h)} = \frac{\boldsymbol{p}(\boldsymbol{y},q|h)}{\int \boldsymbol{p}(\boldsymbol{y},q|h)\boldsymbol{d}q} = \frac{\boldsymbol{f}(\boldsymbol{y}|q)p(q|h)}{\int \boldsymbol{f}(\boldsymbol{y}|q)p(q|h)\boldsymbol{d}q}$$

The integral in the denominator is sometimes written as $m\ (\boldsymbol{y}|h)$, the marginal distribution of the data $\boldsymbol{y}$ given value of the hyperparameter h . The reformed p function is taken as a posterior distribution function which is used to estimate q . Because of h is constant, it is not repeated in the condition of posterior distribution function. It is therefore represented in a simpler form $p\ (q|\boldsymbol{y})$. The posterior mean of random variable q in posterior distribution function is a weighted average of prior mean and observed data with inversely proportional weights to the corresponding variances. Also, the posterior variance is smaller than that of prior variance and variance of random variable of the likelihood function. As seen, inferential statistics Bayesian depending on the posterior distribution function is a more accurate means in parameter estimation q .

Given a sample of n independent observations, the likelihood function $f\ (\boldsymbol{y}|q)$ is $\prod_{i=1}^{n} f(y_i|q)$. One can proceed with the posterior distribution, $p\ (q|\boldsymbol{y},h)$. Evaluating this expression may be simpler if we can find a statistic $S\ (\boldsymbol{y})$ which is sufficient for q , that is, for which $f\ (\boldsymbol{y}|q) = h\ (\boldsymbol{y})\ g\ (S\ (\boldsymbol{y})|q)$. Let $S\ (\boldsymbol{y}) = \boldsymbol{s}$ then

$$\begin{aligned} p\ (q|\boldsymbol{y}) &= \frac{\boldsymbol{f}(\boldsymbol{y}|q)p(q)}{\int \boldsymbol{f}(\boldsymbol{y}|q)p(q)\boldsymbol{d}q} \\ &= \frac{\boldsymbol{h}(y)\boldsymbol{g}(\boldsymbol{S}(y)|q)p(q)}{\int \boldsymbol{h}(y)\boldsymbol{g}(\boldsymbol{S}(y)|q)p(q)\boldsymbol{d}q} \\ &= \frac{\boldsymbol{g}(\boldsymbol{s}|q)p(q)}{\boldsymbol{m}(\boldsymbol{s})} \\ &= p\ (q|\boldsymbol{s}) \end{aligned}$$

By suppressing the dependence of the prior on the known value of h , $p\left(q\,|\,\boldsymbol{y}\right)$ may be expressed in the convenient shorthand

$$p\left(q\,|\,\boldsymbol{y}\right) \propto f\left(\boldsymbol{y}|q\right) p\left(q\right)$$

Bayes' theorem may also be used sequentially: suppose we have two independently collected samples of data, $\boldsymbol{y}_1$ and $\boldsymbol{y}_2$. Then

$$\begin{aligned} p\left(q\,|\,y_1, y_2\right) &\propto f\left(\boldsymbol{y}_1, \boldsymbol{y}_2|q\right) p\left(q\right) \\ &= f_2\left(\boldsymbol{y}_2|q\right) f_1\left(\boldsymbol{y}_1|q\right) p\left(q\right) \\ &\propto f_2\left(\boldsymbol{y}_2|q\right) \boldsymbol{p}\left(q\,|\,\boldsymbol{y}_1\right) \end{aligned}$$

That is, we can obtain the posterior for the full dataset $\left(\boldsymbol{y}_1, \boldsymbol{y}_2\right)$ by first finding $p\left(q\,|\,y_1\right)$ and then treating it as the prior for the second portion of the data $\boldsymbol{y}_2$.

In case the appropriate value of h is not known or uncertain Bayesian inference approach will take h as a random variable with prior distribution function as $h(h)$. Posterior distribution function calculation of q can therefore be done by also marginalizing over h ,

$$\begin{aligned} p\left(q\,|\,\boldsymbol{y}\right) &= \frac{\boldsymbol{p}(\boldsymbol{y},q)}{\boldsymbol{p}(\boldsymbol{y})} \\ &= \frac{\int \boldsymbol{p}(\boldsymbol{y},q,h)\boldsymbol{d}h}{\iint \boldsymbol{p}(\boldsymbol{y},q,h)\boldsymbol{d}h\boldsymbol{d}q} \\ &= \frac{\int \boldsymbol{f}(y|q)\,\boldsymbol{p}(q,h)\boldsymbol{d}h}{\iint \boldsymbol{f}(y|q)\,\boldsymbol{p}(q,h)\boldsymbol{d}h\boldsymbol{d}q} \\ &= \frac{\int \boldsymbol{f}(y|q)p\left(q\,|\,h\right)\boldsymbol{h}(h)\boldsymbol{d}h}{\iint \boldsymbol{f}(y|q)p\left(q\,|\,h\right)\boldsymbol{h}(h)\boldsymbol{d}h\boldsymbol{d}q} \end{aligned}$$

Implementation of the Bayesian approach as indicated in the previous subsection depends on a willingness to assign probability distributions not only to data variables like $\boldsymbol{Y}$, but also to parameter like q . Typically, these distributions are specified based on information accumulated from past studies, or the opinions of subject-area experts. In choosing a prior belonging to a specific distributional family $p\left(q\,|\,h\right)$, some choices may be more convenient computationally than others. In particular, it may be possible to select a member of that family which is conjugate to the likelihood $f\left(\boldsymbol{y}|q\right)$, that is, one that leads to a posterior distribution $p\ (q|\mathbf{y})$ belonging to the same distributional family as the prior. For example, let Y be a Poisson random variable with likelihood function ,

$$f(y|q) \quad = \quad \frac{e^{-q}q^{y}}{y!} \quad , \quad y = 1,2,\mathbf{K} \quad , \quad q > 0 \ .$$

To apply a Bayesian analysis we require a prior distribution for q having support on the positive real line. A reasonably flexible choice is provided by the Gamma distribution,

$$p(q) \quad = \quad \frac{q^{a-1}e^{\frac{-q}{b}}}{G(a)b^{a}} \quad , \quad q > 0, a > 0, b > 0$$

Using Bayes' Theorem to obtain the posterior density, we have

$$\begin{aligned} p(q|y) \quad &\propto \quad f(y|q)p(q) \\ &\propto \quad (e^{-q}q^{y})(q^{a-1}e^{\frac{-q}{b}}) \\ &= \quad q^{y+a-1}e^{-q(1+\frac{1}{b})} \end{aligned}$$

So the posterior $p(q|y)$ is proportional to Gamma distribution with parameters $a^{/}$ *and* $b^{/}$. The parameters are defined by $a^{/} = y + a$ and $b^{/} = (1+\frac{1}{b})^{-1}$.

2.3.2 Markov chain simulation

With a complicated posterior distribution model, it is difficult to directly sample from the posterior distribution. The Markov chain simulation method will be used for running a Markov chain of simulated values whose stationary distribution provides the target posterior distribution, $p(q|y)$. The idea of Markov chain simulation is to simulate a random walk in the space of q which converges to a stationary distribution that is the joint posterior distribution, $p(q|y)$. There are many clever methods that have been devised for constructing and sampling from transitions for arbitrary posterior distributions. The Metropolis-Hastings algorithm is a general term for a family of Markov chain simulation methods that are useful for drawing samples from Bayesian posterior distributions. There are two commonly –used special cases, the Metropolis algorithm and the Gibbs sampler.

1) The Metropolis algorithm

Given a target distribution $p(q|y)$ that can be computed up to a normalizing constant, the Metropolis algorithm creates a sequence of random points $(q^1, q^2, \mathbf{K})$ whose distributions converge to the target distribution. Each sequence can be considered a random walk whose stationary distribution is $p(q|y)$. The algorithm proceeds as follows.

1. Draw a starting point q^0, for which $p(q^0|y) > 0$, from a starting distribution $p_0(q)$.
2. For t = 1, 2, $\mathbf{K}$

   a) Sample a candidate point q^* from a jumping distribution at time t, $J_t(q^*|q^{t-1})$. The jumping distribution must be symmetric; that is, $J_t(q_a|q_b) = J_t(q_b|q_a)$ for all q_a, q_b, and t.

   b) Calculate the ratio of density,

$$r = \frac{p(q^*|y)}{p(q^{t-1}|y)}$$

   c) Set

$$q^t = \begin{cases} q^* \; with \quad probability \min(r,1) \\ q^{t-1} \; otherwise \end{cases}$$

Given the current value θ^{t-1}, the Markov chain transition distribution, $T_t(q^t|q^{t-1})$, is thus a mixture of the jumping distribution, $J_t(q^t|q^{t-1})$, and a point mass at $q^t = q^{t-1}$. The Metropolis-Hastings algorithm generalizes the basic Metropolis algorithm presented above in two ways. First, the jumping rules J_t need no longer be symmetric; that is, there is no requirement that $J_t(q_a|q_b) \equiv J_t(q_b|q_a)$. Second, to correct for the asymmetry in the jumping rule, the ratio r is replace by a ratio of importance ratios:

$$r = \frac{p(q^*|y) / J_t(q^{t-1})}{p(q^{t-1}|y) / J_t(q^{t-1}|q^*)}$$

$$= \frac{p(q^*|y) \cdot J_t(q^{t-1}|q^*)}{p(q^{t-1}|y) \cdot J_t(q^*|q^{t-1})}$$

2) The Gibbs sampler

Casella & George (1992) illustrate the Gibbs sampler as a method that effectively generates a sample $X_1, \mathbf{K}, X_m \sim f(x)$ without requiring $f(x)$. By simulating a large enough sample, the mean, variance, or any other characteristic of $f(x)$ can be calculated to the desired degree of accuracy. To understand the working of the Gibbs sampler, consider the two-variable case. Starting with a pair of random variables (X, Y), the Gibbs sampler generates a sample from $f(x)$ by sampling instead from the conditional distributions $f(x|y)$ and $f(y|x)$. This is done by generating a "Gibbs sequence" of random variables.

$$Y'_0, X'_0, Y'_1, X'_1, Y'_2, X'_2, \mathbf{K}, Y'_k, X'_k$$

The initial value $Y'_0 = y'_0$ is specified, and the rest of the sequence is obtained iteratively by alternately generating values from

$$X'_j \sim f(x|Y'_j = y'_j)$$
$$Y'_{j+1} \sim f(y|X'_j = x'_j)$$

The distribution of X'_k converges to the true marginal distribution of X as $k \to \infty$. Thus, for k large enough, the final observation, namely $X'_k = x'_k$, is effectively a sample point from $f(x)$. The convergence in the distribution of the Gibbs sequence can be exploited in a variety of ways to obtain an approximate sample from $f(x)$. For example, Gelfand and Smith (1990) suggest generating m independent Gibbs sequences of length k, and then using the final value of X'_k from each sequence, if k is chosen large enough, this yields an approximate iid sample $(X_1, \mathbf{K}, X_m)$ from $f(x)$.

Gibbs sampling can be used to estimate the density itself by averaging the final conditional densities from each Gibb sequence. From the Gibbs sequence, just as the values $X'_k = x'_k$ yield a realization of $X_1, \mathbf{K}, X_m \sim f(x)$, the values $Y'_k = y'_k$ yield a realization of $Y_1, \mathbf{K}, Y_m \sim f(y)$. Moreover, the average of the conditional densities $f(x|Y'_k = y'_k)$ will be a close approximation to $f(x)$, and we can estimate $f(x)$ with

$$\hat{f}(x) = \frac{\sum_{i=1}^{m} f(x|y_i)}{m}$$

In the three variables case we would like to calculate the marginal distribution f(x) in the problem with random variables X, Y and Z. The Gibbs sampler would sample iteratively from

$$
\begin{aligned}
X_j' &\sim f(x \mid Y_j' = y_j', Z_j' = z_j') \\
Y_{j+1}' &\sim f(y \mid X_j' = x_j', Z_j' = z_j') \\
Z_{j+1}' &\sim f(z \mid X_j' = x_j', Y_{j+1}' = y_{j+1}')
\end{aligned}
$$

The iteration scheme as above produces a Gibbs sequence

$$Y_0', Z_0', X_0', Y_1', Z_1', X_1', Y_2', Z_2', \mathbf{K}, Y_k', Z_k', X_k'$$

with the property that, for large k, $\mathrm{X}_\mathrm{k}' = \mathrm{x}_\mathrm{k}'$ is effectively a sample point from f(x).
In fact, a defining characteristic of the Gibbs sampler is that it always uses the full set of univariate conditionals to define the iterative.

On the other hand Carlin & Louis (1996) briefly present, a particular Markov chain algorithm that has been found useful in many multidimensional problems. This is alternating conditional sampling, also called the Gibbs sampler, which is defined in terms of sub-vectors of q. Suppose the parameter vector θ has been divided in to d components or sub-vectors, $\mathrm{q} = (\theta_1, \mathbf{K}, \theta_\mathrm{d})$. Each iteration of the Gibbs sampler cycles through the sub-vectors of q, drawing each subset conditional on the value of all the others. There are thus d steps in iteration t. At each iteration t, an ordering of the d sub-vectors of q is chosen and, in turn, each $\theta_\mathrm{j}^\mathrm{t}$ is sampled from the conditional distribution given all the other components of q:

$$p(q_j^t \mid \; q_{-j}^{t-1}, y)$$

where q_{-j}^{t-1} represents all the components of θ, except for θ_j, at their current values:

$$q_{-j}^{t-1} \quad = (q_1^t, \mathbf{K}, q_{j-1}^t, q_{j+1}^{t-1}, \mathbf{K}, q_d^{t-1})$$

Thus, each sub-vector θ_j is updated conditional on the latest value of q for the other components, which are the iterated t values for components already updated and the iterated t-1 values for the others.

There is, of course, no fully satisfactory method for drawing simulations in general, but the following approach is often successful for simulating from posterior distributions in the hierarchical models that arise in Bayesian statistics.

Step 1. Create an approximate posterior density based on the joint or marginal modes. Draw a sample from the approximate distribution and use iterative sampling to sample about 10 draws of the parameter vector. If approximate distributions are multimodal, several draws are generally needed in the region of each mode that has nontrivial mass.

Step 2. Using these as starting points, run independent parallel sequences of an iterative simulation such as the Gibbs sampler or Metropolis algorithm.

Step 3. Run the iterative simulation until approximate convergence appears to have been reached, in the sense that the statistic $\sqrt{\hat{R}}$ is near 1 for each scalar estimand of interest. This will take hundreds of iterations, at least. Here $\sqrt{\hat{R}}$ is defined below

For each scalar estimand φ, we label the draws from J parallel sequences of length n as j_{ij} (i = 1 , 2 , ... , n ; j = 1 , 2 , ... , j) and we compute B and W, the between and within-sequence variances :

$$B = \frac{n}{j-1}\sum_{j=1}^{J}(\bar{J}_{.j}-\bar{J}_{..})^2 \text{, where } \bar{J}_{.j}=\frac{1}{n}\sum_{i=1}^{n} j_{ij} \quad , \bar{J}_{..}=\frac{1}{j}\sum_{i=1}^{j}\bar{J}_{.j}$$

$$W = \frac{1}{J}\sum_{j=1}^{J} s_j^{\,2} \quad , \quad \text{where} \quad s_j^{\,2}=\frac{1}{n-1}\sum_{i=1}^{n}(j_{ij}-\bar{J}_{.j})^2$$

We can estimate $\mathrm{var}(j\,|y)$, the marginal posterior variance of estimand, by a weighted average of W and B , namely

$$\widehat{\mathrm{var}}^{+}(j\,|y)=\frac{n-1}{n}W+\frac{1}{n}B \quad \text{and} \quad \sqrt{\hat{R}}=\sqrt{\frac{\widehat{\mathrm{var}}^{+}(j\,/\,y)}{W}}$$

Step 4. If $\sqrt{\hat{R}}$ is near 1 for all scalar estimands of interest, summarize inference about the posterior distribution by treating the set of all iterates from the second half of the simulated sequences as an identically distributed sample from the target distribution.

Step 5. Compare the posterior inferences from the Markov chain simulation to the approximate distribution used to start the simulation. If they are not close with respect to locations and approximate distribution shape, check for error before believing that the Markov chain simulation has produced a better answer.

2.3.3 Applied Bayesian approach to infer traffic count on network traffic

Tebaldi & West (1998) study Bayesian inference on network traffic using link count data. The purpose of their study similar to Vardi's (1996) work, was to estimate traffic intensity from source to destination in a network system. The starting point of the study, and assumptions about symbol and network structures, are the same as Vardi's method. But traffic intensity estimation is different.

Consider a fixed network of n nodes, arbitrarily labeled A, B, C,… .
Let a = (i ,j) represent the direct route from originating node i to destination node j. If the direct route has no node between node i and node j , we call it a direct link. There are c = n(n-1) direct routes, and r direct links in the network. Let X_a be the traffic count on the direct route a. Let s = (i, j) represent direct link from node i to node j , and Y_s be the traffic count on the direct link s.Then based on the observed traffic counts on direct link , $\boldsymbol{Y} = (Y_1, Y_2, \ldots, Y_r)'$; we are interested in inferring the traffic count on direct route, $\boldsymbol{X} = (X_1, X_2, \ldots, X_c)'$. Note that the number of direct links r is typically smaller than the number of direct routes c. Following Vardi (1996), $\boldsymbol{Y}$ and $\boldsymbol{X}$ are related through the rxc routing matrix. $\boldsymbol{A} = [A_{s,a}]$, where $A_{s,a} = 1$ if the direct link ‘s’ belongs to the direct route ‘a’ through the net work , and $A_{s,a} = 0$ otherwise . We have the defining identity:

$$\boldsymbol{Y} = \boldsymbol{A}\boldsymbol{X} \tag{2.5}$$

Our goal is to infer $\boldsymbol{X}$ when we know $\boldsymbol{Y}$. To solve this problem we must compute and summarize the posterior distribution $p\left(\boldsymbol{X} \middle| \boldsymbol{Y}\right)$ for all route counts $\boldsymbol{X}$ given the observed link count $\boldsymbol{Y}$ to be tied together with the deterministic expression (2.5) that implies $\boldsymbol{Y}$ given $\boldsymbol{X}$ .This requires a model for the prior distribution, $p\left(\boldsymbol{X}\right)$.

$X_a \sim Poisson(\lambda_a)$ independently over a.

Let the Poisson rate be $\Lambda = \{\lambda_1, \ldots, \lambda_c\}$. The prior specification is completed by a prior for Λ, the starting point for analysis is determining a joint model:

$$p\left(\boldsymbol{X}, \Lambda\right) = p(\Lambda) \cdot \prod_{a=1}^{c} \lambda_a^{X_a} \exp(-\lambda_a) / X_a! \tag{2.6}$$

Given the prior (2.6), the observed link count $\boldsymbol{Y}$ is now conditioned to deliver the required posterior $p\left(\boldsymbol{X}, \Lambda \middle| \boldsymbol{Y}\right)$. Naturally, posterior computations are analytically difficult in any other than trivial and quite unrealistic networks, what is needed are iterative MCMC (Markov Chain Monte Carlo) simulation methods. Consider in particular Gibbs sampling, in which we iteratively resample from conditional posteriors for elements of the $\boldsymbol{X}$ and Λ variables.

First, consider simulation of L . We note that

$$p\left(L|\boldsymbol{X},\boldsymbol{Y}\right)\equiv p\left(L|\boldsymbol{X}\right)=\prod_{\boldsymbol{a}=1}^{\boldsymbol{c}} \quad p\left(l_a|X_a\right)$$

The components consist of the form of the prior density $p(l_a)$ multiplied by the gamma form arising in the Poisson-based likelihood function. Thus by employing conditional $\boldsymbol{X}$, we can easily simulate new L values as a set of independent draws from the implied univariate posterior. If $p(l_a)$ is gamma, or a mixture of gammas, then these draws are trivially made from the corresponding gamma or mixture gamma posteriors.

Now we try to simulate $\boldsymbol{X}$ based on the conditional posterior $\boldsymbol{p}\left(\boldsymbol{X}|L,\boldsymbol{Y}\right)$, viewing L as fixed. Our data $\boldsymbol{Y}$ are in form of linear constraints , $\boldsymbol{Y}=\boldsymbol{AX}$ on the route count vector $\boldsymbol{X}$, so that conditioning must be performed directly, algebraically, rather than via the usual application of Bayes' theorem. On the other hand we do not need to simulate X_i for i = 1, 2, …, c , but only simulate X_i for i = r+1, r+2, …, c via the usual application of Bayes' theorem then directly evaluate X_i for i = 1, 2, …, r based on algebra . The following result, which is simply an algebraic deduction from the network structure and defined relation (2.5) among the traffic counts, is the key to ensuring inferential development.

Tebaldi & West (1998) prove that, in the network model $\boldsymbol{Y}=\boldsymbol{AX}$, if $\boldsymbol{A}$ is of full rank r. then we can reorder the columns of $\boldsymbol{A}$ so that the revised routing matrix has the form

$$\boldsymbol{A}=[\boldsymbol{A}_1,\boldsymbol{A}_2] \tag{2.7}$$

where $\boldsymbol{A}_1$ is a nonsingular rxr matrix. Also, similarly reordering the elements of the $\boldsymbol{X}$ vector and conformably partitioning as $\boldsymbol{X}^{/}=[\boldsymbol{X}_1^{/},\boldsymbol{X}_2^{/}]$ it follows that

$$\boldsymbol{X}_1=\boldsymbol{A}_1^{-1}\left(\boldsymbol{Y}-\boldsymbol{A}_2\boldsymbol{X}_2\right) \tag{2.8}$$

From the result of the theorem, the posterior $p\left(\boldsymbol{X}|L,\boldsymbol{Y}\right)$ is concentrated in a subspace of dimension c-r defined by the partition (2.7) of the routing matrix. Having reordered the column of $\boldsymbol{A}$ to the form (2.7), this posterior has the form

$$p\left(\boldsymbol{X}|L,\boldsymbol{Y}\right)= p\left(\boldsymbol{X}_1|\boldsymbol{X}_2,L,\boldsymbol{Y}\right)p\left(\boldsymbol{X}_2|L,\boldsymbol{Y}\right)$$

where $p\left(\boldsymbol{X}_1|\boldsymbol{X}_2,L,\boldsymbol{Y}\right)$ is degenerate at $\boldsymbol{X}_1=\boldsymbol{A}_1^{-1}\left(\boldsymbol{Y}-\boldsymbol{A}_2\boldsymbol{X}_2\right)$ and with

$$\boldsymbol{X}_2 \;=\; (X_{r+1},\mathbf{K},X_c)^{/}$$

$\boldsymbol{X}_1 \;=\; (X_1, \mathbf{K}, X_r)'$ are defined as earlier

and $$p\left(\boldsymbol{X}_2 \middle| L, \boldsymbol{Y}\right) \propto \prod_{a=1}^{c} \frac{l_a^{X_a}}{X_a!} \qquad (2.9)$$

which is over the support defined by $X_a \geq 0$ for a = 1, 2, $\mathbf{K}$, c . This is simply the expression of product of independent Poisson priors for the X_i constrained by the identity (2.5) rewritten in the form (2.8). The utility of this expression is in delivering the set of complete conditional posteriors for elements of the $\boldsymbol{X}_2$ vector to form part of the iterative simulation approach to posterior analysis. Consider each elements X_i of $\boldsymbol{X}_2$ (i = r+1 ,…,c) and write $\boldsymbol{X}_{2,-i}$ for the remaining elements. Then, simply by inspection of (9) we see that the conditional distribution $p\ (X_i| \ \ \boldsymbol{X}_{2,-i}, L, \boldsymbol{Y})$ is

$$p\ (X_i| \ \ \boldsymbol{X}_{2,-i}, L, \boldsymbol{Y}) \quad \propto \quad \frac{l_i^{X_i}}{X_i!} \prod_{a=1}^{r} \frac{l_a^{X_a}}{X_a!} \qquad (2.10)$$

That is over the support defined by $X_i \geq 0$ and $X_a \geq 0$ for each a = r+1 , …, c; this holds for each i = r+1, $\mathbf{K}$, c.

Identifying the support of (2.10) requires the study of the linear constraints on X_i defined by $X_a \geq 0$ for all elements X_a of $\boldsymbol{X}_1 = \boldsymbol{A}_1^{-1}\left(\boldsymbol{Y} - \boldsymbol{A}_2 \boldsymbol{X}_2\right)$. Given i in r+1,…,c, this implies a set of linear constraints as functions of the conditioning values of $\boldsymbol{X}_{2,-i}$ and $\boldsymbol{Y}$. The resulting constraints are the form of $X_i \geq \boldsymbol{d}_i$ or $X_i \leq \boldsymbol{e}_i$,where the values $\boldsymbol{d}_i$ and e_i are functions of the conditioning value of $\boldsymbol{X}_{2,-i}$ and $\boldsymbol{Y}$. Hence, together with $X_i \geq 0$, we obtain a set of at most r+1 constraints on X_i . By directly evaluating these constraints and identifying their intersection, we may deduce the range of X_i over which (2.10) is nonzero, and hence we identify the unnormalized conditional posterior distribution.

Iterative simulation of full posterior $p\left(\boldsymbol{X}, L \middle| \boldsymbol{Y}\right)$ is now enabled as follow:

Step 1. Fix starting values of the route counts **X**

Step 2. Draw sample value of the rate $L \;=\; \left\{l_1, \mathbf{K}, l_c\right\}$ from c conditionally independent posterior distributions

$$p\left(L \middle| \boldsymbol{X}, \boldsymbol{Y}\right) \equiv p\left(L \middle| \boldsymbol{X}\right) = \prod_{a=1}^{c} \quad p\left(l_a \middle| X_a\right)$$

where $p\left(l_a \middle| X_a\right)$ is gamma distributions that for l_a having shape parameter $X_a + 1$ and scale parameter 1

Step 3. Condition these values of L , simulate a new $\boldsymbol{X}$ vector by sequencing through i = r+1, r+2, … , c ., and at each step sample a new X_i from

$$p\ (X_i \mid \ \boldsymbol{X}_{2,-i}, L, \boldsymbol{Y}) \quad \propto \quad \frac{l_i^{X_i}}{X_i!}\prod_{a=1}^{r}\frac{l_a^{X_a}}{X_a!}$$

with conditioning elements $\boldsymbol{X}_{2,-i}$ set at their most recent sampled values.

Step 4. Reevaluate each step $\boldsymbol{X}_1$ based upon step 3 as follow:

$\boldsymbol{X}_1 = \boldsymbol{A}_1^{-1}(\boldsymbol{Y} - \boldsymbol{A}_2\boldsymbol{X}_2)$ as a function of most recently sampled elements of $\boldsymbol{X}_2$

Step 5. Return to step 2 and iterate.

The sampling step in step 3 appears to require evaluation of the support (10). Sampling may be performed directly, treating (10) as a simple multinomial distribution on this relevant range. Indirect but very much more efficient simulation methods are based on embedding Metropolis-Hastings steps within the Gibbs sampling framework. Here the candidate value of the X_i is generated at each stage from suitable proposed distributions such the uniform distribution, and accepted or rejected according to the usual Metropolis-Hastings acceptance probabilities. Specifically, we assume a specified and fixed proposal distribution with probability mass function $q_i(X_i)$ for each element X_i in step 3 . A candidate value X_i^* is drawn from $q_i(\cdot)$ and accepted with probability

$$\min\left[1, \frac{p_i(X_i^*)q_i(X_i)}{p_i(X_i)q_i(X_i^*)}\right]$$

where X_i is the current, most recently sampled value and $p_i(\cdot)$ is the unnormalized conditional posterior in equation (2.10). From the structure of network equations in (2.5), it is possible to identify bounds on each X_i so that a suitable range for the proposal distribution can be computed. For element X_a given $X_{2,-a}$, X_a of $\boldsymbol{X}_2$, X_a = 0 is a gross lower bound whatever the values in $X_{2,-a}$. For an upper bound, $X_a \le \min_i \left\{ Y_i - \sum_{j \ne a} A_{ij} X_j \right\}$, where the index i run over the set of links whose counts include X_a; that is, those links i for $A_{ij} = 1$. Then, based on the specified bounds, the implied vector $\boldsymbol{X}_1$ is recomputed and checked for feasibility; that is, nonnegative value. If any element of $\boldsymbol{X}_1$ is negative, the trial value of X_a is either incremented, in searching for the lower bound on its range, or decremented, in searching for the upper bound. This process terminates and delivers the resulting bounds once the $\boldsymbol{X}_1$ vector has r nonnegative entries.

2.4 The traffic intensities estimation based on a mixture of maximum likelihood and Bayesian inference

This section proposes a new method to estimate traffic intensities. The method uses maximum likelihood estimation to estimate the parameters, the mean population of traffic intensity on direct routes. Then let the estimators and the observed count on direct links to infer the unobserved traffic count on direct routes bases on Bayesian inference.

Let traffic count notation following Vardi (1997), be as follows:

l : mean population vector on direct route.

$l = [l_1, l_2, \ldots, l_c]$

$\boldsymbol{X}$: Traffic intensities vector on direct route.

$\boldsymbol{X} = (X_1, X_2, \mathbf{K}, X_c)$, $X_i \sim Poisson(l_i)$

$\boldsymbol{Y}$: Traffic intensities vector on direct link.

$\boldsymbol{Y} = (Y_1, Y_2, \mathbf{K}, Y_r)$

$\boldsymbol{A}$: Routing matrix.

$\boldsymbol{Y}^{(k)}$: Traffic intensities vector on direct link at measurement period K.

$\boldsymbol{Y}^{(k)} = (Y_1^{(k)}, Y_2^{(k)}, \mathbf{K}, Y_r^{(k)})$

$\overline{\boldsymbol{Y}} = (\overline{Y}_1, \overline{Y}_2, \mathbf{K}, \overline{Y}_r)$

$$\overline{Y}_i = \frac{\sum_{k=1}^{K} Y_i^{(k)}}{K}$$

The equation that presents the relation between $\boldsymbol{X}$ and $\boldsymbol{Y}$ is:

$$\boldsymbol{Y} = \boldsymbol{AX}$$

Expected value of the equation is $\overline{\boldsymbol{Y}} = \boldsymbol{A}\,l$

The canonical form of the EM iteration for solving the equation is

$$l_j \leftarrow \frac{l_j}{\sum_{i=1}^{r} a_{ij}} \sum_{i=1}^{r} \frac{a_{ij}\overline{Y}_i}{\sum_{k=1}^{c} a_{ik}\, l_k} \qquad ; j = 1,2,\mathbf{K}, c$$

Our goal is to infer $\boldsymbol{X}$ given l and $\boldsymbol{Y}$ based on the posterior distribution of $\boldsymbol{X}$ given l and $\boldsymbol{Y}$, $p\left(\boldsymbol{x}\middle|l,\boldsymbol{y}\right)$. Here π is the prior distribution function of $\boldsymbol{X}$.

Consider;
$$\begin{aligned} p\left(\boldsymbol{x}\middle|l,\boldsymbol{y}\right) &\propto f\left(l,\boldsymbol{y}\middle|\boldsymbol{x}\right)p\left(\boldsymbol{x}\right) \\ &= f_2\left(\boldsymbol{y}\middle|\boldsymbol{x}\right) f_1\left(l\middle|\boldsymbol{x}\right)p\left(\boldsymbol{x}\right) \\ &\propto f_2\left(\boldsymbol{y}\middle|\boldsymbol{x}\right) p\left(\boldsymbol{x}\middle|l\right) \end{aligned}$$

where $f_2\left(\boldsymbol{y}\middle|\boldsymbol{x}\right)$ is degenerate at $\boldsymbol{Y} = \boldsymbol{AX}$ and

$$p\left(\boldsymbol{x}\middle|l\right) \ \alpha \ \prod_{i=1}^{c}\frac{l_i^{x_i}}{x_i!}$$

In conclusion, the mixed method to infer $\boldsymbol{X}$ given $\boldsymbol{Y}$ and l is firstly to estimate l based on EM iteration. Then use Marcov Chain simulation and the Gibb sampling algorithm to obtain $\boldsymbol{X}$ from $p\left(\boldsymbol{x}\middle|l\right)$. Finally evaluate $\boldsymbol{Y}$ by the equation

$$\boldsymbol{Y} = \boldsymbol{AX}$$

Iterative simulation of full posterior $p\left(\boldsymbol{x}\middle|l,\boldsymbol{y}\right)$ adapted from Vardi (1996) and Tebaldi&West (1998) is now possible as follow:

Step 1 Let $l = [l_1, l_2,..., l_c]$ be the daily transmission rate

Step 2 Generate daily data on direct links for day K

$$\boldsymbol{Y}^{(1)} = \left(Y_1^{(1)}, Y_2^{(1)}, \mathbf{K}, Y_r^{(1)}\right)$$
$$\boldsymbol{Y}^{(2)} = \left(Y_1^{(2)}, Y_2^{(2)}, \mathbf{K}, Y_r^{(2)}\right)$$
$$\boldsymbol{Y}^{(3)} = \left(Y_1^{(3)}, Y_2^{(3)}, \mathbf{K}, Y_r^{(3)}\right)$$
$$\mathrm{M}$$
$$\boldsymbol{Y}^{(K)} = \left(Y_1^{(K)}, Y_2^{(K)}, \mathbf{K}, Y_r^{(K)}\right)$$

Calculate $$\overline{Y}_i = \frac{\sum_{k=1}^{K} Y_i^{(k)}}{K}$$

Step 3 Estimate $\hat{l} = (\hat{l}_1, \hat{l}_2,..., \hat{l}_c)$ based on applied algorithm

$$l_j \leftarrow \frac{l_j}{\sum_{i=1}^{r} a_{ij}} \sum_{i=1}^{r} \frac{a_{ij}\overline{Y}_i}{\sum_{k=1}^{c} a_{ik}\, l_k} \quad ; \quad j = 1,2,\mathbf{K},c$$

Step 4 Go to step 2 to estimate $\hat{l}$ m times so we derive

$$\hat{l}^{(1)}, \hat{l}^{(2)},..., \hat{l}^{(m)}$$

Step 5 Calculate mean vector $\bar{\hat{l}} = \frac{\sum_{k=1}^{m} \hat{l}^{(k)}}{m}$

Step 6 Draw starting values of the route counts $\boldsymbol{X}$ from the Poisson distribution with the parameter from step 5

Step 7 Draw sample value of rate $l = (l_1, l_2, ..., l_c)$ from c conditionally independent posterior distribution

$$p\left(l|\boldsymbol{x}\right) = \prod_{i=1}^{c} p(l_i|x_i)$$

where $p(l_i|x_i)$ is gamma distribution with shape parameter $x_i + 1$ and scale parameter 1

Step 8 Conditioning on these value of l simulate new $\boldsymbol{X}$ vector by sequencing through i = 1,2,…,c. and at each step sampling new X_i from

$$p\left(x_i|l\right) \propto \frac{l_i}{x_i!} \prod_{a=1}^{c} \frac{l_a^{x_a}}{x_a!}$$

Step 9 Base on step 8 at each step Y is evaluated via

$$\boldsymbol{Y} = \boldsymbol{AX}$$

Step 10 Return to step 7 and iterate.

2.5 Queuing system theory

Gorney (1979) is a useful source for queuing for giving theory terminology. There are four general types of queue: single facility single queue systems, single queue multi facility systems, multi queue single facility systems and multi queue multi facility systems. Ament (1980) applies queuing theory to bank service to the benefits of both bank customers and personnel. The benefits consist of decreased customer throughput time, better use of all existing equipment, improved customer relations, and reduction of teller numbers. Ross & Shanthikumar (2005) study a modem bank with two streams of arriving customers. Drekic & Woolford (2005) analyze a singer-server preemptive priority queuing model with low priority balking customers. Fakinos (1982) provides the limiting probability distribution for the number of customers waiting in single server queue and for customers arrival.

Zhu & Zhang (2004) consider a queue model with two types of customers that consist of positive and negative customers. The management of supply chains and manufacturing systems is an important issue. Liu, Liu, & Yao (2004) look at of the inventory cost. Queuing theory is used to develop an efficient procedure to minimize the overall inventory. Kerbache & Smith (2004) develop a queuing system for the supply chain in manufacturing firms. The study shows that the approach is a very useful tool to analyze congestion problems and to evaluate the performance of the network. Yang, Lee, Chen & Chen (2005) propose a queuing network model for machine time interference. Sarkar & Zangwill (1992) study a cyclic queue system that has one server and n nodes, where each node has its own distinct type of customers that arrive from the outside. The study extends to permit special nodes.

Aquilar-Iqartua, Postiqo-Boix, & Garcia-Haro (2002) apply queuing theory to a high speed network. Brown, Gans, Mandelbaum & Sakov (2005) develop queuing for a call center in which agents provide telephone-based services, to decease delay in telephone queues.

Cruz, MacGreqor & Queiroz (2005) analyze queuing and develop algorithms to compute the optimal capacity allocation in a service system. Halachmi (1978) utilizes the technique of embedded Markov chains for queuing systems.
Chen (2004) develops performance measures in finite capacity queuing by using fuzzy logic that is widely used in finite capacity queuing models. Maqlaras & Mieqhem (2005) present an approach based on a fluid-model to control a multi product queuing system. The benefit of the approach is construction of scheduling and multi-product admission policies for lead time control. Takine (2005) applies a continuous-time Markov chain for single server queues with several customer classes.

Das & Levinson (2004) use queuing analysis to treat traffic flow parameters such as flow, density and speed. Their study area is on Interstate 94 in the Minneapolis St. Paul metro. In addition Omari, Masaeid & Shawabkah (2004)
develop a delay model based on data selection that comes from different cities in Jordan. The study show that the random arrivals, random services, and a single service channel queuing delay model (M/M/1) is also validated using the field delay data, and it was found that it estimates delay with high variability, especially for high delay ranges. Fu, Hu, & Naqi (1995) apply two techniques, perturbation analysis and the likelihood ratio method, to a single queue system with non identical multiple servers in a traffic system. Rolls, Michailidis, & Hernandez-Campos (2005) apply several queuing metrics

to provide a network traffic trace through trace-driven queuing. Cheng & Allam (1992) present knowledge of the delay and queuing processes of vehicles that pass along minor road to deliver timing that is suitable for traffic flow for traffic controlled intersections. Cruz, Smith & Medeiros (2005) develops a discrete-event digital simulation model to study performance of queuing in traffic flow. The study shows that the simulation model is an effective and insightful tool. Mahmoud & Araby (1999) develop a dynamic macroscopic traffic simulation model to respond to high-density and low-density traffic flows. Dewees (1979) develop a traffic simulation model to produce new estimates of congestion costs on specific streets during the morning rush hour. Ellis & Durgee (1982) present an engineering approach for Voice network designers to decide whether queuing or route-advance or forcing user retrials are appropriate selections for a particular network. Nam & Drew (1998) use the principle of traffic dynamic analyze freeway traffic flows. They use the fundamental concept of conservation to analysis queuing and discharging mechanisms.

Kleinrock (1976) presents the essence of queuing theory as of the characterization of the arrival time, the service time and the evaluation of their effect on queuing phenomena . Additionally, Vivanichkool (1995) extends the knowledge of queuing using a queuing system consisting of: customers who are waiting in queue and customers who are receiving service. The number of elements at any time in the system are the number of customers in the queue plus the number of customers being serviced. The characteristics of queue models are : interarrival time distribution, service time distribution, number of servers, service regulation and maximum elements that the system permits.

2.5.1 Notation in queuing

The important notation used in the queuing system are as follows:

n : the number of elements in the system,

$p_n(t)$: probability that the transient system has n elements at time t based on the assumption that the system starts at t = 0,

p_n : probability that the steady system has n elements,

l : rate of arrival, number of elements that arrive at the system per unit of time,

m : rate of departure, number of elements that depart the system per a unit of time,

C : number of servers,

r : utilization factor , $r = \frac{l}{m}$, $0 \leq r < 1$

$\frac{r}{C}$: utility factor of C servers,

$W(t)$: probability density distribution function (pdf) for wait time,

W_s : wait time for an element in the system,

W_q : wait time for an element in the queue,

L_s : expected number of elements in the system, and

L_q : expected number of element in the queue.

The relation between W_s, W_q, L_s and L_q can be shown by equations as:

$$L_s = l W_S ,$$

$$L_q = l W_q ,$$

$$W_q = W_s - \frac{1}{m} ,$$

$$l W_q = l W_s - \frac{l}{m} ,$$

$$L_q = L_s - r .$$

2.5.2 Arrival distribution

Based on the assumption that the arrival rate is l per a unit of time and that there are no elements in the system at time $t = 0$, the probabilities $p_n(t)$ and $p_n(t+h)$, the probable change of the system between time t and t+h falls in to two cases as follows:

Case 1. For $n > 0$, there are n elements in the system at $t + h$ if ,

a) there are n elements at time t and no element arrives in length h or,

b) there are $n - 1$ elements at time t and there is one element arriving in length h.

Case 2. For $n = 0$, there are no elements at time t and time t+h and there are no elements in length h.

Based on the two cases equations are derived as follows:

$$p_n(t+h) \cong p_n(t)(1-lh) + p_{n-1}(t)\,lh \qquad for \quad n > 0$$

$$p_0(t+h) \cong p_0(t)(1-lh)$$

So $$\frac{p_n(t+h)-p_n(t)}{h} \cong -l p_n(t) + l p_{n-1}(t)$$

and $$\frac{p_0(t+h)-p_0(t)}{h} \cong -l p_0(t)$$

Let limit h trend to 0

$$\lim_{h\to 0} \frac{p_n(t+h)-p_n(t)}{h} = -l p_n(t) + l p_{n-1}(t)$$

$$\lim_{h\to 0} \frac{p_0(t+h)-p_0(t)}{h} = -l p_0(t)$$

that is $$\frac{d}{dt} p_n(t) = -l p_n(t) + l p_{n-1}(t)$$

and $$\frac{d}{dt} p_0(t) = -l p_0(t)$$

$\therefore$ $$p_n(t) = \frac{(lt)^n e^{-lt}}{n!} \qquad n = 0,1,2,\mathbf{K}$$

The proof above illustrates that the arrival distribution is the Poisson distribution with mean lt and variance lt.

2.5.3 Interarrival time distribution

Interarrival time is the interval time between two sequent arrivals. Let the arrival distribution be a Poisson distribution. Interarrival time distribution will be considered as follows:

Let $f(t)$, $t > 0$ be inter-arrival time distribution function, and

$F(t)$ be cumulative distribution function of $f(t)$, so

$$F(t) = \int f(m)dm$$

No element arrives in interval (0,t); this means that the inter-arrival time is longer than t, that is

$$p_0(t) = \int_t^{\infty} f(u)du$$

$$= 1 - \int_0^t f(m)dm$$

$$= 1 - F(t)$$

Q $\quad p_0(t) = e^{-lt}$

$\therefore \quad e^{-lt} = 1 - F(t)$

Differentiating $F(t)$ by t, derive $f(t)$ as follows:

$$f(x) = \begin{cases} l e^{-lt} & t > 0 \\ 0 & t \le 0 \end{cases}$$

The proof above illustrates that the interarrival time distribution is exponential with mean $\frac{1}{l}$ and variance $\frac{1}{l^2}$

2.5.4 Departure distribution

Based on the assumption that: there are N element in the system at time $t = 0$ and there are no element arrival at the system, rate of departure is m per a unit of time. Probability of no element departing the system is equal to $1 - mh$, so

$$p_n(t+h) \cong p_n(t)(1-mh) \qquad ; \quad n = N$$

$$p_n(1+h) \cong p_n(t)(1-mh) + p_{n+1}(t)mh \qquad ; \quad 0 < n < N$$

$$p_0(1+h) \cong p_0(t) \cdot 1 + p_1(t)mh \qquad ; \quad n = 0$$

$$\lim_{h \to 0} \frac{p_n(t+h) - p_n(t)}{h} = -mp_n(t) \qquad ; \quad n = N$$

$$\lim_{h \to 0} \frac{p_n(t+h) - p_n(t)}{h} = -mp_n(t) + mp_{n+1}(t) \qquad ; \quad 0 < n < N$$

$$\lim_{h \to 0} \frac{p_0(t+h) - p_0(t)}{h} = -mp_0(t) \qquad ; \quad n = 0$$

So

$$\frac{d}{dt} p_n(t) = -mp_n(t) \qquad ; \quad n = N$$

$$\frac{d}{dt} p_n(t) = -mp_n(t) + mp_{n+1}(t) \qquad ; \quad 0 < n < N$$

$$\frac{d}{dt} p_0(t) = -mp_1(t) \qquad ; \quad n = 0$$

The result from the equations above are as follows:

$$p_n(t) = \frac{(mt)^{N-n} e^{-mt}}{(N-n)!} \qquad ; \; n = 1, 2, \ldots, N$$

$$p_0(t) = 1 - \sum_{n=1}^{N} p_n(t)$$

This illustrates that the departure distribution is a truncated Poisson distribution.

2.5.5 Service time distribution

Let $g(t)$ be the probability distribution function of service time, notice that probability of no service in interval time, $(0,T)$ will equal the probability of no element departing the system at the same time, so

$$P\text{(service time } t > T \text{) = P(no element depart system between } T\text{)}$$

or
$$1 - \int_0^T g(t)dt = P_N(T) = e^{-mT}$$

Therefore
$$\int_0^T g(t)dt = 1 - e^{-mT}$$

Differentiation of both two sides of equation gives:

$$g(t) = \begin{cases} me^{-ut} & t > 0 \\ 0 & t \le 0 \end{cases}$$

This illustrates that the service time distribution is exponential with mean $\frac{1}{m}$ and variance $\frac{1}{m^2}$.

2.5.6 Queuing model

Let A/B/S denote the queuing model that consists of S servers, interarrival time distribution A and service time distribution B. Particular choices of A and B are as follows:

M : Exponential distribution,

E_r : r-stage Erlangian distribution,

H_R : R-stage Hyperexponential distribution,

D : Deterministic distribution, and

G : General

An important queuing model is described in the next section.

1) The M/M/1 Queue

The characteristic of this model are :

1) Interarrival time distribution is exponential ;

2) Service time distribution is exponential ;

3) There is only one server;

4) Service regulation is first come, first served; and

5) Indefinite number of elements.

Probability of n>0 in the system at time t+h is approximated by the summation of probabilities as follows:

1) The probability of n elements in the system at time t, and no element arrival, and no element departing in length h, is approximated by

$$p_n(t)\{(1-lh)(1-mh)\}$$

2) The probability of n elements in the system at time t, and no element arrival, and one element departing in length h, is approximated by

$$p_n(t)\{(lh)(mh)\}$$

3) The probability of n-1 elements in the system at time t, and one element arrival, and no element departing in length h, is approximated by,

$$p_{n-1}(t)\{(lh)(1-mh)\}$$

4) The probability of n+1 elements in the system at time t, and no element arrival, and one element departing in length h, is approximated by,

$$p_{n+1}(t)(1-lh)(mh)$$

So $p_n(t+h) \cong p_n(t)\{(1-lh)(1-mh)\} + p_n(t)\{(lh)(mh)\} + p_n(t)\{(lh)(mh)\} +$

$$p_{n-1}(t)\{(lh)(1-mh)\}$$

Since h^2 converges to zero,

$$p_n(t+h) \cong p_n(t)\{1-lh-mh\} + p_{n-1}(t)(lh) + p_{n+1}(t)(mh)$$

In the same way when n = 0

$$p_0(t+h) \cong p_0(t)\{(1-lh)\cdot 1\} + p_1(t)(mh)(1-lh) = p_0(t)(1-lh) + p_1(t)(mh)$$

$$\lim_{h\to 0}\frac{p_n(t+h)-p_n(t)}{h} = lp_{n-1}(t) + mp_{n+1}(t) - (l+m)p_n(t) \quad ; \qquad n>0$$

$$\lim_{h\to 0}\frac{p_0(t+h)-p_0(t)}{h} = -lp_0(t) + mp_1(t) \qquad ; \quad n=0$$

So $$\frac{d}{dt} p_n(t) = l p_{n-1}(t) + m p_{n+1}(t) - (l + m) p_n(t)$$

$$\frac{d}{dt} p_0(t) = - l p_0(t) + m p_1(t)$$

For steady system, $t \to \infty$ when $l < m$, that is

$$r = \frac{l}{m} < 1$$

When $t \to \infty$, $\quad \frac{d}{dt} p_n(t) \to 0 \quad$ *and* $\quad p_n(t) \to p_n \ , \ n = 0,1,2,\mathbf{K}$

$$- l p_0 + m p_1 = 0 \qquad ; \quad n = 0$$

$$l p_{n-1} + m p_{n+1} - (l + m) p_n = 0 \qquad ; \quad n > 0$$

The difference equation results in the target distribution as follows:

$$p_n = (1 - r) r^n \quad ; \quad n = 0,1,2,\mathbf{K}$$

The distribution is a geometric distribution with mean and variance as follows:

$$E(n) = \frac{r}{1 - r}$$

$$Var(n) = \frac{r}{(1 - r)^2}$$

The geometric mean illustrates the important characteristic of queuing system as follows:

$$L_S = E(n) = \frac{r}{1 - r}$$

$$L_q = L_S - \frac{l}{m} = \frac{r^2}{1 - r}$$

$$W_S = \frac{L_S}{l} = \frac{1}{m(1 - r)}$$

$$W_q = \frac{L_q}{l} = \frac{r}{m(1 - r)}$$

$$P_0 = 1 - \frac{l}{m}$$

$$P_n = P_0 \left(\frac{l}{m} \right)^n$$

2) The M/G/1 Queue

The characteristic of this model is composed of :

1) Interarrival time distribution which is Exponential distribution;
2) Service time distribution that is general distribution;
3) Only one capacity;
4) Service regulation which is first come, first served; and
5) Indefinite number of element.

In this case we need to know mean and variance of departing distribution, assume that the mean is equal m , the variance is equal s^2. Mean of service time is equal $\frac{1}{m}$ and variance of service time is equal s^2. The important characteristic of queue system is as follows:

$$r = \frac{l}{m} < 1$$

$$P_0 = 1 - r$$

$$L_q = \frac{l^2 r^2 + r^2}{2(1 - r)}$$

$$W_q = \frac{L_q}{l}$$

$$W_q = W_q + \frac{1}{m}$$

3) The M/M/S Queue

The characteristic of this model is composed of :

1) Interarrival distribution which is Exponential distribution;
2) Service time distribution which is Exponential distribution with mean $\frac{1}{m}$;
3) m servers;
4) Service regulation which is first come, first served; and
5) Indefinite number of element.

Assume that there are S service capacities, and each capacity has one server. Service rate of each capacity is equal m, so the mean of all capacities is equal $m_n = nm$ when $n \le S$, if $n \ge S$ and all capacities are maximum service, $\mu_n = S\mu$ and $l_n = l$.

$$m_n = \begin{cases} nm & 0 \le n \le S \\ Sm & n \ge S \end{cases}$$

$$n = 0,1,2,\mathbf{K}$$

Since $l > Sm$, so means of arrival rate is less than the maximum of service rate.

$$P_0 = \frac{1}{\sum_{n=0}^{S-1} \frac{\left(\frac{l}{m}\right)^n}{n!} + \frac{\left(\frac{l}{m}\right)^S}{S!} \cdot \frac{1}{1 - \frac{l}{Sm}}}$$

$$P_n = \begin{cases} \frac{\left(\frac{l}{m}\right)}{n!} \cdot P_0 & ,0 \le n \le S \\ \frac{\left(\frac{l}{m}\right)^n}{S! S^{n-S}} \cdot P_0 & ,n \ge S \end{cases}$$

$$r = \frac{l}{mS}$$

$$L_q = \frac{P_0 \left(\frac{l}{m}\right)^S r}{S!(1-r)^2}$$

$$W_q = \frac{L_q}{l}$$

$$W_s = W_q + \frac{1}{m}$$

$$L_s = l(W_q + \frac{1}{m}) = L_q + \frac{l}{m}$$

4) G/G/1 for The heavy –traffic approximation

Kleinrock (1976) applied the G/G/1 queue for the heavy-traffic approximation when $\rho \cong 1$. The wait time distribution is an approximation exponential distribution with the mean given as follows:

$$W(y) \cong 1 - \exp\left(-\frac{2\bar{t}(1-r)}{s_a^2 + s_b^2} \cdot y \right)$$

$$W \cong \frac{\left(s_a^2 + s_b^2\right)}{2(1-r)\bar{t}}$$

where $\quad r = \frac{l}{m}; \quad \bar{t} = \frac{1}{l},$

s_a^2 : variance of interarrival time; and

s_b^2 : variance of service time

2.6 Queuing generation

Consider a queuing system (Banks, Carson, Nelsun, &Nicol ,2001) over a period of time T , and $L(t)$ denote the number of customers in the system at time t.

Let T_i denote the total time during $[0,T]$ in which the system contained exactly i customers.

We can estimate the number of customers in the system over a period of time T at any time t by $\hat{L}$, the time-weighted-average number.

$$\hat{L} = \frac{\sum_{i=1}^{\infty} iT_i}{T}$$

Since the total area under the function $L(t)$ can be decomposed into rectangles of height i and length T_i

$$\hat{L} = \frac{1}{T}\int_0^T L(t)dt$$

$$\rightarrow L \quad \text{as} \quad T \rightarrow \infty$$

Here L is the long-run time-average number in the system.

$L_Q(t)$ denotes the number of customers waiting in line(queue)

T_i^Q denotes the total time during $[0,T]$ in which exactly i customers are waiting in the queue.

We can estimate the number of customers waiting in the queue from time 0 to time T by $\hat{L}_Q$, the observed time-average number of customers waiting in the queue as follows:

$$
\begin{aligned}
\hat{L}_Q = & \sum_{i=0}^{\infty} i T_i^Q \\
= & \frac{1}{T}\int_0^T L_Q(t)dt \\
& \rightarrow L_Q \quad \text{as} \quad T \rightarrow \infty
\end{aligned}
$$

Here L_Q is the long-run time-average number of customers waiting in the queue.

In queuing simulation over a period of time T, we can record W_i, the wait time that customer i spends in the system during $[0,T]$, for $i = 1,2,\mathbf{K},N$. The average time spent in the system per customer is called the average system time. The formula to compute average system time is given by :

$$\hat{W} = \frac{\sum_{i=1}^{N} W_i}{N}$$

For a stable system, as $N \rightarrow \infty$

$$\hat{W} \rightarrow W$$

Here W is called the long-run average system time.

In addition, we specially consider the time that customer i spends in the queue. Let W_i^Q denote the total time that customer i spends waiting in the queue. We can compute the observed average time is spent in the queue (called delay) by the formula:

$$
\begin{aligned}
\hat{W}_Q = & \frac{\sum_{i=1}^{N} W_i^Q}{N} \\
& \rightarrow W_Q \quad , \quad \text{as} \quad N \rightarrow \infty
\end{aligned}
$$

Here W_Q is the long-run average per customer.

2.7 The evaluation function

The evaluation of the effectiveness of the traffic control at the intersection is generally based on the delay or wait time which is known as the ‘wait mean’. It is obtained by the calculation of the combined time of each car spent on its wait time at the red light divided by the total number waiting cars. As a consequence, the longer the wait mean the less effective is the traffic control. However, wait mean should not be the only indicator to judge the effectiveness of the traffic control; the number of cars moving in and out of the intersection including drive mean should also be taken into account to evaluate the effectiveness. This is supported by the model of Kelsey & Bisset (1993) presenting a cost function that consist of such factors to evaluate traffic flow performance. The value of the function will be used to evaluate the performance of traffic flow under fuzzy controller against the conventional controller. The lower the cost function the better the performance.

$$\text{Cost} = \frac{Wait_{mean}}{100 \cdot \left(\frac{Car_{out}}{Car_{in}} \right) Drive_{mean}}$$

$Wait_{mean}$: The average waiting time in seconds that all cars spend behind the red light.

$Drive_{mean}$: The average time in seconds that all cars spend behind the green light.

Car_{out} : The number of cars that are exiting the intersection.

Car_{in} : The number of cars that are entering the intersection.

Chapter 3

Research Methodology

3.1 The conceptual research

To calculate the optimal length of the traffic signal on each phase of the cycle, firstly we need to estimate traffic intensity that arrives and departs at the intersections and the length of the current cycle time based on statistical methods. These estimators are crisp inputs for fuzzy logic control. Then crisp outputs are produced by using the process of fuzzy logic control. The crisp output is the degree of traffic signal change for each phase. Finally the optimal length of traffic signal is the period of time between the connective change points. This concept can be conceptualized as shown below:

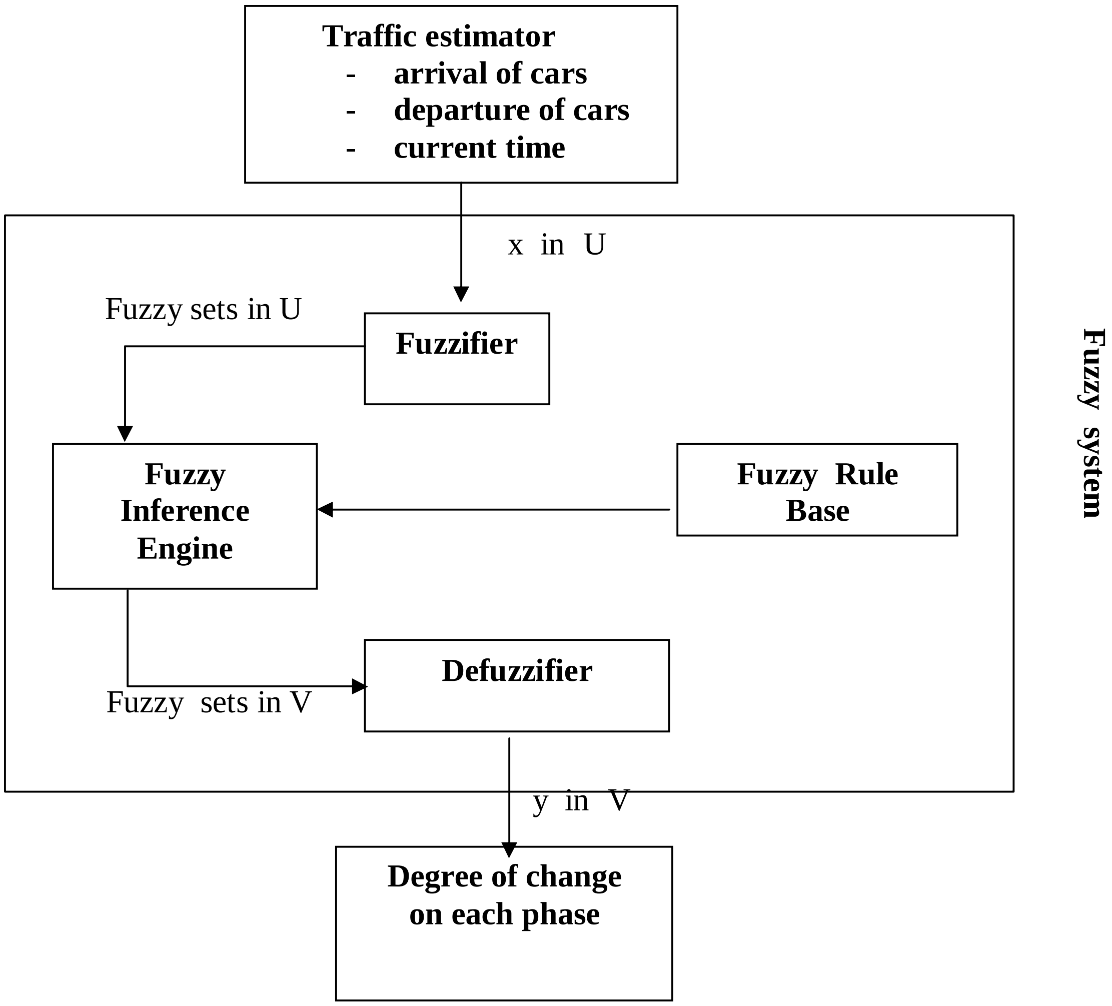


Figure 3.1 Conceptual map(Adapted from Wang, 1994, p. 6)

3.2 The input process methodology

There are three important inputs consisting of: the number of cars passing the green light, the number of cars stopping behind the red light and length of the current cycle time. To estimate the value of these inputs, we need to study traffic at the actual intersections, and use statistical methods to estimate the number of cars and the length of current cycle time.

3.2.1 Traffic control at actual intersections studied

1) Traffic network studied

The optimal traffic signal light time was studied at four important intersections in the inner city of Ubon Rachathani Province consisting of : Uboncharearnsri , Clock Hall, Chonlaprathan , and Airport intersections. The network diagram representing the four intersections is shown as Figure 3.2

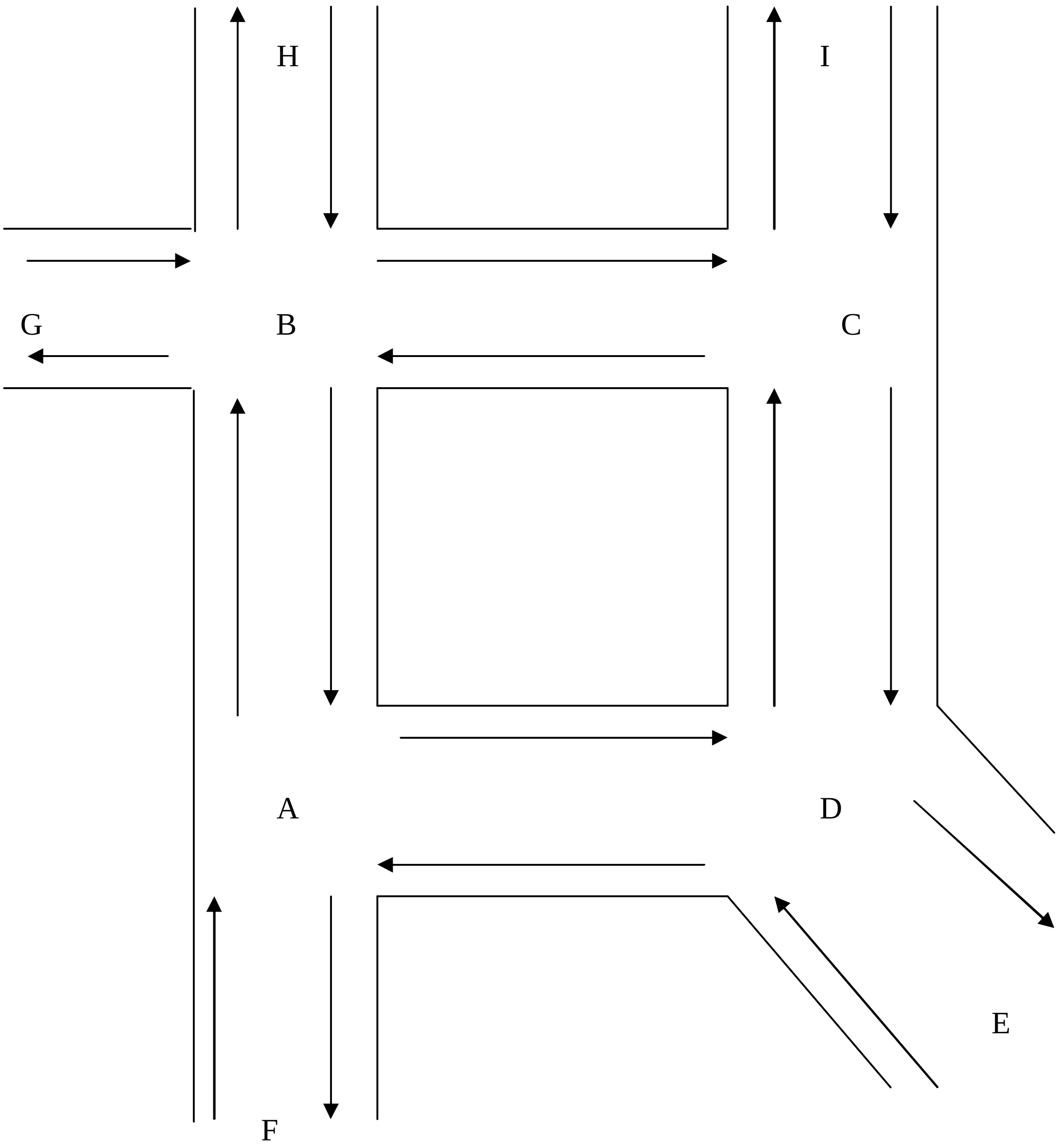


Figure 3.2 Diagram of traffic network consisting of the four intersections A, B, C and D with car flow from E, F,G, H and I

According to Vardi's notation (1996), there are 72 source-destination pairs (SD), made up of 54 direct routes and 18 direct links.

The 54 direct routes are as follows;

$AC \equiv\ A \rightarrow D \rightarrow C$	$AE \equiv\ A \rightarrow D \rightarrow E$	$AG \equiv\ A \rightarrow B \rightarrow G$
$AH \equiv\ A \rightarrow B \rightarrow H$	$AI \equiv\ A \rightarrow D \rightarrow C \rightarrow I$	$CA \equiv\ C \rightarrow B \rightarrow A$
$EA \equiv\ E \rightarrow D \rightarrow A$	$GA \equiv\ G \rightarrow B \rightarrow A$	$HA \equiv\ H \rightarrow B \rightarrow A$
$IA \equiv\ I \rightarrow C \rightarrow D \rightarrow A$	$ED \equiv\ B \rightarrow C \rightarrow D$	$BE \equiv\ B \rightarrow C \rightarrow D \rightarrow E$
$BF \equiv\ B \rightarrow A \rightarrow F$	$BI \equiv\ B \rightarrow C \rightarrow I$	$DB \equiv\ D \rightarrow A \rightarrow B$
$EB \equiv\ E \rightarrow D \rightarrow A \rightarrow B$	$FB \equiv\ F \rightarrow A \rightarrow B$	$IB \equiv\ I \rightarrow C \rightarrow B$
$CE \equiv\ C \rightarrow D \rightarrow E$	$CF \equiv\ C \rightarrow B \rightarrow A \rightarrow F$	$CG \equiv\ C \rightarrow B \rightarrow G$
$CH \equiv\ C \rightarrow B \rightarrow H$	$EC \equiv\ E \rightarrow D \rightarrow C$	$FC \equiv\ F \rightarrow A \rightarrow D \rightarrow C$
$GC \equiv\ G \rightarrow B \rightarrow C$	$HC \equiv\ H \rightarrow B \rightarrow C$	$DF \equiv\ D \rightarrow A \rightarrow F$
$DG \equiv\ D \rightarrow C \rightarrow B \rightarrow G$	$DH \equiv\ D \rightarrow C \rightarrow B \rightarrow H$	$DI \equiv\ D \rightarrow C \rightarrow I$
$FD \equiv\ F \rightarrow A \rightarrow D$	$GD \equiv\ G \rightarrow B \rightarrow C \rightarrow D$	$HD \equiv\ H \rightarrow B \rightarrow A \rightarrow D$
$ID \equiv\ I \rightarrow C \rightarrow D$	$EF \equiv\ E \rightarrow D \rightarrow A \rightarrow F$	$FG \equiv\ F \rightarrow A \rightarrow B \rightarrow G$
$IE \equiv\ I \rightarrow C \rightarrow D \rightarrow E$	$FH \equiv\ F \rightarrow A \rightarrow B \rightarrow H$	$GF \equiv\ G \rightarrow B \rightarrow A \rightarrow F$
$HF \equiv\ H \rightarrow B \rightarrow A \rightarrow F$	$IF \equiv\ I \rightarrow C \rightarrow D \rightarrow A \rightarrow F$	$GH \equiv\ G \rightarrow B \rightarrow H$
$GI \equiv\ G \rightarrow B \rightarrow C \rightarrow I$	$HG \equiv\ H \rightarrow B \rightarrow G$	$IG \equiv\ I \rightarrow C \rightarrow B \rightarrow G$
$HI \equiv\ H \rightarrow B \rightarrow C \rightarrow I$	$IH \equiv\ I \rightarrow C \rightarrow B \rightarrow H$	$FE \equiv\ F \rightarrow A \rightarrow D \rightarrow E$
	$EI \equiv\ E \rightarrow D \rightarrow C \rightarrow I$	
	$EH \equiv\ E \rightarrow D \rightarrow A \rightarrow B \rightarrow H$	
	$GE \equiv\ G \rightarrow B \rightarrow C \rightarrow D \rightarrow E$	
	$HE \equiv\ H \rightarrow B \rightarrow C \rightarrow D \rightarrow E$	
	$FI \equiv\ F \rightarrow A \rightarrow D \rightarrow C \rightarrow I$	
	$EG \equiv\ E \rightarrow D \rightarrow A \rightarrow B \rightarrow G$	

The 18 direct links are as follows:

$AB \equiv \; A \rightarrow B$	$BA \equiv \; B \rightarrow A$	$BC \equiv \; B \rightarrow C$
$CB \equiv \; C \rightarrow B$	$CD \equiv \; C \rightarrow D$	$DC \equiv \; D \rightarrow C$
$AD \equiv \; A \rightarrow D$	$DA \equiv \; D \rightarrow A$	$AF \equiv \; A \rightarrow F$
$FA \equiv \; F \rightarrow A$	$DE \equiv \; D \rightarrow E$	$ED \equiv \; E \rightarrow D$
$CI \equiv \; C \rightarrow I$	$IC \equiv \; I \rightarrow C$	$BH \equiv \; B \rightarrow H$
$HB \equiv \; H \rightarrow B$	$BG \equiv \; B \rightarrow G$	$GB \equiv \; G \rightarrow B$

2) Flow phase of each intersection studied

Flow phase refers to the time length of the green lights which allows the cars to directly move toward their targeted directions. The phase is in fact counted from the end of the red light and the start of the ember light. This means that phase stands between the red and the ember light. Each intersection has different phase form.

The next subsection will present the phase at each intersection by a diagram.

Let $\longrightarrow$ represent cars that pass the green light

$- - - \rightarrow$ represent cars that stop behind the red light

The diagrams presenting the phases at each intersection are as follows:

2.1) The form of flow phase at Uboncharearnsri intersection

There are three phases at Uboncharearnsri intersection.

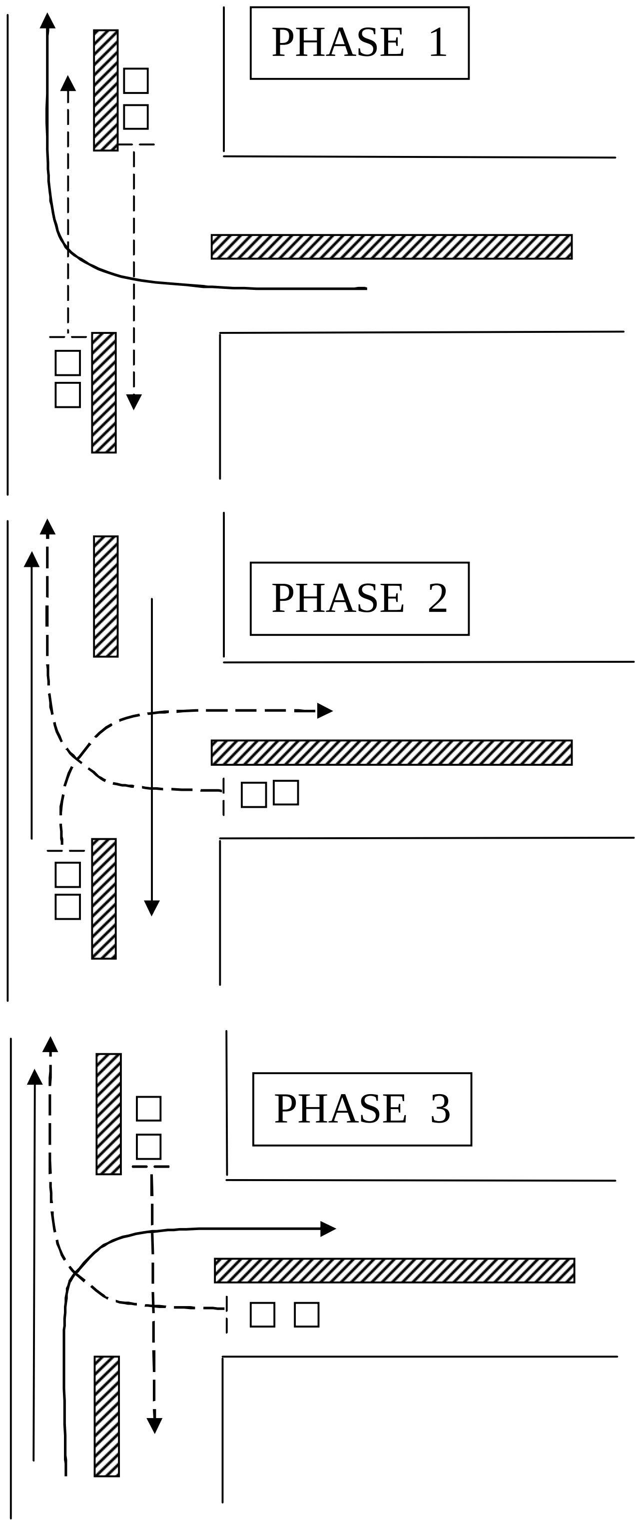


Figure 3.3 Diagram to present the flow phases at Uboncharearnsri intersection.

2.2) The form of flow phases at Clock Hall intersection

There are three phases at Clock Hall intersection.

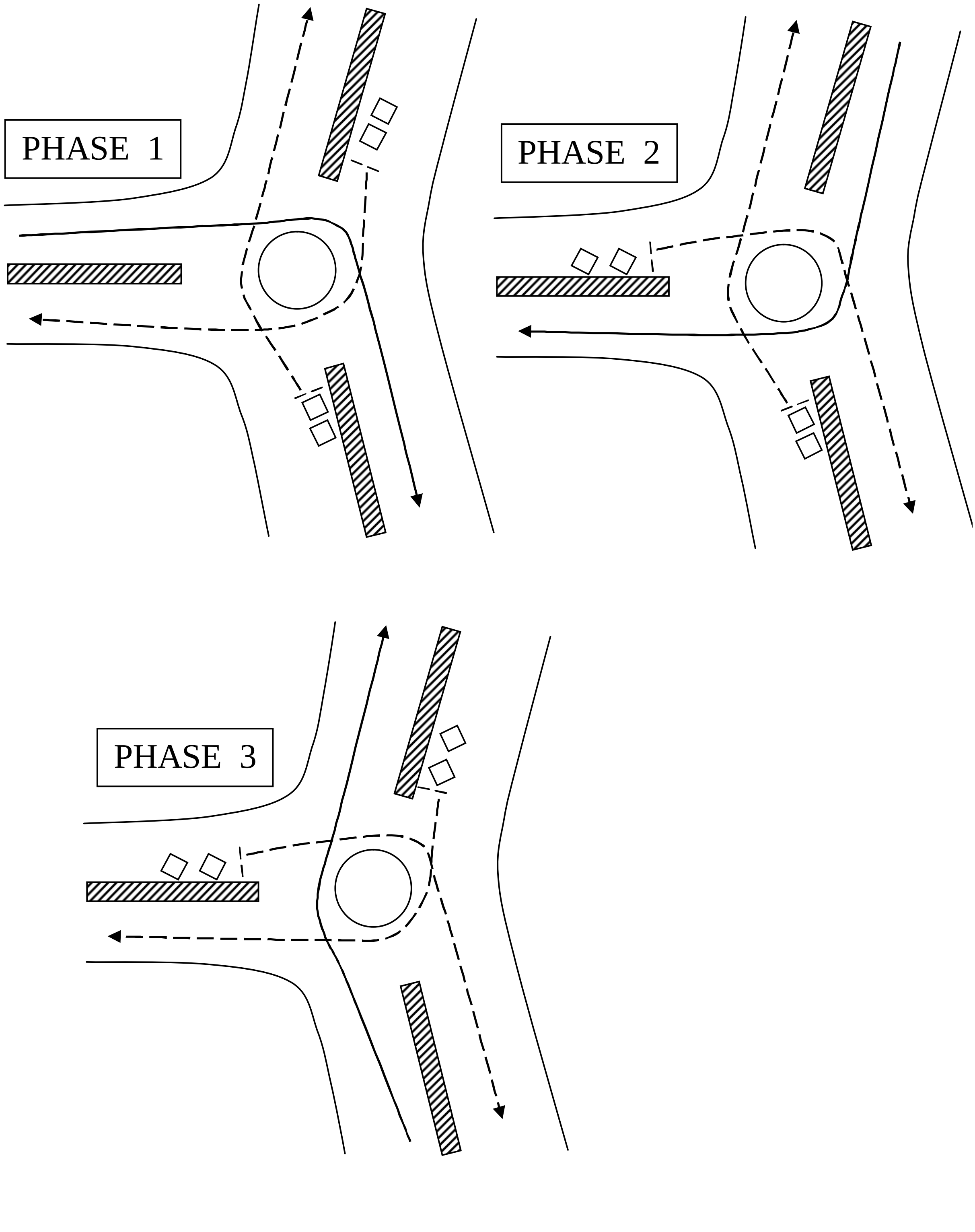


Figure 3.4 Diagram to present the flow phases at Clock Hall intersection.

2.3) The form of the flow phases at Chonraprathan intersection

There are three phases at Chonraprathan intersection.

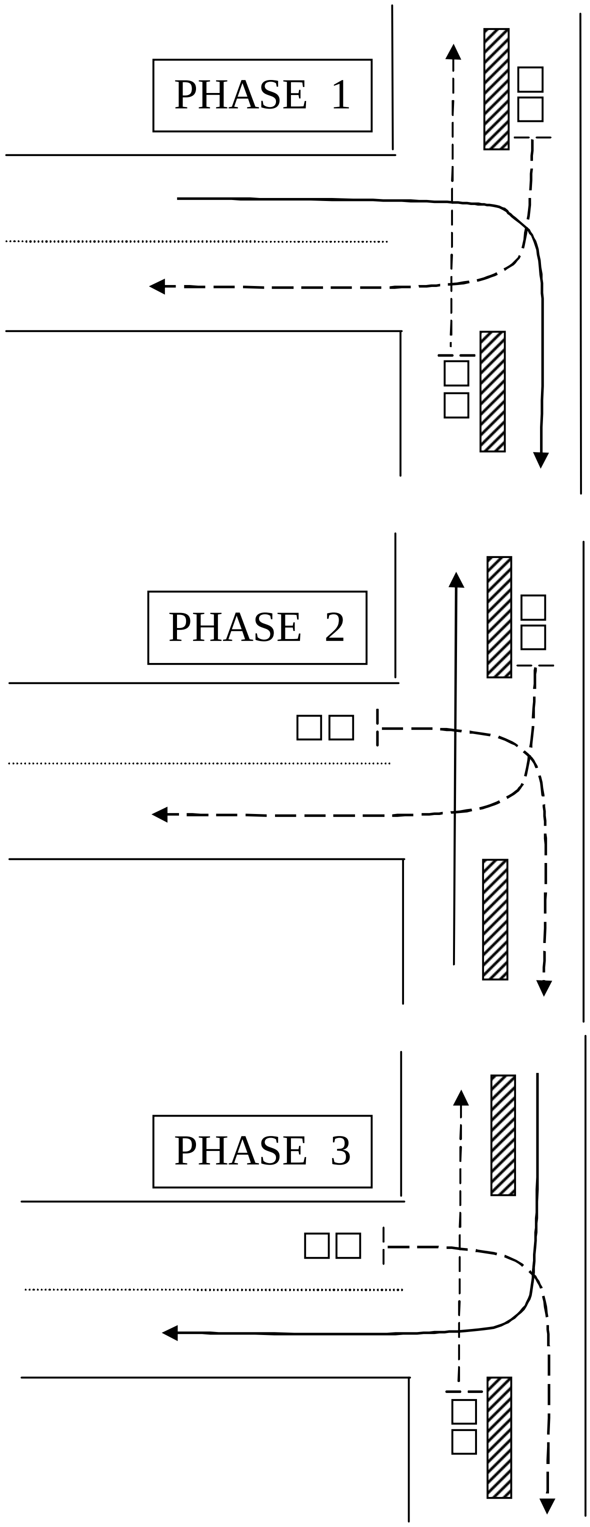


Figure 3.5 Diagram to presenting the flow phases at Chonlaprathan intersection

2.3) The form of the flow phases at Airport intersection

There are four phases at Airport intersection :

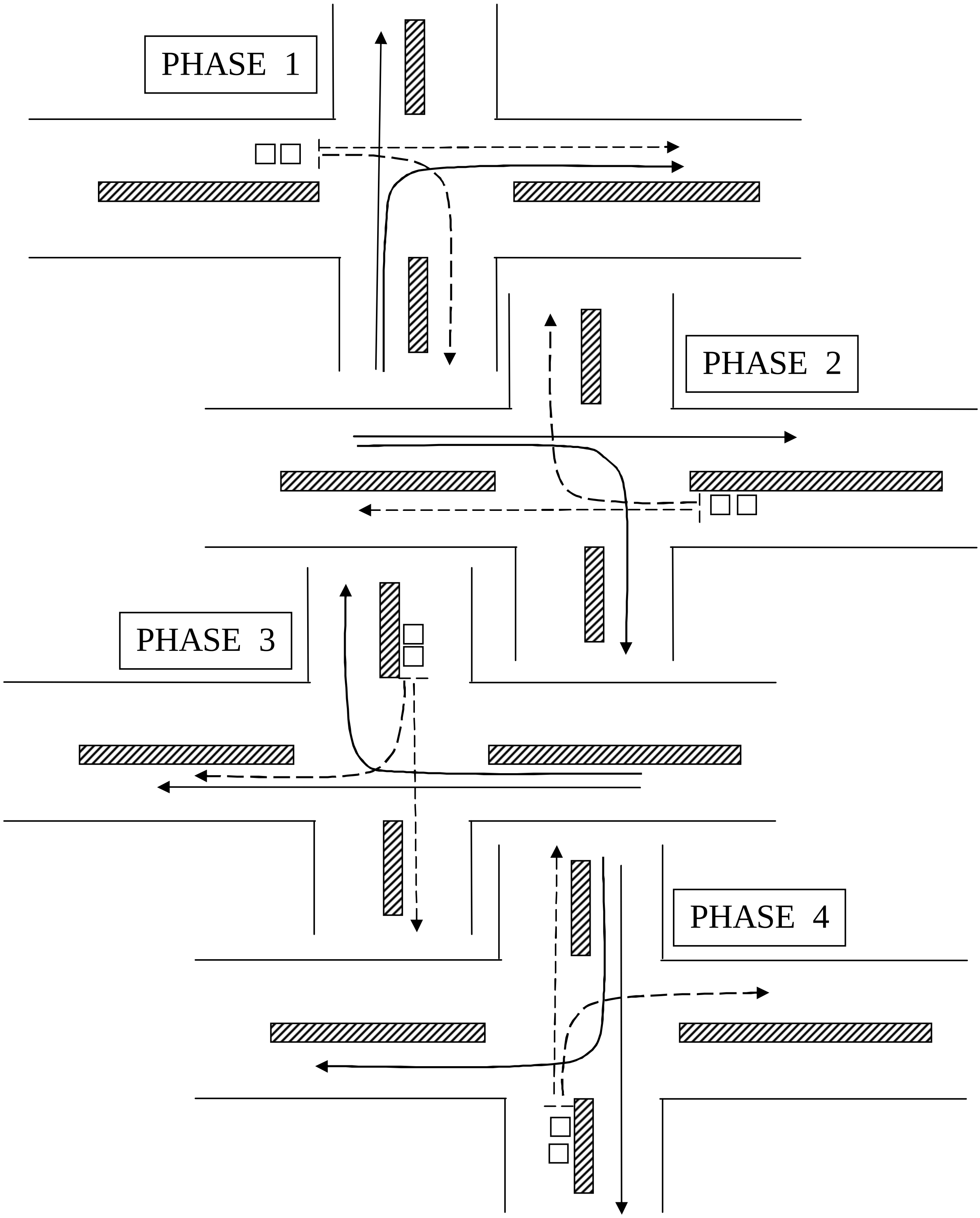


Figure 3.6 Diagram to present the flow phases at Airport intersection

3.2.2 Traffic estimation by using mix models

This section presents the statistical method used to estimate the number of cars that depart from an intersection to other intersections according to a mixture of maximum likelihood estimation and Bayesian inference. This section also explains the method used to compute the length of current cycle time.

3.2.2.1 Notation used

Let X_j denote the route count belonging to a direct route , or the number of cars that depart from specified sources to destination, for $j = 1, 2, 3, \ldots, 72$. The details of each X_j are as follows:

X_1 : the number of cars from source A to destination B

X_2 : the number of cars from source A to destination C

X_3 : the number of cars from source A to destination D

X_4 : the number of cars from source A to destination E

X_5 : the number of cars from source A to destination F

X_6 : the number of cars from source A to destination G

X_7 : the number of cars from source A to destination H

X_8 : the number of cars from source A to destination I

X_9 : the number of cars from source B to destination A

X_{10} : the number of cars from source C to destination A

X_{11} : the number of cars from source D to destination A

X_{12} : the number of cars from source E to destination A

X_{13} : the number of cars from source F to destination A

X_{14} : the number of cars from source G to destination A

X_{15} : the number of cars from source H to destination A

X_{16} : the number of cars from source I to destination A

X_{17} : the number of cars from source B to destination C

X_{18} : the number of cars from source B to destination D

X_{19} : the number of cars from source B to destination E

X_{20} : the number of cars from source B to destination F

X_{21} : the number of cars from source B to destination G

X_{22} : the number of cars from source B to destination H

X_{23} : the number of cars from source B to destination I

X_{24} : the number of cars from source C to destination B

X_{25} : the number of cars from source D to destination B

X_{26} : the number of cars from source E to destination B

X_{27} : the number of cars from source F to destination B

X_{28} : the number of cars from source G to destination B

X_{29} : the number of cars from source H to destination B

X_{30} : the number of cars from source I to destination B

X_{31} : the number of cars from source C to destination D

X_{32} : the number of cars from source C to destination B

X_{33} : the number of cars from source C to destination F

X_{34} : the number of cars from source C to destination G

X_{35} : the number of cars from source C to destination H

X_{36} : the number of cars from source C to destination I

X_{37} : the number of cars from source D to destination C

X_{38} : the number of cars from source E to destination C

X_{39} : the number of cars from source F to destination C

X_{40} : the number of cars from source G to destination C

X_{41} : the number of cars from source H to destination C

X_{42} : the number of cars from source I to destination C

X_{43} : the number of cars from source D to destination E

X_{44} : the number of cars from source D to destination F

X_{45} : the number of cars from source D to destination G

X_{46} : the number of cars from source D to destination H

X_{47} : the number of cars from source D to destination I

X_{48} : the number of cars from source E to destination D

X_{49} : the number of cars from source F to destination D

X_{50} : the number of cars from source G to destination D

X_{51} : the number of cars from source H to destination D

X_{52} : the number of cars from source I to destination D

X_{53} : the number of cars from source E to destination F

X_{54} : the number of cars from source G to destination E

X_{55} : the number of cars from source H to destination E

X_{56} : the number of cars from source I to destination E

X_{57} : the number of cars from source F to destination G

X_{58} : the number of cars from source F to destination H

X_{59} : the number of cars from source F to destination I

X_{60} : the number of cars from source G to destination F

X_{61} : the number of cars from source H to destination F

X_{62} : the number of cars from source I to destination F

X_{63} : the number of cars from source G to destination H

X_{64} : the number of cars from source G to destination I

X_{65} : the number of cars from source H to destination G

X_{66} : the number of cars from source I to destination G

X_{67} : the number of cars from source H to destination I

X_{68} : the number of cars from source I to destination H

X_{69} : the number of cars from source E to destination G

X_{70} : the number of cars from source E to destination H

X_{71} : the number of cars from source E to destination I

X_{72} : the number of cars from source F to destination E

Let $\boldsymbol{X}$ denote the direct route count matrix, $\boldsymbol{X}$ is the row matrix with dimension 1X72 as follows:

$$\boldsymbol{X} = \left[X_1, X_2, \mathbf{K}, X_{72}\right]$$

Let Y_i denote the route count corresponding to direct link, or the number of cars that depart from the source to destination, for $i = 1, 2, 3, \ldots, 18$. The details of each Y_i are as follows:

Y_1 : the number of cars from source A to destination B

Y_2 : the number of cars from source B to destination A

Y_3 : the number of cars from source B to destination C

Y_4 : the number of cars from source C to destination B

Y_5 : the number of cars from source C to destination D

Y_6 : the number of cars from source D to destination C

Y_7 : the number of cars from source A to destination D

Y_8 : the number of cars from source D to destination A

Y_9 : the number of cars from source A to destination F

Y_{10} : the number of cars from source F to destination A

Y_{11} : the number of cars from source D to destination E

Y_{12} : the number of cars from source E to destination D

Y_{13} : the number of cars from source C to destination I

Y_{14} : the number of cars from source I to destination C

Y_{15} : the number of cars from source B to destination H

Y_{16} : the number of cars from source H to destination B

Y_{17} : the number of cars from source B to destination G

Y_{18} : the number of cars from source G to destination B

Let $\boldsymbol{Y}$ denote the direct link count matrix, $\boldsymbol{Y}$ is the row matrix with dimension 1X18 as follows:

$$\boldsymbol{Y} = \left[Y_1, Y_2, \mathbf{K}, Y_{18}\right]$$

Let l_j denote the population mean of the number of cars that depart from source to destination, for j = 1, 2, 3, …, 72 .

Let l denote the population mean route count matrix with dimension 1×72 as follows

$$l = [\, l_1, l_2, \mathbf{K}, l_{72}\,]$$

3.2.2.2 Estimation of route count mean based on the EM

This section presents the statistical method to estimate the route count mean based on the EM algorithm. Observe Y_i at period k in the actual situation, and

Let $X_j^{(k)}$ denote the number of cars for direct route j at measurement period k . We assume that

$X_j^{(k)} \sim$ Poisson (m_j) ; $j = 1,2,\mathbf{K}\,72$. $k = 1,2,\mathbf{K},K$ is independent.

$\boldsymbol{X}^{(\boldsymbol{k})}$ is the number of cars in vector form for direct route.

$$\boldsymbol{X}^{(\boldsymbol{k})} = \left[X_1^{(k)}, X_2^{(k)}, \mathbf{K}, X_{72}^{(k)}\right]^{/}$$

$Y_i^{(k)}$ is the number of cars that are observed from direct link i at measurement period k.

$\boldsymbol{Y}^{(\boldsymbol{k})}$ is the number of cars in vector form for direct links.

$$\boldsymbol{Y}^{(\boldsymbol{k})} = \left[Y_1^{(k)}, Y_2^{(k)}, \mathbf{K}, Y_{18}^{(k)}\right]^{/}$$

Let $\boldsymbol{A}$ denote the 18x72 routing matrix for this network. The matrix $\boldsymbol{A}$ is a zero-one matrix whose rows correspond to the direct link, its columns correspond to direct routes, and its entries, a_{ij} are 1 or 0 according to whether link i does or does not belong to the direct path of the SD pair j.

So matrix $\boldsymbol{A}$ is defined by $\boldsymbol{A} = [a_{ij}]$

a_{ij} = 1; for (i, j) = (1,1),(1,6),(1,7),(1,25),(1,26),(1,27),(1,57),(1,58),(1,69),(1,70) (2,9),(2,10),(2,14),(2,15),(2,20),(2,33),(2,51),(2,60),(2,61)(3,17),(3,18),(3,19),(3,23), (3,40),(3,41),(3,54),(3,55),(3,64),(3,67),(4,10),(4,24),(4,4,30),(4,33),(4,34),(4,35), (4,46),(4,66),(4,68),(5,16),(5,18),(5,19),(5,31),(5,32),(5,50),(5,52),(5,54),(5,55),(5,56)(5 ,62)(6,2),(6,8),(6,37),(6,38),(6,39),(6,45),(6,46),(6,47),(6,59),(6,71)(7,2),(7,3),(7,4),(7,8) ,(7,39),(7,49),(7,51),(7,59),(7,72),(8,11),(8,12),(8,16),(8,25),(8,26),(8,44),(8,53), (8,62),(8,69),(8,70),(9,5),(9,20),(9,33),(9,44),(9,53),(9,60),(9,61),(9,62)(10,13), (10,27),(10,39),(10,49),(10,57),(10,58),(10,59),(10,72)(11,4),(11,19),(11,32),(11,43), (11,55),(11,56),(11,72)(12,12),(12,26),(12,38),(12,53),(12,68),(12,69),(12,70),(12,71), (13,8),(13,23),(13,36),(13,47),(13,59),(13,64),(13,67),(13,71)(14,16),(14,30),(14,42), (14,52),(14,56),(14,62),(14,66),(14,68),(15,7),(15,22),(15,35),(15,46),(16,58),(15,63), (15,68),(15,70),(16,15),(16,29),(16,41),(16,51),(16,55),(16,61),(16,65),(16,67)(17,6), (17,21),(17,34),(17,57),(17,65),(17,66),(17,69),(18,14),(18,28),(18,40),(18,54),(18,60)(1 8,63),(18,64)

a_{ij} = 0; for the other (i, j)

So we derive the relation between $\boldsymbol{Y}^{(\boldsymbol{k})}$ and $\boldsymbol{X}^{(\boldsymbol{k})}$ in equation form as follow:

$$\boldsymbol{Y}^{(\boldsymbol{k})} = \boldsymbol{A}\boldsymbol{X}^{(\boldsymbol{k})} \qquad ,k = 1,2,\mathbf{K},K$$

From the matrix form we can write 18 equations that present Y_i and X_j as follow:

$Y_1 = X_1+X_6+X_7+X_{25}+X_{26}+X_{27}+X_{57}+X_{58}+X_{69}+X_{70}$

$Y_2 = X_9+X_{10}+X_{14}+X_{15}+X_{20}+X_{33}+X_{51}+X_{60}+X_{61}$

$Y_3 = X_{17}+X_{18}+X_{19}+X_{23}+X_{40}+X_{41}+X_{54}+X_{55}+X_{64}+X_{67}$

$Y_4 = X_{10}+X_{24}+X_{30}+X_{33}+X_{34}+X_{35}+X_{46}+X_{66}+X_{68}$

$Y_5 = X_{16}+X_{18}+X_{19}+X_{31}+X_{32}+X_{50}+X_{52}+X_{54}+X_{55}+X_{56}+X_{62}$

$Y_6 = X_2+X_8+X_{37}+X_{38}+X_{39}+X_{45}+X_{46}+X_{47}+X_{59}+X_{71}$

$Y_7 = X_2+X_3+X_4+X_8+X_{39}+X_{49}+X_{51}+X_{59}+X_{72}$

$Y_8 = X_{11}+X_{12}+X_{16}+X_{25}+X_{26}+X_{44}+X_{53}+X_{62}+X_{69}+X_{70}$

$Y_9 = X_5+X_{20}+X_{33}+X_{44}+X_{53}+X_{66}+X_{61}+X_{62}$

$Y_{10} = X_{13}+X_{27}+X_{39}+X_{49}+X_{57}+X_{58}+X_{59}+X_{72}$

$Y_{11} = X_4+X_{19}+X_{32}+X_{43}+X_{55}+X_{56}+X_{72}$

$Y_{12} = X_{12}+X_{26}+X_{38}+X_{53}+X_{68}+X_{69}+X_{70}+X_{71}$

$Y_{13} = X_8+X_{23}+X_{36}+X_{47}+X_{59}+X_{64}+X_{67}+X_{71}$

$$\begin{aligned}
Y_{14} &= X_{16}+ X_{30}+X_{42}+X_{52}+X_{56}+X_{56}+X_{62}+X_{66}+X_{68}\\
Y_{15} &= X_{7}+X_{22}+X_{35}+X_{46}+X_{58}+X_{63}+X_{68}+X_{70}\\
Y_{16} &= X_{15}+X_{29}+X_{41}+X_{51}+X_{55}+X_{61}+X_{65}+X_{67}\\
Y_{17} &= X_{6}+X_{21}+X_{34}+X_{57}+X_{65}+X_{66}+X_{69}\\
Y_{18} &= X_{14}+X_{28}+X_{40}+X_{54}+X_{60}+X_{63}+X_{64}
\end{aligned}$$

Our goal is to estimate $m = (m_1, m_2, \ldots, m_{72})$ from $\boldsymbol{Y}^{(1)}, \boldsymbol{Y}^{(2)}, \ldots, \boldsymbol{Y}^{(K)}$ based on maximum likelihood estimation and sample moments using the following 7 steps.

Step 1 Let positive refer to the population of number of car passing direct route on traffic network

$$m = (m_1, m_2, \ldots, m_{72}) \qquad \text{; arbitrary.}$$

Step 2 Observe daily data on direct links for 20 days from 08:00 – 08:30 am

$$\begin{aligned}
\boldsymbol{Y}^{(1)} &\equiv \left(Y_1^{(1)}, Y_2^{(1)}, \ldots, Y_{18}^{(1)}\right)\\
\boldsymbol{Y}^{(2)} &\equiv \left(Y_1^{(2)}, Y_2^{(2)}, \ldots, Y_{18}^{(2)}\right)\\
&\vdots \qquad\qquad \vdots\\
\boldsymbol{Y}^{(20)} &\equiv \left(Y_1^{(20)}, Y_2^{(20)}, \ldots, Y_{18}^{(20)}\right)
\end{aligned}$$

Calculate $\qquad \bar{Y}_i = \dfrac{\sum_{k=1}^{20} Y_i^{(k)}}{20}$

Step 3 Estimate m with $\hat{m} = (\hat{m}_1, \hat{m}_2, \ldots, \hat{m}_{72})$ based on applied algorithm

$$m_j \leftarrow \frac{m_j}{\sum_{i=1}^{18} a_{ij}} \sum_{i=1}^{18} \frac{a_{ij}\bar{Y}_i}{\sum_{k=1}^{72} a_{ik} m_k}$$

Step 4 Generate X_j from Poisson distribution with the estimator of parameter

$$\hat{m}_j \text{;} \quad j = 1, 2, \ldots, 72 \quad \text{for 100 days}$$

Step 5 Generate daily data on direct links for 100 days depending on X_j in step 4

$$\begin{aligned}
\boldsymbol{Y}^{(1)} &\equiv \left(Y_1^{(1)}, Y_2^{(1)}, \ldots, Y_{18}^{(1)}\right)\\
\boldsymbol{Y}^{(2)} &\equiv \left(Y_1^{(2)}, Y_2^{(2)}, \ldots, Y_{18}^{(2)}\right)\\
&\vdots \qquad\qquad \vdots\\
\boldsymbol{Y}^{(100)} &\equiv \left(Y_1^{(100)}, Y_2^{(100)}, \ldots, Y_{18}^{(100)}\right)
\end{aligned}$$

Calculate $\qquad \bar{Y}_i = \dfrac{\sum_{k=1}^{100} Y_i^{(k)}}{100}$

Step 6 Go to step 3 to estimate m 50 times so we get $\hat{m}^{(1)}, \hat{m}^{(2)}, \ldots, \hat{m}^{(50)}$

Step 7 Calculate mean vector ; $\bar{\hat{m}} = \frac{1}{50}\sum_{k=1}^{50} \hat{m}^{(k)}$ based on 50 estimations then we get $\bar{\hat{m}}$ as the unbiased estimator of m , route count.

3.2.2.3 Estimation of route count base on Bayesian inference

This section presents the statistical method to estimate route count based on Bayesian inference. The Bayesian inference use to infer route count X_j when we know l_j (from EM algorithm) and Y_i (from observation), for i = 1,2, …, 18. j = 1,2,…,72.

From the posterior distribution of $\boldsymbol{X}$ by given l and $\boldsymbol{Y}$;

$$p\left(\boldsymbol{x}|l,\boldsymbol{y}\right) \propto f_2\left(\boldsymbol{y}|\boldsymbol{x}\right) p\left(\boldsymbol{x}|l\right)$$

where $f_2\left(\boldsymbol{y}|\boldsymbol{x}\right)$ is degenerate at $\boldsymbol{Y} = \boldsymbol{AX}$ and

$$p\left(\boldsymbol{x}|l\right) \ \alpha \ \prod_{i=1}^{c} \frac{l_i^{x_i}}{x_i!}$$

So we can infer the route count, X_i by using the Gibb sampler to draw $\boldsymbol{X}$ from $p\left(\boldsymbol{x}|l\right)$, and then evaluate $\boldsymbol{Y}$ by the equation $\boldsymbol{Y} = \boldsymbol{AX}$

The detailed procedure to estimate X_j , $j = 1,2,\mathbf{K},72$ by given l_j and Y_i , $i = 1,2,\mathbf{K},18$ by using the mixure of maximum likelihood and Bayesian inference is based on seven steps as follows:

Step 1 Generate 10 vectors $\boldsymbol{X}$ from 72 independent Poisson distributions with parameter vector m that has already been estimated based on EM in section 3.2.2.2

Step 2 Draw sample value of 10 parameter vectors l from 72 conditionally independent posterior distributions; $p(l_j|X_j)$ is a Gamma distribution with shape parameter $X_j + 1$ and scale parameter 1; $j = 1,2,\mathbf{K},72$.

Step 3 For each parameter vector l , in iteration t sample a candidate X_j^* of the element of $\boldsymbol{X}$ with priority from conditionally Poisson distribution produces all the other elements :

$$X_j^* \sim \text{Poisson}(\ X_j^* \,|\boldsymbol{X}_{-j}^{t-1}) \quad ;$$

Where $\boldsymbol{X}_{-j}^{t-1}$ represents all the element of $\boldsymbol{X}$ except for X_j at their current values:

$$\boldsymbol{X}_{-j}^{t-1} = (X_1^t, \mathbf{K}, X_{j-1}^t, X_{j+1}^{t-1}, \mathbf{K}, X_{72}^{t-1})$$

$$\text{set} \quad X_j^t = \begin{cases} X_j^* & \text{with probability} \quad \min(r,1) \\ X_j^{t-1} & \text{otherwise} \end{cases}$$

$$r = \frac{P(X_j^*)U(X_j^{t-1})}{P(X_j^{t-1})U(X_j^*)}$$

$$\text{where} \qquad p(X_j) = \frac{e^{-l_j} l_j^{x_j}}{x_j!} \quad , \quad U(X_j) = \frac{e^{-m_j} m_j^{x_j}}{x_j!}$$

Step 4 Directly compute the element of $\boldsymbol{Y}$ by $\boldsymbol{Y} = \boldsymbol{AX}$

Step 5 Let X_{tj}^k be the draw from 10 parallel sequences of iteration t of the k^th element of $\boldsymbol{X}$ (t =1, 2, …, n ; j = 1, 2, …, 10), compute B and W, the between and within-sequence variances for each k^th:

$$B = \frac{n}{9}\sum_{j=1}^{10}(\bar{X}_{.j} - \bar{X}_{..})^2 \ , \ \text{where} \ \bar{X}_{.j} = \frac{1}{n}\sum_{i=1}^{n} X_{ij}^k \quad , \quad \bar{X}_{..} = \frac{1}{10}\sum_{i=1}^{10}\bar{X}_{.j}$$

$$W = \frac{1}{10}\sum_{j=1}^{10} S_j^2 \quad , \ \text{where} \quad S_j^2 = \frac{1}{n-1}\sum_{i=1}^{n}(X_{ij}^k - \bar{X}_{.j})^2 \ \text{and} \quad \hat{R} = \frac{1}{n}(\frac{B}{W} + n - 1)$$

Step 6 Return to step 2 and iterate until $\sqrt{\hat{R}} \to 1$ for all k^th element.

Step7 Estimate route count for each direct route by

$$\hat{X}_k = \frac{1}{10}\sum_{j=1}^{10} X_{nj}^k \quad , \ k = 1,2, \ldots, 72$$

where $\hat{X}_k$ is the estimator of route count for direct route k^th

X_{nj}^k is the latest draw for parallel j

3.2.3 Calculation of the length of the current cycle time

This section presents the statistical formula to calculate the length of the current cycle time on each phase of actual intersections studied. Treat each intersection as a service system and cars as customers with each phase of the intersection as a server.
As discussion in the previous chapter, interarrival time follows the exponential distribution. Therefore, we can generate interarrival time from an exponential distribution. The parameter of the distribution is defined by traffic intensity estimated from an the mixed model in section 4.2.2. Finally the length of the current cycle time since the last traffic light change to the moment that any car arrives at the intersection is

the total of all interarrival times of the cars that arrive at the intersection in the current period.

Let C_i be the i[th] car that arrive at the intersection,

A_i be interarrival time between C_i and C_{i+1},

C_j be the first car after the last traffic light change,

C_n be the car at the moment,

and L be the length of the current cycle time

So $$L = \sum_{i=j}^{n-1} A_i$$

3.3 The fuzzy system process methodology

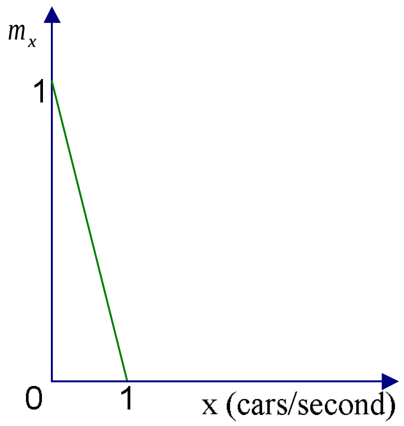
This section presents the method to combine the linguistic and numerical information from the previous input process methodology to derive output, the degree of traffic light change of each phase. Fuzzy logic systems with fuzzifier and defuzzifier will be used.

3.3.1 Fuzzifier

The fuzzifier performs a mapping from numerical information input such as number of cars behind the green light, number of cars behind the red light and the length of the current cycle time in to a fuzzy set. The number of cars behind the red or the green lights are assigned to fuzzy set as “zero”, “low”, “medium” and “high”. And the length of the current cycle time is assigned to fuzzy set as “short”, “medium” and “long”. Numerical information output , degree of traffic light change are assigned to fuzzy set as “no”, “probably no” “maybe”, “probably yes” and “yes”. The membership function of these fuzzy sets are defined below.

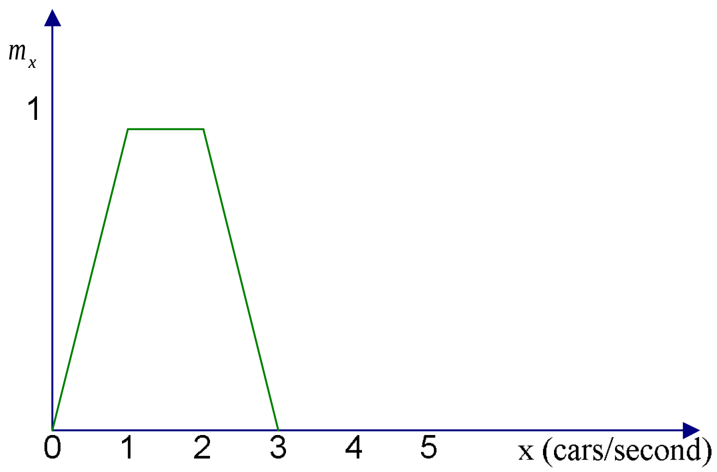
3.3.1.1 The membership function of the fuzzy set defined by the number of cars behind the green light

Fuzzy sets of the number of cars behind green light are assigned as “zero”, “low”, “medium”, and “high. The membership function of the fuzzy sets are triangular or trapezoidal according to the Figure 3.7- 3.10 as follows:



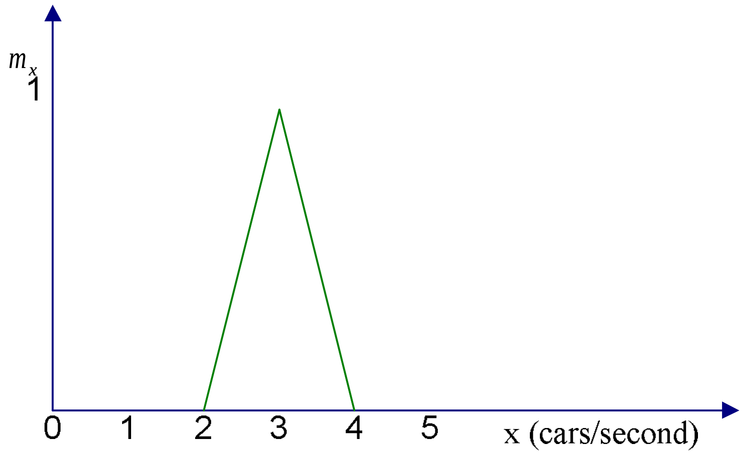
$$m_x(x_0) = \begin{cases} 1 - x_0 & , \quad 0 \le x_0 \le 1 \\ 0 & , \quad x_0 > 1 \end{cases}$$

Figure 3.7 The membership function form for fuzzy set "zero"
(Adapted from Kelsey & Bisset, 1993, P. 266 and Teodorovic & Vukadinovc, 1998, p. 51)



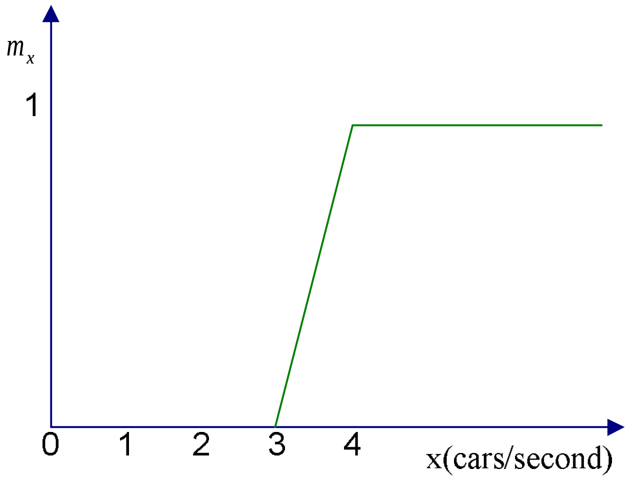
$$m_x(x_0) = \begin{cases} x_0 & , \quad 0 \le x_0 \le 1 \\ 1 & , \quad 1 < x_0 \le 2 \\ 3 - x_0 & , \quad 2 < x_0 \le 3 \\ 0 & , \quad x_0 > 3 \end{cases}$$

Figure 3.8 The membership function form for fuzzy set "low"
(Adapted from Kelsey & Bisset, 1993, P.266 and Teodorovic& Vukadinovc, 1998, p. 51)



$$m_x(x_0) = \begin{cases} 0 & , \quad x_0 < 2 \\ x_0 - 2 & , \quad 2 \le x_0 < 3 \\ 4 - x_0 & , \quad 3 \le x_0 < 4 \\ 0 & , \quad x_0 \ge 4 \end{cases}$$

Figure 3.9 The membership function form for fuzzy set "medium" (Adapted from Kelsey & Bisset, 1993, P.266 and Teodorovic& Vukadinovc, 1998, p. 51)

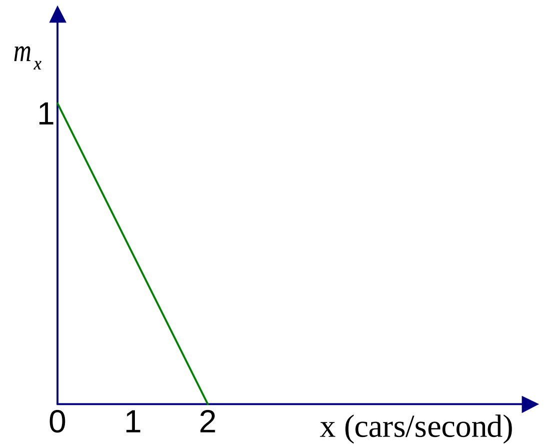


$$m_x(x_0) = \begin{cases} 0 & , \quad x_0 < 3 \\ x_0 - 3 & , \quad 3 \le x_0 < 4 \\ 1 & , \quad x_0 \ge 4 \end{cases}$$

Figure 3.10 The membership function form for fuzzy set "high" (Adapted from Kelsey&Bisset, 1993, p.266 and Teodorovic& Vukadinovc, 1998, p. 51)

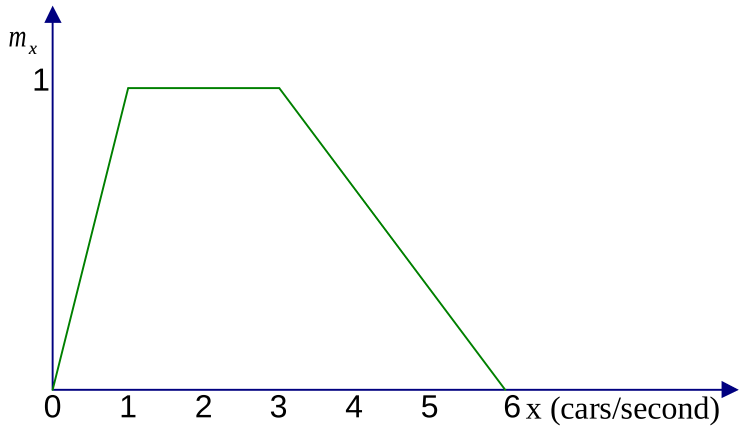
3.3.1.2 The membership function of the fuzzy set defined by the number of cars behind the red light.

Fuzzy sets of the number of cars behind the red light are assigned as “zero”, “low”, “medium”, and “high”. The membership function of the fuzzy sets are triangular or trapezoid according to the Figure 3.11-3.14 as follows:



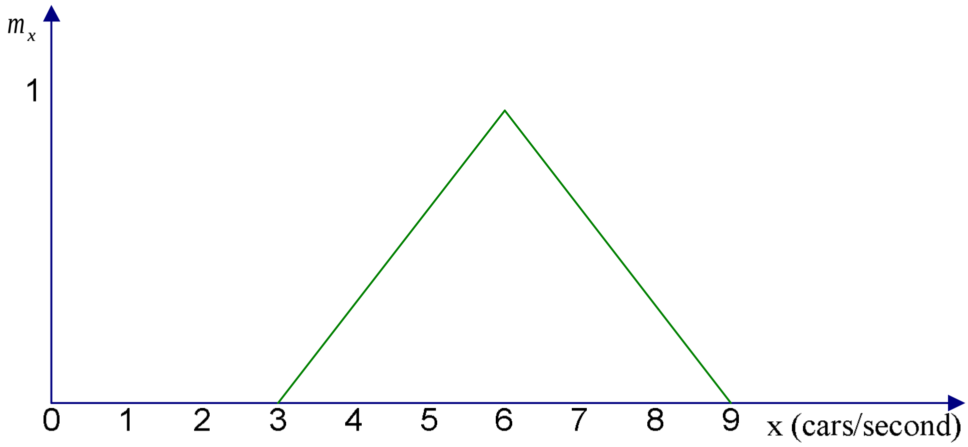
$$m_x(x_0) = \begin{cases} 1 - x_0 & , \quad 0 \le x_0 < 1 \\ 0 & , \quad x_0 \ge 1 \end{cases}$$

Figure 3.11 The membership function form for fuzzy set “zero” (Adapted from Kelsey & Bisset, 1993, p. 267 and Teodorovic& Vukadinovc, 1998, p. 51)



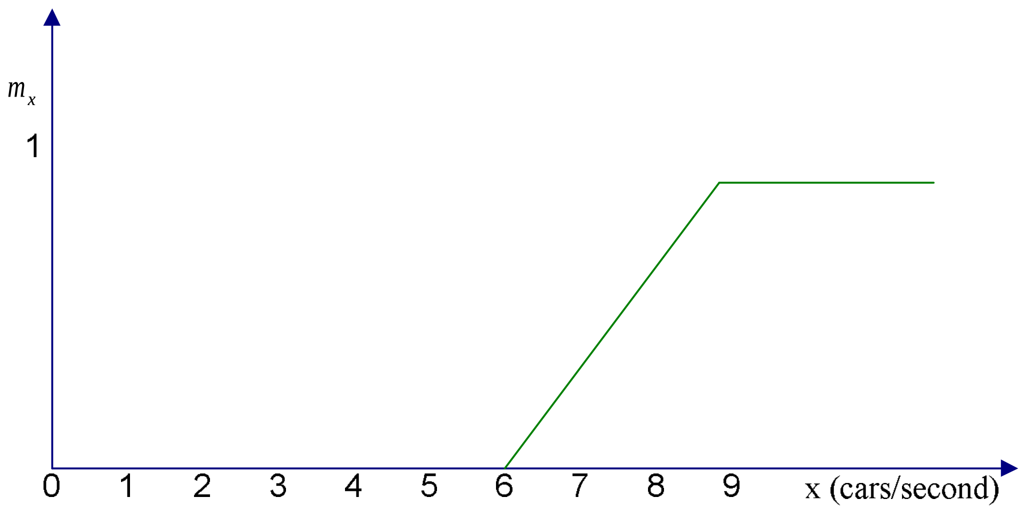
$$m_x(x_0) = \begin{cases} x_0 & , \quad 0 \le x_0 < 1 \\ 1 & , \quad 1 \le x_0 < 3 \\ 2 - \dfrac{x_0}{3} & , \quad 3 \le x_0 < 6 \\ 0 & , \quad x_0 \ge 6 \end{cases}$$

Figure 3.12 The membership function form for fuzzy set “low” (Adapted from Kelsey & Bisset, 1993, p. 267 and Teodorovic& Vukadinovc, 1998, p. 51)



$$m_x(x_0)=\begin{cases}0 & , \quad x_0<3\\ \dfrac{x_0}{3}-1 & , \quad 3\le x_0<6\\ 3-\dfrac{x_0}{3} & , \quad 6\le x_0<9\\ 0 & , \quad x_0\ge 9\end{cases}$$

Figure 3.13 The membership function form for fuzzy set "medium"(Adapted from Kelsey&Bisset, 1993, p.267 and Teodorovic& Vukadinovc, 1998, p. 51)

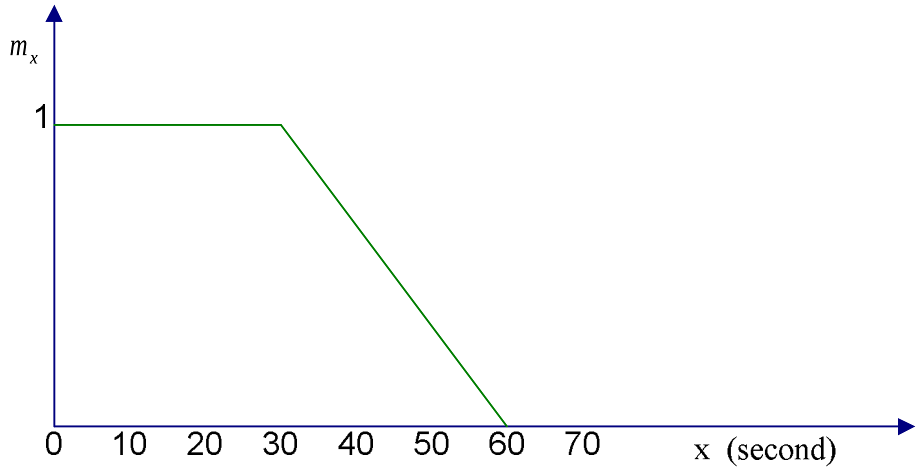


$$m_x(x_0)=\begin{cases}\dfrac{x_0}{3}-2 & , \quad 6\le x_0<9\\ 1 & , \quad x_0\ge 9\\ 0 & , \quad x_0<6\end{cases}$$

Figure 3.14 The membership function form for fuzzy set "high"(Adapted from Kelsey&Bisset, 1993, p.267 and Teodorovic& Vukadinovc, 1998, p. 51)

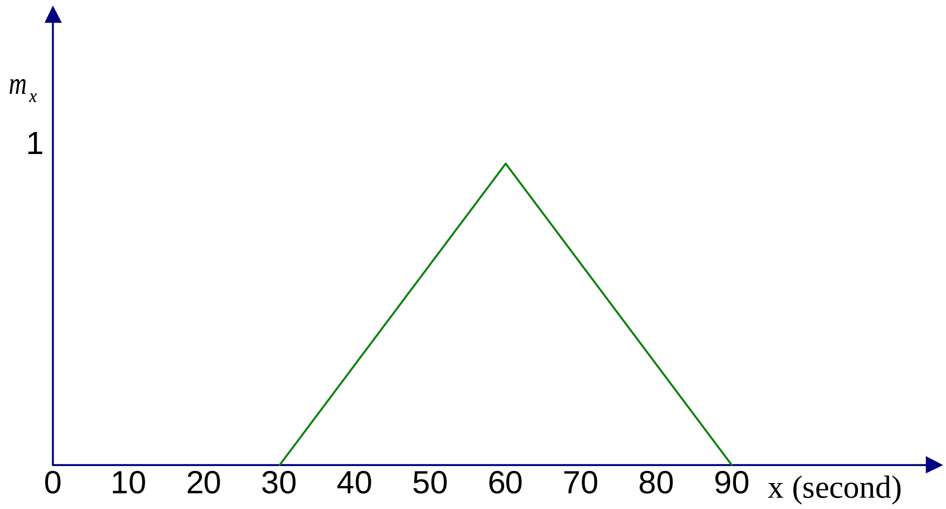
3.3.1.3 The membership function of the fuzzy set defined by the length of the current cycle time

Fuzzy sets of the length of current cycle time are assigned as “short”, “medium” and “long”. The membership function of the fuzzy sets are trapezoid according to the Figure 3.15-3.17 as follow:



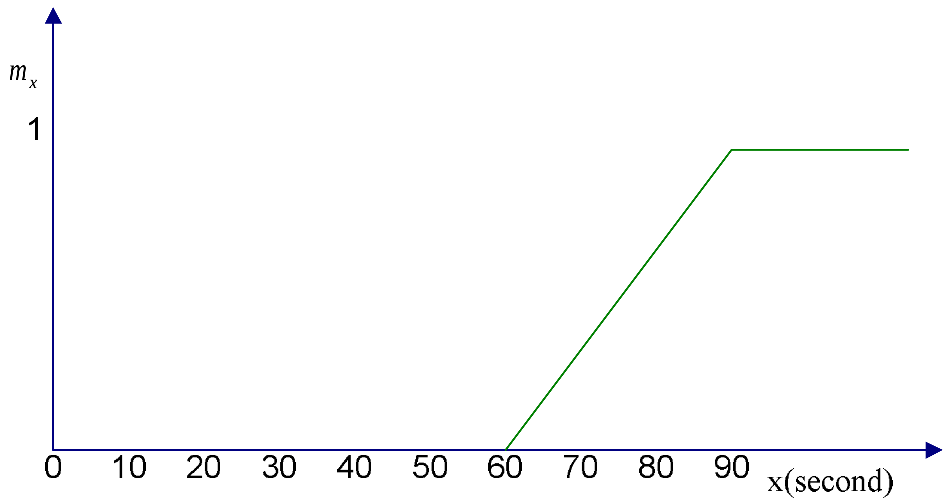
$$m_x(x_0) = \begin{cases} 1, & 0 \le x_0 < 30 \\ 2 - \dfrac{x_0}{30}, & 30 \le x_0 < 60 \\ 0, & x_0 \ge 60 \end{cases}$$

Figure 3.15 The membership function form for fuzzy set “short”(Adapted from Kelsey & Bisset, 1993, p.267 and Teodorovic & Vukadinovc, 1998, p. 51)



$$m_x(x_0)=\begin{cases} 0, & x_0<30 \\ \dfrac{x_0}{30}-1, & 30\le x_0<60 \\ 3-\dfrac{x_0}{30}, & 60\le x_0<90 \\ 0, & x_0\ge 90 \end{cases}$$

Figure 3.16 The membership function form for fuzzy set “medium”(Adapted from Kelsey&Bisset, 1993, p.267 and Teodorovic & Vukadinovc, 1998, p. 51)



$$m_x(x_0)=\begin{cases} 0, & x_0<60 \\ \dfrac{x_0}{20}-\dfrac{2}{3}, & 60\le x_0<90 \\ 1, & x_0\ge 90 \end{cases}$$

Figure 3.17 The membership function form for fuzzy set “long”(Adapted from Kelsey & Bisset, 1993, p. 267 and Teodorovic & Vukadinovc, 1998, p. 51)

3.3.2 Fuzzy rule base

This section provides a list of rules in notation form that govern traffic control at intersections. The fuzzy rule base consists of a collection of fuzzy IF-THEN rules according to Kelsey and Bisset's (1993) fuzzy rules base.

Let x_1 : number of cars that are behind the green light.

x_2 : number of cars that are behind the red light.

x_3 : current of cycle time.

y : degree of change.

F_1 : fuzzy set for the number of cars behind the green light is zero.

F_2 : fuzzy set for the number of cars behind the green light is low.

F_3 : fuzzy set for the number of cars behind the green light is medium.

F_4 : fuzzy set for the number of cars behind the green light is high.

F_5 : fuzzy set for the number of cars behind the red light is zero.

F_6 : fuzzy set for the number of cars behind the red light is low.

F_7 : fuzzy set for the number of cars behind the red light is medium.

F_8 : fuzzy set for the number of cars behind the red light is high.

F_9 : fuzzy set for length of the current cycle time is short.

F_{10} : fuzzy set for length of the current cycle time is medium.

F_{11} : fuzzy set for length of the current cycle time is long.

G_1 : fuzzy set for degree of change is no.

G_2 : fuzzy set for degree of change is probably no.

G_3 : fuzzy set for degree of change is maybe.

G_4 : fuzzy set for degree of change is probably yes.

G_5 : fuzzy set for degree of change is yes.

Using to the previous notations, the rule base in notation form are as follows:

Rule 1 IF x_1 is F_1 and x_2 is F_5 THEN y is G_1

Rule 2 IF x_1 is F_1 and x_2 is F_6 THEN y is G_5

Rule 3 IF x_1 is F_1 and x_2 is F_7 THEN y is G_5

Rule 4 IF x_1 is F_1 and x_2 is F_8 THEN y is G_5

Rule 5 IF x_1 is F_5 THEN y is G_1

Rule 6 IF x_1 is F_2 and x_2 is F_6 THEN y is G_1

Rule 7 IF x_1 is F_3 and x_2 is F_7 THEN y is G_1

Rule 8 IF x_1 is F_4 and x_2 is F_8 THEN y is G_1

Rule 9 IF x_1 is F_2 and x_2 is F_7 and x_3 is F_9 THEN y is G_3

Rule 10 IF x_1 is F_2 and x_2 is F_7 and x_3 is F_{10} THEN y is G_4

Rule 11 IF x_1 is F_2 and x_2 is F_7 and x_3 is F_{11} THEN y is G_5

Rule 12 IF x_1 is F_2 and x_2 is F_8 and x_3 is F_9 THEN y is G_2

Rule 13 IF x_1 is F_2 and x_2 is F_8 and x_3 is F_{10} THEN y is G_3

Rule 14 IF x_1 is F_2 and x_2 is F_8 and x_3 is F_{11} THEN y is G_4

Rule 15 IF x_1 is F_3 and x_2 is F_6 and x_3 is F_9 THEN y is G_2

Rule 16 IF x_1 is F_3 and x_2 is F_6 and x_3 is F_{10} THEN y is G_2

Rule 17 IF x_1 is F_3 and x_2 is F_6 and x_3 is F_{11} THEN y is G_3

Rule 18 IF x_1 is F_3 and x_2 is F_8 and x_3 is F_9 THEN y is G_3

Rule 19 IF x_1 is F_3 and x_2 is F_8 and x_3 is F_{11} THEN y is G_4

Rule 20 IF x_1 is F_3 and x_2 is F_8 and x_3 is F_{12} THEN y is G_5

Rule 21 IF x_1 is F_4 and x_2 is F_6 and x_3 is F_9 THEN y is G_3

Rule 22 IF x_1 is F_4 and x_2 is F_6 and x_3 is F_{10} THEN y is G_4

Rule 23 IF x_1 is F_4 and x_2 is F_6 and x_3 is F_{11} THEN y is G_5

Rule 24 IF x_1 is F_4 and x_2 is F_7 and x_3 is F_9 THEN y is G_2

Rule 25 IF x_1 is F_4 and x_2 is F_7 and x_3 is F_{10} THEN y is G_2

Rule 26 IF x_1 is F_4 and x_2 is F_7 and x_3 is F_{11} THEN y is G_3

3.3.3 Fuzzy inference engine

The fuzzy inference engine is used to infer a consequence fuzzy set from the rule base and facts received from the input process methodology. The product-sum-gravity method will be used to infer the consequence fuzzy set.

Let the facts of input be as follows:

$x_1^{/}$: fact of number of cars behind the green light

$x_2^{/}$: fact of number of car behind the red light

$x_3^{/}$: fact of the length of the current cycle time

Let $G_i^{/}$ denote the resulting fuzzy set from rule i. The membership function of $G_i^{/}$ is as follows:

$$m_{G_1'}(y) = m_{F_1}(x_1') \cdot m_{F_5}(x_2') \cdot m_{G_1}(y)$$

$$m_{G_2'}(y) = m_{F_1}(x_1') \cdot m_{F_6}(x_2') m_{G_5}(y)$$

$$m_{G_3'}(y) = m_{F_1}(x_1') \cdot m_{F_7}(x_2') \cdot m_{G_5}(y)$$

$$m_{G_4'}(y) = m_{F_1}(x_1') \cdot m_{F_8}(x_2') \cdot m_{G_5}(y)$$

$$m_{G_5'}(y) = m_{F_5}(x_1') \cdot m_{G_1}(y)$$

$$m_{G_6'}(y) = m_{F_2}(x_1') \cdot m_{F_6}(x_2') \cdot m_{G_1}(y)$$

$$m_{G_7'}(y) = m_{F_3}(x_1') \cdot m_{F_7}(x_2') \cdot m_{G_1}(y)$$

$$m_{G_8'}(y) = m_{F_4}(x_1') \cdot m_{F_8}(x_2') \cdot m_{G_1}(y)$$

$$m_{G_9'}(y) = m_{F_2}(x_1') \cdot m_{F_7}(x_2') \cdot m_{G_3}(y)$$

$$m_{G_{10}'}(y) = m_{F_2}(x_1') \cdot m_{F_7}(x_2') \cdot m_{F_{10}}(x_3') \cdot m_{G_4}(y)$$

$$m_{G_{11}'}(y) = m_{F_2}(x_1') \cdot m_{F_7}(x_2') \cdot m_{F_{11}}(x_3') \cdot m_{G_5}(y)$$

$$m_{G_{12}'}(y) = m_{F_2}(x_1') \cdot m_{F_8}(x_2') \cdot m_{F_9}(x_3') \cdot m_{G_2}(y)$$

$$m_{G_{13}'}(y) = m_{F_2}(x_1') \cdot m_{F_8}(x_2') \cdot m_{F_{10}}(x_3') \cdot m_{G_3}(y)$$

$$m_{G_{14}'}(y) = m_{F_2}(x_1') \cdot m_{F_8}(x_2') \cdot m_{F_{11}}(x_3') \cdot m_{G_4}(y)$$

$$m_{G_{15}'}(y) = m_{F_3}(x_1') \cdot m_{F_6}(x_2') \cdot m_{F_9}(x_3') \cdot m_{G_2}(y)$$

$$m_{G_{16}'}(y) = m_{F_3}(x_1') \cdot m_{F_6}(x_2') \cdot m_{F_{10}}(x_3') \cdot m_{G_2}(y)$$

$$m_{G_{17}'}(y) = m_{F_3}(x_1') \cdot m_{F_6}(x_2') \cdot m_{F_{11}}(x_3') \cdot m_{G_3}(y)$$

$$m_{G_{18}'}(y) = m_{F_3}(x_1') \cdot m_{F_8}(x_2') \cdot m_{F_9}(x_3') \cdot m_{G_3}(y)$$

$$m_{G_{19}'}(y) = m_{F_3}(x_1') \cdot m_{F_8}(x_2') \cdot m_{F_{11}}(x_3') \cdot m_{G_4}(y)$$

$$m_{G_{20}'}(y) = m_{F_2}(x_1') \cdot m_{F_8}(x_2') \cdot m_{F_{12}}(x_3') \cdot m_{G_5}(y)$$

$$m_{G_{21}'}(y) = m_{F_4}(x_1') \cdot m_{F_6}(x_2') \cdot m_{F_9}(x_3') \cdot m_{G_3}(y)$$

$$m_{G_{22}'}(y) = m_{F_4}(x_1') \cdot m_{F_6}(x_2') \cdot m_{F_{10}}(x_3') \cdot m_{G_4}(y)$$

$$m_{G_{23}'}(y) = m_{F_4}(x_1') \cdot m_{F_6}(x_2') \cdot m_{F_{11}}(x_3') \cdot m_{G_5}(y)$$

$$m_{G_{24}'}(y) = m_{F_4}(x_1') \cdot m_{F_7}(x_2') \cdot m_{F_9}(x_3') \cdot m_{G_2}(y)$$

$$m_{G'_{25}}(y) = m_{F_4}(x'_1) \cdot m_{F_7}(x'_2) \cdot m_{F_{10}}(x'_3) \cdot m_{G_2}(y)$$

$$m_{G'_{26}}(y) = m_{F_4}(x'_1) \cdot m_{F_7}(x'_2) \cdot m_{F_{11}}(x'_3) \cdot m_{G_3}(y)$$

Let G' be the consequence fuzzy set which is infered from the rule base and the facts. The membership function of G' is defined by

$$m_{G'}(y) = \sum_{i=1}^{26} m_{G'_i}(y)$$

3.3.4 Defuzzifier

The defuzzifier performs a mapping from fuzzy set G' to the crisp point, the center of gravity of G'.

Let y_i denote the center of gravity of the inference result G'_i and let S_i denote the area of G'_i as in Figure 3.3 Then y_i is defined as :

$$y_i = \frac{\int y \cdot m_{G'_i}(y)dy}{\int m_{G'_i}(z)dz} = \frac{\int y \cdot m_{G'_i}(y)dy}{S_i}$$

The leads to the center of gravity y' of the final consequence G' being given by

$$y' = \frac{\int y \cdot m_{G'}(y)dy}{\int m_{G'}(y)dy}$$

$$= \frac{\int y \cdot [m_{G'_1}(y) + \ldots + m_{G'_{26}}]dy}{\int [m_{G'_1}(y) + \ldots + m_{G'_{26}}(y)]dy} = \frac{\sum_{i=1}^{26} S_i \cdot y_i}{\sum_{i=1}^{26} S_i}$$

In practice, the identification of the center of gravity of G'_i is based on algebraic calculation . The center of gravity is the horizontal coordinate of the centroid of the area under the membership function. If the form of membership function is triangular, the centroid is the intersection of the straight line from each vertex to the middle points of the corresponding side. The centers of gravity of $G_i^{/}$ are computed in the following section

1) Identification of center of gravity of G_1

Consider the membership function form of G_1 in Figure 3.18

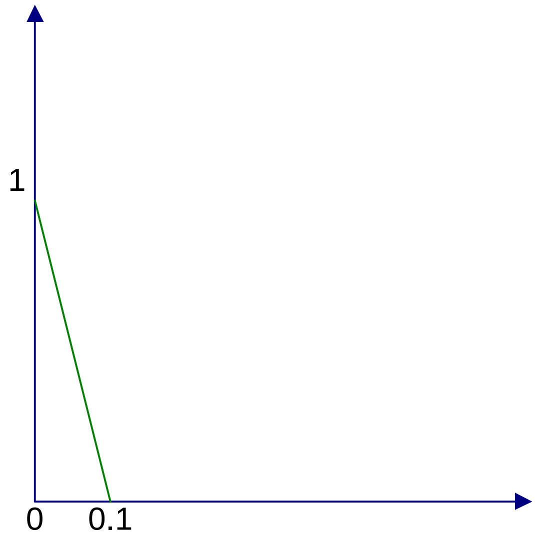


Figure 3.18 The membership function form of G_1

(Adapted from Kelsey & Bisset, 1993, p. 267)

From Figure 3.18 the center of gravity of G_1 is 0.033. And also the center of gravity of $G_i^{/}$ is 0.033

$$\therefore \quad y_i = 0.033 \quad \text{for } i = 1,5,6,7,9$$

2) Identification of center of gravity of G_2

Consider the membership function form of G_2 in Figure 3.19

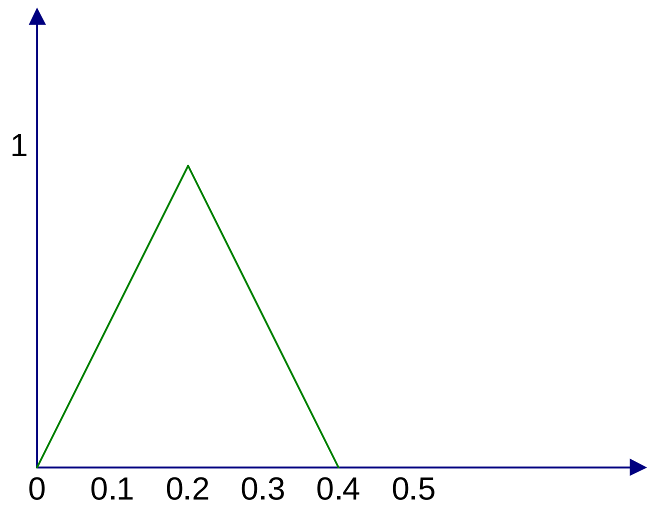


Figure 3.19 The membership function form of G_2

(Adapted from Kelsey & Bisset, 1993, p. 267)

From figure 3.19 center of gravity of $G_2^{/}$ is 0.2 . And also the center of gravity of

$G_i^{/} = 0.2$

$$\therefore \quad y_i = 0.2 \; ; \; i = 12,15,16,25$$

3) Identification of center of gravity of G_3

Consider the membership function form of G_3 in Figure 3.20

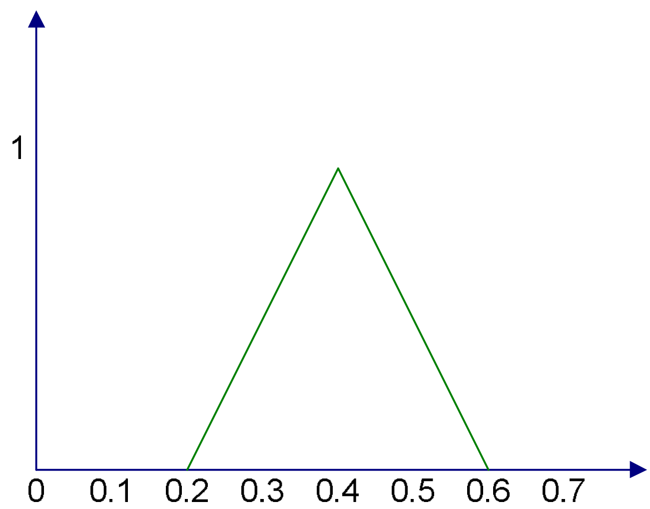


Figure 3.20 The membership function form of G_3

(Adapted from Kelsey & Bisset, 1993, p. 267)

From figure 3.20 center gravity of G_3 is 0.4. And also the center of gravity of $G_i^{/}$ is 0.4

$$\therefore \quad y_i = 0.4 \; ; \; i = 9,13,17,18,21,26$$

4) Identification of center of gravity of G_4

Consider the membership function form of G_4 in Figure 3.21

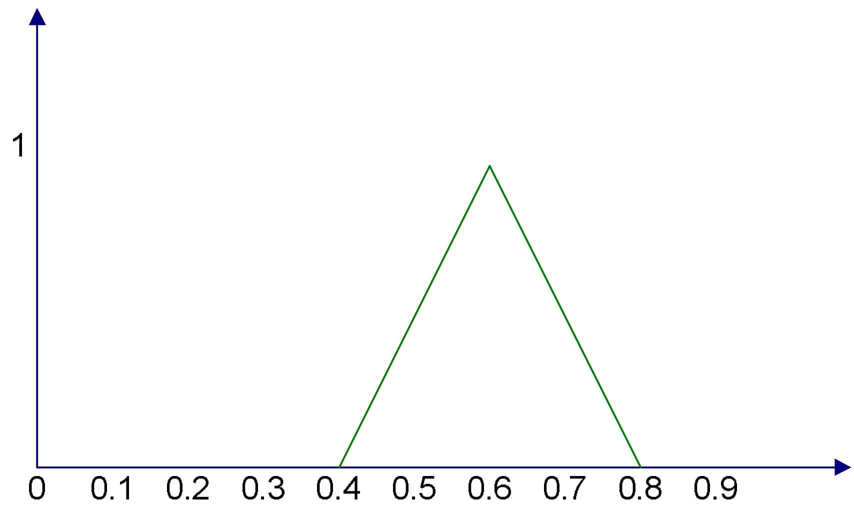


Figure 3.21 The membership function form of G_4

(Adapted from Kelsey & Bisset, 1993, p. 267)

From figure 3.21 center gravity of G_4 is 0.6. And also the center of gravity of $G_i^{/}$ is 0.6

$$\therefore \quad y_i = 0.6 \quad ; i = 10,14,19,22$$

5) Identification of center of gravity of G_5

Consider the membership function form of G_5 in Figure 3.22.

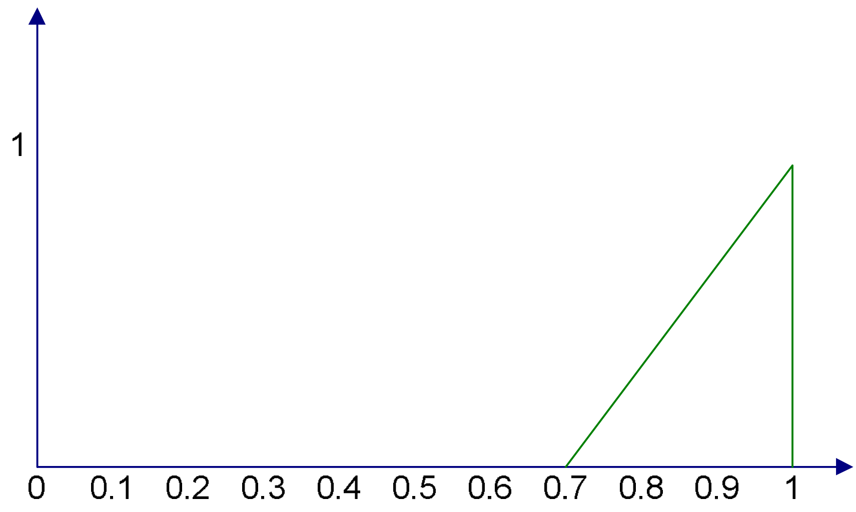


Figure 3.22 The membership function form of G_5

(Adapted from Kelsey & Bisset, 1993, p. 267)

From Figure 4.22 the center of gravity of G_5 is 0.85. And also the center of gravity of $G_i^/$ is 0.85.

$$\therefore \quad y_i = 0.85 \quad ; i = 2,3,4,11,20,23$$

3.4 The output process methodology

This section presents the method to simulate the current cycle time for each phase. Fuzzy logic control will be used to find the optimal moment that occurs when the optimal number of cars are behind the red light and the optimal number of cars that pass the green light. The optimal length on each phase of the cycle is the current cycle time at the optimal moment. The algorithm of simulation at each intersection is as follows:

Step 1. Let phase 1 of traffic signal cycle be the start phase.

Step 2. Iteratively generate cars and assign each car to each branch of the intersection based on proportion of cars from the branch that are computed in the input process.

Step 3. Generate interarrival time of each car in step 2 by exponential distribution with parameter beta.The value of beta is assigned by traffic intensity in the input process.

Step 4. Compute the important parameters of the simulation process, the input of fuzzy logic system such as:

$x_1^/$: number of cars that pass the green light.

$x_1^/$: is computed by counting the number of cars from the branch that are allowed to pass the intersection by the green light.

$x_2^/$: number of cars that stop behind the the red light.

$x_2^/$: is computed by counting the number of cars from the branch that are prohibited to pass the intersection by the red light.

$x_3^/$: the current cycle time.

$x_3^/$ is computed by the summation of interarrival time.

Step 5. Compute degree of change by using information from section 3.3 according to the following procedure:

Let S_i denote area of $G_i^/$; i = 1, 2,…,26

A_i denote area of G_i ; i = 1, 2, 3, 4, 5

D denote degree of change

y_i denote the center of gravity of $G_i^{/}$; i = 1, 2, …, 26

$$D = \frac{\sum_{i=1}^{26} y_i \cdot S_i}{\sum_{i=1}^{26} S_i}$$

From the figures 3.18-3.22, A_1 = 0.05, A_2 = 0.2, A_3 =0.2, A_4 = 0.2 and A_5 = 0.15 and

$$S_1 = m_{F_1}(x_1^{/}) \cdot m_{F_5}(x_2^{/}) \cdot A_1$$

$$S_2 = m_{F_1}(x_1^{/}) \cdot m_{F_6}(x_2^{/}) \cdot A_5$$

$$S_3 = m_{F_1}(x_1^{/}) \cdot m_{F_7}(x_2^{/}) \cdot A_5$$

$$S_4 = m_{F_1}(x_1^{/}) \cdot m_{F_8}(x_2^{/}) \cdot A_5$$

$$S_5 = m_{F_5}(x_1^{/}) \cdot A_1$$

$$S_6 = m_{F_2}(x_1^{/}) \cdot m_{F_6}(x_2^{/}) \cdot A_1$$

$$S_7 = m_{F_3}(x_1^{/}) \cdot m_{F_7}(x_2^{/}) \cdot A_1$$

$$S_8 = m_{F_4}(x_1^{/}) \cdot m_{F_8}(x_2^{/}) \cdot A_1$$

$$S_9 = m_{F_2}(x_1^{/}) \cdot m_{F_7}(x_2^{/}) \cdot A_3$$

$$S_{10} = m_{F_2}(x_1^{/}) \cdot m_{F_7}(x_2^{/}) \cdot m_{F_{10}}(x_3^{/}) \cdot A_4$$

$$S_{11} = m_{F_2}(x_1^{/}) \cdot m_{F_7}(x_2^{/}) \cdot m_{F_{11}}(x_3^{/}) \cdot A_5$$

$$S_{12} = m_{F_2}(x_1^{/}) \cdot m_{F_8}(x_2^{/}) \cdot m_{F_9}(x_3^{/}) \cdot A_1$$

$$S_{13} = m_{F_2}(x_1^{/}) \cdot m_{F_8}(x_2^{/}) \cdot m_{F_{10}}(x_3^{/}) \cdot A_3$$

$$S_{14} = m_{F_2}(x_1^{/}) \cdot m_{F_8}(x_2^{/}) \cdot m_{F_{11}}(x_3^{/}) \cdot A_4$$

$$S_{15} = m_{F_3}(x_1^{/}) \cdot m_{F_6}(x_2^{/}) \cdot m_{F_9}(x_3^{/}) \cdot A_2$$

$$S_{16} = m_{F_3}(x_1^{/}) \cdot m_{F_6}(x_2^{/}) \cdot m_{F_{10}}(x_3^{/}) \cdot A_2$$

$$S_{17} = m_{F_3}(x_1^{/}) \cdot m_{F_6}(x_2^{/}) \cdot m_{F_{11}}(x_3^{/}) \cdot A_3$$

$$S_{18} = m_{F_3}(x_1^{/}) \cdot m_{F_8}(x_2^{/}) \cdot m_{F_9}(x_3^{/}) \cdot A_3$$

$$S_{19} = m_{F_3}(x_1^{/}) \cdot m_{F_8}(x_2^{/}) \cdot m_{F_{11}}(x_3^{/}) \cdot A_4$$

$$S_{20} = m_{F_2}(x_1^{/}) \cdot m_{F_8}(x_2^{/}) \cdot m_{F_{12}}(x_3^{/}) \cdot A_5$$

$$S_{21} = m_{F_4}(x_1^{/}) \cdot m_{F_6}(x_2^{/}) \cdot m_{F_9}(x_3^{/}) \cdot A_3$$

$$S_{22} = m_{F_4}(x_1^{/}) \cdot m_{F_6}(x_2^{/}) \cdot m_{F_{10}}(x_3^{/}) \cdot A_4$$

$$S_{23} = \mu_{F_4}(x_1^{/}) \cdot \mu_{F_6}(x_2^{/}) \cdot \mu_{F_{11}}(x_3^{/}) \cdot A_5$$

$$S_{24} = m_{F_4}(x_1^{/}) \cdot m_{F_7}(x_2^{/}) \cdot m_{F_9}(x_3^{/}) \cdot A_2$$

$$S_{25} = m_{F_4}(x_1^{/}) \cdot m_{F_7}(x_2^{/}) \cdot m_{F_{10}}(x_3^{/}) \cdot A_2$$

$$S_{26} = m_{F_4}(x_1^{/}) \cdot m_{F_7}(x_2^{/}) \cdot m_{F_{11}}(x_3^{/}) \cdot A_3$$

Step 7 Generate Bernoulli random variable X, with parameter $P = D$, degree of change. If value of the random variable is equal to zero retain the phase then go to step 1.

Step 8 If value of the random variable is equal to 1 then change the previous phase to the next phase and go to step 2.

Step 9 Iterative until length of time equal 1800 second and covers all intersections.

Chapter 4
Input and Analysis

4.1 The data collection

This section presents the method used to collect data for direct links, namely number of cars that pass through a direct link in the traffic netwrok studied.

The method starts by assigning the collectors to 18 positions, 45 metres from the intersection as shown in Figure 4.1. Each collector must count the cars that pass them during 8.00-8.30 AM for 20 days.

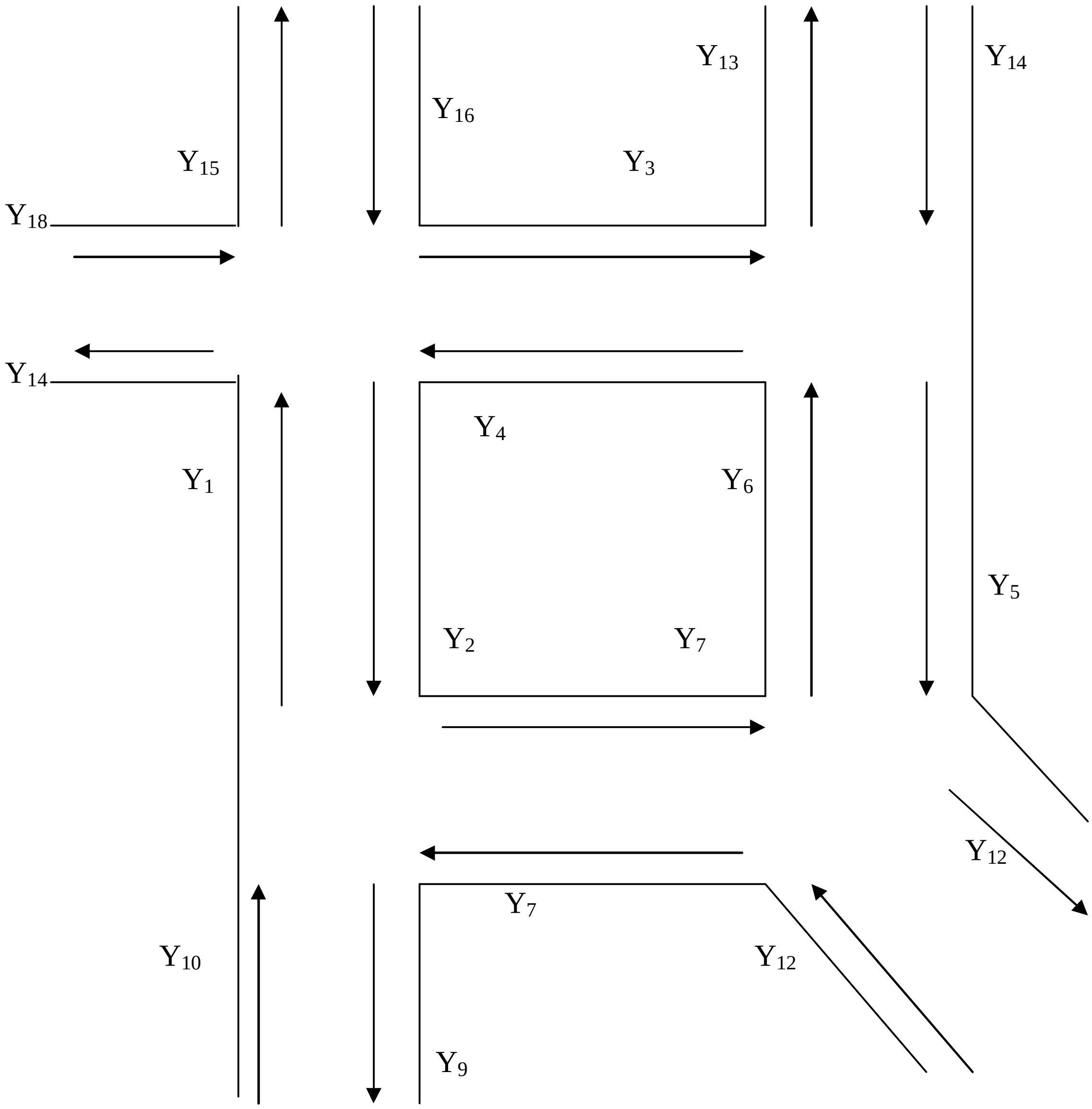


Figure 4.1 Diagram of 18 positions to count the cars that pass direct link

The number of cars that pass a direct link during the 20 days are given below.

Table 4.1 Table of the number of cars at 18 positions for 20 days

DAY	Y_1	Y_2	Y_3	Y_4	Y_5	Y_6	Y_7	Y_8	Y_9	Y_{10}	Y_{11}	Y_{12}	Y_{13}	Y_{14}	Y_{15}	Y_{16}	Y_{17}	Y_{18}
1	455	461	145	70	117	413	99	403	144	665	497	359	420	390	419	315	510	406
2	486	535	144	68	142	367	90	358	153	662	562	255	419	435	401	329	508	366
3	418	504	126	79	168	395	137	342	171	749	411	387	414	560	351	276	486	330
4	421	481	146	69	162	371	112	461	174	720	433	344	309	468	327	263	440	402
5	452	514	145	81	149	449	98	423	184	789	581	399	557	480	417	345	464	486
6	423	490	134	82	180	471	100	350	166	757	551	442	451	436	401	275	487	359
7	450	470	140	62	163	414	92	410	150	702	548	338	458	489	396	308	465	316
8	433	525	147	68	123	368	116	445	145	679	408	349	450	441	419	328	478	337
9	415	538	142	85	163	479	143	414	157	804	461	411	509	405	471	317	470	416
10	427	546	129	70	134	463	117	421	162	779	413	339	209	466	204	278	272	411
11	462	528	124	72	194	531	128	415	162	880	494	407	480	531	421	361	436	411
12	438	489	134	83	120	381	134	390	163	822	526	343	350	459	432	317	517	409
13	423	522	131	55	179	446	111	379	164	801	407	357	409	548	401	305	471	320
14	471	522	130	85	175	388	104	435	146	758	555	325	487	481	452	289	483	309
15	410	483	102	69	152	448	122	403	140	698	455	447	435	483	366	328	456	368
16	425	506	153	90	161	423	94	382	131	705	447	373	473	524	353	322	443	333
17	413	512	122	86	159	463	125	478	163	819	459	451	450	510	333	296	477	354
18	452	522	118	91	167	427	96	386	179	684	486	350	433	518	338	373	485	370
19	410	545	130	91	182	405	112	394	168	768	412	356	460	496	354	387	483	403
20	464	512	183	68	178	524	187	420	124	502	488	430	430	498	348	317	477	414

The number of cars from Table 4.1 will be used to estimate traffic counts for all direct routes.

4.2 The algorithm to simulate random variables

Rubinstein (1981) illustrates the algorithm to simulate random variable
based on its distribution. The important algorithms that are needed for the study are as follows.

4.2.1 The algorithm to generate random number

There are many methods to generate random numbers, such as the mid-square method, congruent metnods and so on, but the algorithm used to generate random numbers for this study is as follows.

1. Set arbitrary number, I
2. $K \leftarrow \dfrac{I}{127{,}773}$

3. $I \leftarrow 16{,}807 \cdot (I - 127{,}773K) - 2836 \cdot K$
4. If $I < 0$, deliver $I = I + 2{,}147{,}483{,}647$
5. $X = (4.656612875e - 10)$
6. $I \leftarrow I$

4.2.2 Gamma Distribution

A random variable X has a gamma distribution if its probability
density function (pdf) is defined as

$$f(x) = \begin{cases} \dfrac{x^{a-1} e^{\frac{-x}{b}}}{b^{a} G(a)} & , 0 \leq x \leq \infty, a > 0, b > 0 \\ 0 & , otherwise, \end{cases}$$

and denoted by $G(\alpha, \beta)$. One of the most important properties of the gamma distribution is the reproductive property, which can be successfully used for gamma generation. Let X_i , i = 1, 2, $\mathbf{K}$, n , be a sequence of independent random variables from $G(\alpha_i, \beta)$. Then $X = \sum_{i=1}^{n} X_i$ is from $G(\alpha, \beta)$ where $a = \sum_{i=1}^{n} a_i$. If α is and integer, say, $a = m$, a random variable from gamma distribution $G(m, b)$ can be obtained by summing m independent exponential random variables, that is,

$$X = b \sum_{i=1}^{m} (-\ln U_i) = -b \ln \prod_{i=1}^{m} U_i$$

which is called the Erlang distribution and denoted by $Er(m, \beta)$. The algorithm to generate a random variable from $Er(m, \beta)$ is as follows:

1. $X \leftarrow 0$.
2. Generate V from exponential distribution with $b = 1$, exp(1).
3. $X = X + V$
4. IF $a = 1$, $X \leftarrow bX$ and deliver X .
5. $a \leftarrow a - 1$.
6. Go to step 2.

4.2.3 Poisson Distribution.

An random variable has a Poisson distribution if its probability distribution function is equal to

$$f(x) = \frac{e^{-l}\, l^{x}}{x!} \quad , \ x = 0,\ 1,\ \mathbf{K},\ l > 0$$

and is denoted by P(λ). It is well known that, if the time intervals between events are from an exponential distribution with $b = \frac{1}{l}$, the number of events occurring in an unit interval of time is from P(l).

Mathematically, it can be written

$$\sum_{i=0}^{X} T_i \le 1 \le \sum_{i=0}^{X+1} T_i ,$$

where T_i , i = 0, 1, $\mathbf{K}, X+1$, are from exp($\frac{1}{l}$). Since T_i = -$\left(\frac{1}{l}\right)\ln U_i$, the last formula can be written as

$$-\sum_{i=0}^{X} \ln U_i \le l \le -\sum_{i=0}^{X+1} \ln U_i \qquad X = 0,\ 1,\ \ldots$$

or
$$\prod_{i=0}^{X} U_i \ge e^{-l} \ge \prod_{i=0}^{X+1} U_i \qquad X = 0,\ 1,\ \ldots$$

The following algorithm is written to generate a Poisson distribution:

1. $A \leftarrow 1$
2. $K \leftarrow 0$.
3. Generate random number, U_K from interval [0,1]
4. $A \leftarrow U_K A$
5. If $A < e^{-l}$, deliver $X = K$.
6. $K \leftarrow K + 1$.
7. Go to step 3.

4.2.4 Exponential distribution

The exponential distribution is the special case of the Gamma when $a = 1$, so a random variable X has an Exponential distribution if its p.d.f. is defined as

$$f(x)=\begin{cases}\frac{1}{b}\cdot e^{\frac{-x}{b}} & ,0\le x, \quad b>0\\ 0 & ,otherwise\end{cases}$$

The algorithm to generate an Exponential random variable with parameter b, is as follows:

1. Generate random number, U from interval [0, 1].
2. $X \leftarrow -b\ln(U)$

4.2.5 Bernoulli distribution

For a random experiment occurring only once and with output success or failure, let X be equal 1 for success with probability p, and X be 0 for failure with probability $1-p$, X is a Bernoulli random variable if its distribution function is defined as

$$f(x)=p^x(1-p)^{1-x}; \quad x=0,1$$

The algorithm to generate a Bernoulli random variable with parameter p is as follows:

1. Generate random number, U from interval [0, 1].
2. If $U \le 1-p$, deliver $X=0$.
3. $X=1$.

4.2.6 Uniform distribution

Let X be defined on the interval [a, b], and any value of X occur with equal probability, $\frac{1}{b-a}$, X is a uniform random variable and its distribution function is defined by

$$f(x)=\begin{cases}\frac{1}{b-a} & ,a\le x\le b\\ 0 & otherwise\end{cases}$$

The algorithm to generate a Uniform random variable with parameters a and b is as follows:

1. Generate random number, U
2. $X=a+(b-a)U$

4.3 Data algorithm analysis

To accomplish the research objective, the length of time appropriacy of the traffic lights, this section presents algorithm analysis steps. This can be done by developing a computer Fortran language program which is created on the important basis of three types of algorithms: EM algorithm, Metropolis-Hasting algorithm, in particular, the Gibbs sampler and Fuzzy logic algorithm. The process is comprised of 23 steps as follows:

Step 1 Let positive mean population of number of car that travel on direct route on traffic network

$$m = (m_1, \ldots, m_{72}) \text{ ; arbitrary.}$$

Step 2 Observe daily data on direct links for 20 days on 08:00 – 08:30 am

$$\boldsymbol{Y}^{(1)} \equiv \left(Y_1^{(1)}, Y_2^{(1)}, \ldots, Y_{18}^{1}\right)$$
$$\boldsymbol{Y}^{(2)} \equiv \left(Y_1^{(2)}, Y_2^{(2)}, \ldots, Y_{18}^{(2)}\right)$$
$$\vdots \qquad\qquad \vdots$$
$$\boldsymbol{Y}^{(20)} \equiv \left(Y_1^{(20)}, Y_2^{(20)}, \ldots, Y_{18}^{(20)}\right)$$

Calculate $$\overline{Y}_i = \frac{\sum_{k=1}^{20} Y_i^{(k)}}{20}$$

Step 3 Estimate m by $\hat{m} = (\hat{m}_1, \hat{m}_2, \ldots, \hat{m}_{72})'$ based on applied algorithm

$$m_{\mathrm{j}} \leftarrow \frac{m_j}{\sum_{i=1}^{18} a_{ij}} \sum_{i=1}^{18} \frac{a_{ij}\overline{Y}_i}{\sum_{k=1}^{72} a_{ik} m_k}$$

Step 4 Generate X_j from Poisson distribution with parameter m_j , $1,2,\ldots,72$

for 100 day

Step 5 Generate daily data on direct links for 100 days depend on X_j in step 4

$$\boldsymbol{Y}^{(1)} \equiv \left(Y_1^{(1)}, Y_2^{(1)}, \ldots, Y_{18}^{1}\right)$$
$$\boldsymbol{Y}^{(2)} \equiv \left(Y_1^{(2)}, Y_2^{(2)}, \ldots, Y_{18}^{(2)}\right)$$
$$\vdots \qquad\qquad \vdots$$
$$\boldsymbol{Y}^{(100)} \equiv \left(Y_1^{(100)}, Y_2^{(100)}, \ldots, Y_{18}^{(100)}\right)$$

Calculate $\bar{Y}_i = \dfrac{\sum_{k=1}^{20} Y_i^{(k)}}{100}$

Step 6 Go to step 3 to calculate $\hat{m}$ 50 times to get $\hat{m}^{(1)}, \hat{m}^{(2)}, \ldots, \hat{m}^{(50)}$

Step 7 Calculate mean vector ; $\bar{\hat{m}} = \dfrac{1}{50}\sum_{k=1}^{50} \hat{m}^{(k)}$ based on 50 estimations. Then $\bar{\hat{m}}$ is the unbiased estimator of m, route count.

Step 8 Generate 10 vectors $\boldsymbol{X}$ from 72 independent Poisson distributions with parameter vector m (already estimated from step 7)

Step 9 Draw sample value of 10 parameter vectors l from 72 conditionally independent posterior distributions, $p(l_j | X_j)$, that is Gamma distribution with shape parameter $X_j + 1$ and scale parameter 1; $j = 1, 2, \mathbf{K}, 72$.

Step 10 For each parameter vector l at iteration t draw a candidate X_j^* from Poisson distribution function as below.

$$X_j^* \sim \text{Poisson}(X_j^* | \boldsymbol{X}_{-j}^{t-1}) \quad ;$$

Where $\boldsymbol{X}_{-j}^{t-1}$ represents all the element of $\boldsymbol{X}$ except X_j, at their current values:

$$\boldsymbol{X}_{-j}^{t-1} = (X_1^t, \mathbf{K}, X_{j-1}^t, X_{j+1}^{t-1}, \mathbf{K}, X_{72}^{t-1})$$

$$\text{set} \quad X_j^t = \begin{cases} X_j^* & \textit{with probability} \quad \min(r,1) \\ X_j^{t-1} & \textit{otherwise} \end{cases}$$

$$r = \frac{P(X_j^*)U(X_j^{t-1})}{P(X_j^{t-1})U(X_j^*)}$$

$$\text{where} \quad P(X_j) = \frac{e^{-l_j} l_j^{x_j}}{x_j!} \quad , \quad U(X_j) = \frac{e^{-m_j} m_j^{x_j}}{x_j!}$$

Step 11 Directly compute the element of $\boldsymbol{Y}$ by $\boldsymbol{Y} = \boldsymbol{AX}$

Step 12 Let X_{tj}^k be the drawn from 10 parallel sequences of iteration t of the k^th^ element of $\boldsymbol{X}$ $(t = 1, 2, \mathbf{K}, n; \quad j = 1, 2, \mathbf{K}, 10)$, compute B and W, the between and within-sequence variances for each k^th^:

$$B = \frac{n}{9}\sum_{j=1}^{10}(\bar{X}_{.j} - \bar{X}_{..})^2 \ , \ \text{where} \ \bar{X}_{.j} = \frac{1}{n}\sum_{i=1}^{n} X_{ij}^k \quad , \quad \bar{X}_{..} = \frac{1}{10}\sum_{i=1}^{10}\bar{X}_{.j}$$

$$W = \frac{1}{10}\sum_{j=1}^{10} S_j^2 \quad , \ \text{where} \quad S_j^2 = \frac{1}{n-1}\sum_{i=1}^{n}(X_{ij}^k - \bar{X}_{.j})^2$$

and $\hat{R} = \frac{1}{n}(\frac{B}{W} + n - 1)$

Step 13 Return to step 8 and iterate until $\sqrt{\hat{R}} \to 1$ for all kth element.

Step 14 Estimate route count for each direct route by

$$\hat{X}_k = \frac{1}{10}\sum_{j=1}^{10} X_{nj}^{k} \quad , \; k = 1,2,\mathbf{K},72$$

where $\hat{X}_k$ is the estimator of route count for direct route k^{th}

X_{nj}^{k} is the latest draw for parallel j

Step 15. Set the start phase of traffic signal cycle.

Step 16. Create cars and find the probability, which is emerged from the calculation of route counts in Step 14, for each of the created car in order to randomise its moving from each branch of the intersection.

Step 17. Generate interarrival time of each car in step 16 by exponential distribution with parameter beta that is fixed by traffic intensity in the part of input process.

Step 18. Compute the important parameter of simulation process, input of fuzzy logic system such as:

$x_1^{/}$: number of cars that pass the green light.

$x_1^{/}$: number of cars from the branch that are allowed to pass the intersection by the green light.

$x_2^{/}$: number of car that stop behind the red light.

$x_2^{/}$: number of cars from the branch that are prohibited passing the intersection by the red light.

$x_3^{/}$: the current cycle time.

$x_3^{/}$: summation of interarrival time.

Step 19. Caculate the value of the cost function, by using information from section 3.4

Step 20 Generate Bernoulli random variable X, with parameter $P = D$, degree of change. If value of the random variable is equal zero then go to step 15.

Step 21 If value of the random variable is equal 1 then change the previous phase to the next phase and go to step 16.

Step 22 Caculate the value of the cost function.

Step 23 Iterate until length of time is complete and all intersections are covered.

4.4 The computer program in the Fortran language

The computer program is composed of a main program and 7 sub-programs.

4.4.1 Main program to estimate traffic intensity by the mixed model.

The main program is used to estimate traffic intensity using the mixed model. The optimal length of traffic signal lights is also calculated. The program consist of three parts.

4.4.1.1 Program to estimate traffic intensity by the EM algorithm.

This program takes the traffic intensity from the daily data observations to estimate the population mean of traffic counts on 72 direct routes. The program reads the input data that consists of traffic counts on the 18 direct links from daily data observation. Then it computes the sample mean of the traffic count for 20 days. The sample mean are used to estimate the population mean based on EM algorithm iteration. Finally the outputs of the program are populations mean of traffic counts on 72 direct routes.

4.4.1.2 Program to estimate traffic intensity by Gibbs sampler.

The population means estimated in 4.4.1.1 provides important information for this program. The function of this program is to estimate traffic intensity for 72 direct routes, given the population means and the data observations. The algorithm for the program is based on Gibb sampling. The outputs of this program are traffic intensities on each of the 72 direct routes.

4.4.1.3 Program to calculate optimal length of traffic signal light.

This program is used to calculate optimal length of signal light. The outputs from the program in 4.4.1.2 are traffic estimators for each of 72 direct routes. The estimators provide important information for this program that can be used to generate value of exponential variable. The value of exponential variable is the interarrival time. The interarrival time is used to define each car that arrives at the intersection. The current cycle time is also computed by summation of the interarrival times. The traffic intensity from the program in 4.4.1.2 and the current cycle time are the input data of the fuzzy logic system. The inputs are used to infer the degree of change for each phase based on the fuzzy logic system. Finally the degree of change is use to calculate the optimal length of the signal light.

4.4.2 Sub-Program

The sub-programs are designed to support the main program when the main program needs to compute the same object many times. There are 7 sub-program as follows:

4.4.2.1 Sub-Program to define any car belonging to each branch of road.

The function of this sub-program is to define any car belonging to each branch of the road at the intersection. The sub-program firstly generates random number. The random number is then separated to each branch based on the proportional traffic intensity in 4.4.1.2. Finally any car can be defined to belong to a particular branch by the random number. The technique of this program is branch index generation. The branch index is fixed by random number that are separated based on the proportional traffic intensity .

4.4.2.2 Sub-Program to generate an exponential random variable.

The function of this program is to generate an exponential random variable. The value of the variable is the interarrival time. This program supports the main program in 4.4.1.3 .

4.4.2.3 Sub-Program to generate a gamma random variable.

The function of this program is to generate a gamma random variable. The value of the variable is the population mean of traffic intensity. This program support the main program in 4.4.1.2 .

4.4.2.4 Sub-Program to generate a Poisson random variable.

The function of this program is to generate a Poisson random variable. The value of the variable is the number of cars. This program support the main program in 4.4.1.1 .

4.4.2.5 Sub-Program to generate a Bernoulie random variable.

The function of this program is to generate a Bernoulie random variable. The value of the variable is the decision index to decide whether to choose something or not based on its probability. So this program supports the main program in 4.4.1.2 and sub-program 4.4.2.1

4.4.2.6 Sub-Program to generate a random number.

The function of this program is to generate a random number. The value of random number is used to generate a random variable from any distribution. So this program supports the sub-program in 4.4.2.2-4.4.2.5

4.4.2.7 Sub-Program for fuzzy logic controller

The function of this program is to compute the degree of change in each phase based on the fuzzy logic system. The input of this program comes from the main program in 4.4.1.2 and 4.4.1.3

Chapter 5
Results of the Study

5.1 The number of cars on each direct route

There are 72 source-destination pairs (SD). The software estimated the number of cars on each SD by mixure of maximum likelihood and Baysian estimation. The output are shown in Table 5.1.

Table 5.1 Rate of cars on each SD (per second) from estimation.

SD.	NO.	SD.	NO.	SD.	NO.	SD.	NO.	SD.	NO.	SD.	NO.
X_1	2.183	X_{13}	0.032	X_{25}	2.476	X_{37}	0.02	X_{49}	2.208	X_{61}	0.038
X_2	0.062	X_{14}	0.005	X_{26}	2.551	X_{38}	0.017	X_{50}	2.187	X_{62}	0.055
X_3	2.228	X_{15}	0.088	X_{27}	0.163	X_{39}	0.06	X_{51}	1.917	X_{63}	0.022
X_4	0.018	X_{16}	0.043	X_{28}	2.168	X_{40}	2.267	X_{52}	0.02	X_{64}	2.047
X_5	0.035	X_{17}	0.015	X_{29}	0.040	X_{41}	2.18	X_{53}	2.415	X_{65}	0.02
X_6	2.1	X_{18}	0.023	X_{30}	0.023	X_{42}	0.017	X_{54}	0.023	X_{66}	2.248
X_7	2.668	X_{19}	0.015	X_{31}	0.278	X_{43}	0.012	X_{55}	0.025	X_{67}	2.072
X_8	1.873	X_{20}	0.052	X_{32}	0.067	X_{44}	0.015	X_{56}	0.075	X_{68}	0.328
X_9	0.016	X_{21}	0.01	X_{33}	0.032	X_{45}	0.33	X_{57}	0.038	X_{69}	0.02
X_{10}	0.023	X_{22}	0.055	X_{34}	0.08	X_{46}	0.052	X_{58}	0.052	X_{70}	0.045
X_{11}	0.35	X_{23}	0.035	X_{35}	0.133	X_{47}	0.032	X_{59}	0.113	X_{71}	0.052
X_{12}	0.0267	X_{24}	0.035	X_{36}	0.042	X_{48}	0.032	X_{60}	0.097	X_{72}	0.027

Note: SD. denote direct route.
No. denote rate of cars belong SD.

From Table 5.1 shows the rate of cars on direct links rather than the rate on direct routes.

5.2 The performance of traffic flow

The computer program generated the important parameters of traffic flow performance under the fuzzy logic controller and conventional controller.The parameters were the length of each phase, the number of cars behind the green light and the red light. The outputs of the parameters are shown in Table 5.2-5.9. To understand the numbers in each column, the No. green and No. red, are definded as follows:

1) No. Green denotes the number of cars behind the green light.
2) No. Red denotes the number of cars behind the red light.

3) The first number of No.Green is the number of cars stopping behind the red light at pre-phase includes the other cars moving past the green light in the first group at the current phase. .

4) The second number of No. Green is the order of the last car that moves to pass the green light or the number of all cars that pass green light at the current phase.

5) The first number of No. Red is the number of cars that stop behind the red light at pre-phase and still stop behind the red light including the other cars behind the red light in the first group at the current phase.

6) The second number of No. Red is the order of the last car behind the red light or the number of all cars that stop behind the red light at the current phase.

The criterion of length is defined as follows:

Less than 35 seconds indicates that the length is short
Between 35-70 seconds indicates that the length is moderate
Greater than 70 seconds indicates that the length is long

According to the criterion of length it is assumed that the average car uses 1 second to pass the intersection behind the green light. The criterion of No. Green and No. Red are defined in terms of length as follows:

Less than 35 cars show that No. Green or No. Red are few.
Between 35-70 cars show that No. Green or No. Red are moderate.
Greater than 70 cars show as No. Green or No. Red are many.

5.2.1 The performance of traffic flow based on fuzzy logic controller

The computer program generated the parameters of traffic flow performance for the fuzzy controller. The outputs of the parameters are shown as in Table 5.2-5.5

Table 5.2 Pattern of traffic flow during each phase at Ubonchareamsri intersection Based on fuzzy logic controller.

Cycle	Phase	No. Green	No.Red	Length(sec)
1	1	0	2-2	0.37
	2	4-5	1-1	1.67
	3	0	3-3	0.36
2	1	4-4	1-4	3.53
	2	5-7	1-1	1.33
	3	2-3	1-1	1.32
3	1	1-2	2-5	1.62
	2	7-8	0-1	2.14
	3	2-2	1-1	1.39
4	1	1-1	2-2	2.54
	2	4-4	1-1	1.52
	3	1-1	2-2	1.79
5	1	4-5	0-1	1.92
	2	2-3	1-1	2.04
	3	0	3-3	1.64
6	1	3-6	2-6	3.8
	2	7-11	1-3	4.38
	3	0	5-5	0.47
7	1	4-5	3-9	3.16
	2	10-11	1-1	1.70
	3	0	3-3	0.69
8	1	4-4	1-6	7.17
	2	6-90	2-63	87.54
	3	21-36	44-73	31.49
9	1	61-80	14-64	47.61
	2	54-130	12-75	73.90
	3	28-53	49-101	45.18
10	1	74-93	29-83	52.12
	2	69-139	16-67	69.262
	3	28-53	49-101	45.18

11	1	65-107	25-105	60.55
	2	94-185	13-81	76.65
	3	32-71	51-114	59.47
12	1	82-131	34-135	72.47
	2	121-168	16-57	62.32
	3	29-45	30-75	38.88
13	1	48-69	29-75	39.50
	2	64-122	13-66	68.77
	3	34-53	34-89	45.16
14	1	56-80	35-86	45.23
	2	79-129	9-46	52.88
	3	16-40	32-67	35.21
15	1	47-81	22-75	47.05
	2	62-122	15-64	67.15
	3	28-49	38-90	42.06
16	1	57-88	35-94	49.08
	2	91-160	5-56	61.53
	3	22-45	36-74	42.03
17	1	49-70	27-80	39.81
	2	72-104	10-23	28.82
	3	11-23	14-37	20.22
18	1	24-34	15-39	21.43
	2	36-58	5-17	21.66
	3	10-18	9-23	22.50
19	1	18-23	7-26	14.63
	2	23-98	5-63	66.82
	3	23-40	42-82	36.18
20	1	62-82	22-80	46.64
	2	72-114	10-40	46.78
Average Standardeviation		$\overline{X}$ = 53.6441 S = 51.5316	$\overline{X}$ = 44.8814 S = 38.7495	$\overline{X}$ = 36.022 S = 26.4409

Table 5.2 shows that the average of the number of cars behind the green light and the red light on each phase are respectively 54 and 49 cars. The average of the optimal

length on each phase is 36.022 seconds. The optimal length of each phase in early cycles (cycle 1-cycle 7) is very short. For late cycles (cycle 8 and later) the optimal length of each phase is moderate. The optimal length of phase 2 seems longer than the others. There are a few cars behind both the green and the red light in the early cycles. However, there are moderate numbers of the cars behind both the green and the red lights at the late cycles. In detail of cycle 1 (see Figure 4.3), each figure shows that there are no cars behind the green light and there are 2 cars behind the red light on phase 1 so it should be used only 0.37 seconds on this phase.

On phase 2 of cycle 1, 2 cars from phase 1 including the other 2 cars pass the green light and the last car that passes the green light on this phase is the 5th ; the number of all cars that pass the green light on this phase are 5 cars while 1 car stops behind the red light. This phase uses only 1.67 secconds. On phase 3 of cycle 1, 1 car from phase 2 still stops behind the red light and there are no other cars passing the green light while there are the other 2 cars behind the red light; the number of all cars behind the red light on this phase are 3 cars. The phase uses 0.35 seconds. The describtion of the other cycles are similar to the description of cycle 1 in which the number of cars on the current phase are impacted by the number of cars on the
pre- phase.

Table 5.3 Pattern of traffic flow during each phase at Clock Hall intersection based on fuzzy logic controller.

Cycle	Phase	No. Green	No. Red	Length(sec)
1	1	0	2-2	0.19
	2	3-5	1-9	6.53
	3	0	11-11	4.15
2	1	11-62	2-50	57.24
	2	44-121	8-109	91.63
	3	16-16	95-102	22.01
3	1	99-151	5-65	67.96
	2	61-85	6-57	62.34
	3	10-10	49-52	11.80
4	1	51-60	3-10	12.67
	2	10-21	2-15	22.29
	3	2-2	15-17	12.49
5	1	17-17	2-2	2.93
	2	1-1	3-3	1.63
	3	2-2	3-4	15.14
6	1	4-8	2-2	3.4
	2	3-3	1-2	2.734
	3	1-1	3-3	11.96

7	1	3-5	2-2	2.97
	2	2-5	2-8	9.29
	3	2-2	8-8	8.69
8	1	8-8	2-2	4.32
	2	1-8	3-16	19.95
	3	2-3	16-19	14.46
9	1	16-79	5-62	66.38
	2	56-128	8-101	85.91
	3	18-18	85-93	22.02
10	1	89-95	6-10	13.68
	2	10-43	2-57	79.68
	3	8-10	51-67	42.51
11	1	62-91	7-39	47.97
	2	35-59	6-35	46.59
	3	10-11	27-32	17.03
12	1	31-39	3-15	24.41
	2	14-72	3-67	79.91
	3	8-8	61-61	9.65
13	1	60-78	3-17	22.37
	2	18-116	1-144	128.44
	3	14-14	132-140	18.05
14	1	134-226	8-88	80.05
	2	82-149	8-89	80.01
	3	13-13	78-93	44.34
15	1	85-165	10-92	81.43
	2	82-130	12-63	73.69
	3	19-19	46-51	22.64
16	1	49-128	4-89	81.10
	2	81-160	10-101	86.23
	3	24-25	79-97	31.11
17	1	88-170	11-70	17.15
	2	-	-	-
	3	-	-	-
Average Standardeviation		$\bar{X}$ = 53.9184 S = 59.9992	$\bar{X}$ = 47.8163 S = 40.7455	$\bar{X}$ = 36.1454 S = 32.6769

Table 5.3 shows that the average of the number of cars behind the green light and the red light on each phase are respectively 54 and 48 cars. The average of the optimal length on each phase is 36.1454 seconds. There is an instability in the performance of traffic flow at early cycle (cycle 1-cycle 8). At cycle 1, there are a few cars behind the green and the red light and very short optimal length. For cycle 2 and cycle 3, the most number of cars behind the green light are many but the most number of cars behind the red light and the optimal length are moderate. The performance of traffic flow at cycle 4-cycle 8 is the same as the performance at cycle 1. At late cycles

(cycle 9 and beyond), the most optimal lengths are long. The most number of cars behind the green and red light are many. In detail of cycle 1 (see Figure 4.4), each figure shows that there are no cars behind the green light and there are 2 cars behind the red light on phase 1 so it should be used only 0.19 seconds on this phase.

On phase 2 of cycle 1, 2 cars from phase 1 including another one pass the green light and the last car passing the green light on this phase is the 5th; the number of all cars that pass the green light on this phase are 5 cars while 9 cars stop behind the red light. The phase uses 6.53 secconds. On phase 3 of cycle 1, 9 cars from phase 2 still stop behind the red light and there are no other cars passing the green light while there are the other 2 cars are behind the red light; the number of all cars that behind the red light on this phase are 11 cars. The phase uses 4.15 seconds. The description of the other cycles are similar to the description of cycle 1 in which the numbers of cars on the current phase are impacted by the number of cars on the pre- phase.

Table 5.4 Pattern of traffic flow during each phase at Chonlaprathan intersection based on fuzzy logic controller.

Cycle	Phase	No. Green	No. Red	Length(sec)
1	1	0	2-2	1.48
	2	3-3	1-1	1.18
	3	1-1	2-2	0.75
2	1	0	4-4	2.21
	2	4-4	2-5	3.61
	3	2-3	5-8	2.41
3	1	5-26	5-35	15.78
	2	22-167	15-196	129.63
	3	107-195	91-273	109.33
4	1	178-288	97-357	161.24
	2	246-449	113-386	213.46
	3	254-409	134-481	229.39
5	1	280-456	203-628	255.56
	2	441-720	189-611	337.61
	3	399-685	214-826	383.23
Average Standardeviation		$\bar{X} = 120.33$ S = 200.7387	$\bar{X} = 131.2121$ S = 219.3564	$\bar{X} = 68.3625$ S = 104.167

Table 5.4 shows that the average of the number of cars behind the green light and the red light on each phase are respectively 120 and 131 cars. The average of the optimal length on each phase is 68.3625 seconds. There are only five cycles during a specified time. There are a few cars and very short optimal lengths on all phases at cycle 1 and

cycle 2. For cycle 3,4 and 5 there are many cars behind the green and the red lights, while the optimal length is very long on all phases. In detail of cycle 1,
(see Figure 4.5) each figure shows that there are no cars behind the green light and there are 2 cars behind the red light on phase 1 so it should be used only 1.48 seconds for this phase. On phase 2 of cycle 1, 2 cars from phase 1 include another one passing the green linght; the number of all cars that pass the green light on this phase are 3 cars while 1 car stops behind the red light. The phase use 1.18 secconds. On phase 3 of cycle 1, 1 car from phase 2 passes the green light while there are 2 cars behind the red light; the number of all cars behind the red light on this phase are 2 cars. The phase uses 0.75 seconds. The describtion of other cycles are similar to the description of cycle 1 in which the number of cars on the current phase are impacted by the number of cars on the pre phase.

Table 5.5 Pattern of traffic flow during each phase at Airport intersection from fuzzy logic controller.

Cycle	Phase	No. Green	No. Red	Length(sec)
1	1	0	2-2	0.64
	2	1-6	3-16	6.05
	3	5-10	13-20	7.52
	4	12-16	10-28	10.27
2	1	17-41	13-60	39.69
	2	26-78	36-166	91.71
	3	51-73	117-183	41.39
	4	94-146	91-215	82.64
3	1	119-167	98-259	93.96
	2	106-198	155-453	166.36
	3	193-274	262-504	153.67
	4	256-381	250-594	213.85
4	1	289-412	307-675	230.63
	2	347-586	330-997	413.64
	3	396-560	603-1164	313.46
	4	-	-	-
Average Standardeviation		$\overline{X} = 92.6579$ S = 153.865	$\overline{X} = 154.3421$ S = 281.2448	$\overline{X} = 60.1001$ S = 94.7208

Table 5.5 shows that the average of the number of cars behind the green light and the red light on each phase are respectively 93 and 154 cars. The average of the optimal length on each phase is 60.1001 seconds. there are only four cycles during the specified time. There are a few cars and very short optimal length on all phases at cycles 1. For cycles 2,3 and 4 there are many cars behind the green and the red lights, while the optimal length is very long on most phases. In detail of cycle 1

(see Figure 4.6), each figure shows that there are no car behind the green light and there are 2 cars behind the red light on phase 1 so it should be used only 0.64 seconds for this phase.

On phase 2 of cycle 1, there are 6 cars passing the green light , the number of all cars that pass the green light on this phase is 6 cars while 2 cars from phase 1 still stop behind the red light including another one; the number of all cars that stop behind the red light on this phase are 16 cars. The phase uses 6.05 secconds. On phase 3 of cycle 1, 5 cars from 16 cars on phase 2 pass the green light and the last car that passes the green light on this phase is the 10th; the number of all cars that pass the green light on this phase is 10 cars while the 11 cars from 16 cars on phase 2 still stop behind the red light, including the other 2 cars are also behind the red light; the number of all cars behind the red light on this phase are 20 cars. The phase uses 7.52 seconds. On phase 4 of cycle 1, 12 cars from the 20 cars on phase 3 pass the green light and the last cars that pass the green light on this phase is the 16th; the number of all cars behind the green light on this phase are 16 cars. There are 8 cars from 20 cars on phase 3 still stopping behind the red light include the other 2 cars; the number of the all cars behind the red light on this phase are 28 cars. The phase uses 10.27 seconds. The description of the other cycles are similar to the description of cycle 1 in which the number of cars on the current phase are impacted by the number of cars on the
pre- phase.

5.2.2 The performance of traffic flow based on conventional controller

The computer program generated the parameters of traffic flow performance for the conventional controller. The outputs of the parameters are shown as Table 5.6-5.9

Table 5.6 Pattern of traffic flow during each phase at Uboncharearnsri intersection based on conventional control.

Cycle	Phase	No. Green	No. Red	Length(sec)
1	1	0-11	2-30	20
	2	28-56	4-20	25
	3	4-20	18-36	25
2	1	26-35	12-29	20
	2	24-47	4-28	25
	3	12-30	18-55	25
3	1	31-41	26-37	20
	2	36-55	3-16	25
	3	7-19	11-39	25

4	1	21-31	20-40	20
	2	38-63	4-21	25
	3	7-27	16-52	25
5	1	37-51	17-44	20
	2	39-66	7-24	25
	3	14-27	12-28	25
6	1	20-31	10-34	20
	2	30-53	6-27	25
	3	11-25	18-42	25
7	1	28-42	16-36	20
	2	37-56	1-22	25
	3	9-18	15-51	25
8	1	30-40	23-55	20
	2	52-79	5-30	25
	3	9-22	23-47	25
9	1	32-43	17-48	20
	2	46-69	4-34	25
	3	16-34	20-60	25
10	1	36-52	26-42	20
	2	35-59	9-24	25
	3	13-24	13-43	25
11	1	25-40	20-49	20
	2	48-77	3-32	25
	3	12-25	22-47	25
12	1	36-46	13-35	20
	2	35-48	2-15	25
	3	8-20	9-41	25
13	1	18-32	25-49	20
	2	45-63	6-25	25
	3	13-28	14-42	25
14	1	22-31	22-40	20
	2	35-51	7-30	25
	3	15-27	17-58	25
15	1	32-41	28-46	20
	2	43-67	5-24	25
	3	10-23	16-43	25
16	1	29-39	16-36	20
	2	33-59	5-24	25
	3	11-29	15-41	25
17	1	27-42	16-43	20
	2	37-56	8-31	25
	3	13-25	20-48	25
18	1	37-42	13-30	20
	2	24-54	8-25	25
	3	11-22	16-46	25
19	1	25-35	23-50	20
	2	52-82	0-19	25
	3	4-47	17-58	25
20	1	37-45	23-46	20
	2	42-71	6-27	25

20	3	12-24	17-27	25
21	1	28-35	21-44	20
	2	40-70	6-23	25
	3	10-23	15-38	25
22	1	27-34	13-39	20
	2	40-72	1-16	25
	3	4-18	14-41	25
23	1	26-31	17-36	20
	2	32-49	6-24	25
	3	12-26	14-42	25
24	1	23-37	21-57	20
	2	52-75	7-30	25
	3	12-22	20-56	25
25	1	36-42	22-46	20
	2	42-68	6-22	25
	3	10-24	14-36	25
26	1	23-33	15-30	20
	2	-	-	25
	3	-	-	25
Average Standardeviation		$\overline{X}$ = 41.7895 S = 17.554	$\overline{X}$ = 36.8553 S = 11.4108	

Table 5.6 shows that the average of the number of cars behind the green light and the red light on each phase are respectively 42 and 37 cars. The number of cars behind the green and the red lights on most phases are moderate. In detail of cycle 1 (see Figure 4.3), each figure shows that there are 11 cars behind the green light and there are 30 cars behind the red light on phase 1, it uses 20 seconds for this phase. On phase 2 of cycle 1, 28 cars from 30 cars on phase 1 pass the green light and the last car that passes the green light on this phase is the 56th; the number of all cars that pass the green light on this phase is 56 cars while 2 cars from 30 cars on phase 1 still stop behind the red light including the other 2 cars; the last car behind the red light on this phase is the 20th so that the number of all cars stopping behind the red light on this phase is 20 cars.The phase use 25 secconds. On phase 3 of cycle 1, 4 cars from the 20 cars on phase 2 pass the green light and the last car that passes the green light is the 20th ; so the number of all cars passing the green light on this phase is 20. There are 16 cars from the 20 cars on phase 2 still stopping behind the red light including the other 2 cars are also behind the red linght on this phase; the number of all cars behind the red light on this phase are 36 cars. The phase uses 25 seconds. The description of other cycles are similar to the description of cycle 1 in which the number of car on the current phase are impacted by the number of car on the pre-phase.

Table 5.7 Pattern of traffic flow during each phase at Clock Hall intersection based on conventional controller.

Cycle	Phase	No. Green	No. Red	Length(sec)
1	1	0-23	2-18	20
	2	17-40	3-29	25
	3	4-4	27-33	25
2	1	32-44	3-17	20
	2	14-34	5-27	25
	3	8-8	21-34	25
3	1	31-49	5-21	20
	2	20-46	3-39	25
	3	4-4	37-46	25
4	1	40-59	8-32	20
	2	31-50	3-25	25
	3	3-4	24-36	25
5	1	29-52	9-26	20
	2	24-34	4-24	25
	3	5-5	21-28	25
6	1	27-46	3-16	20
	2	16-24	2-16	25
	3	3-3	15-22	25
7	1	17-36	7-16	20
	2	13-18	5-18	25
	3	7-7	13-23	25
8	1	17-38	8-23	20
	2	24-45	1-31	25
	3	2-4	31-44	25
9	1	38-52	8-25	20
	2	22-38	5-32	25
	3	6-6	28-40	25
10	1	36-51	6-25	20
	2	23-42	4-25	25
	3	8-8	19-30	25
11	1	24-35	8-17	20
	2	16-33	3-29	25
	3	4-4	27-32	25
12	1	30-35	4-4	20
	2	12-22	4-34	25
	3	7-8	29-39	25
13	1	33-48	8-27	20
	2	24-41	5-28	25
	3	7-7	23-28	25
14	1	26-33	4-14	20
	2	12-26	5-18	25
	3	4-5	16-33	25
15	1	25-45	10-33	20
	2	33-48	2-17	25
	3	3-5	16-35	25
16	1	28-29	9-15	20
	2	14-29	3-26	25

16	3	2-3	26-39	25
17	1	35-52	6-22	20
	2	22-42	2-38	25
	3	4-6	36-41	25
18	1	40-58	3-20	20
	2	16-42	6-37	25
	3	6-6	33-46	25
19	1	37-51	11-33	20
	2	31-47	4-33	25
	3	7-7	28-35	25
20	1	33-62	4-27	20
	2	28-53	1-29	25
	3	1-2	30-40	25
21	1	34-51	8-29	20
	2	28-46	3-28	25
	3	3-3	27-38	25
22	1	33-53	7-27	20
	2	26-53	3-32	25
	3	4-5	30-41	25
23	1	36-47	7-27	20
	2	26-43	3-29	25
	3	6-6	25-43	25
24	1	37-52	8-28	20
	2	25-39	5-18	25
	3	7-7	13-24	25
Average Standarderviation		$\bar{X}$ = 30.0417 S = 19.5358	$\bar{X}$ = 28.5278 S = 8.617	

From Table 5.7 shows that the average of the number of that behind the green light and the red light on each phase are respectively 30 and 29 cars. the number of cars behind the green and the red light on most phases are moderate. In detail of cycle 1 (see Figure 4.4), each figure shows that there are 23 cars passing the green light and there are 18 cars stopping behind the red light; it uses 20 seconds on this phase. On phase 2 of cycle 1, 17 cars from 18 cars on phase 1 pass the green linght and the last car that passes the green light on this phase is the 40th; the number of all cars that pass the green light on this phase is 40 cars while 1 car from 18 cars on phase 1 still stop behind the red light including the other 2 cars. The last car that stops behind the red light on this phase is the 29th; so that the number of all cars that stop behind the red light on this phase is 29 cars.The phase uses 25 secconds. On phase 3 of cycle 1, there are only 4 cars from 29 cars on phase 2 passing the green light while there are 25 cars from the 29 cars on phase 2 still stopping behind the red light including the other 2 cars; the number of all cars stopping behind the red light on this phase are 33 cars. The phase uses 25 seconds. The description of the other cycles are similar to the description of cycle 1 in which the number of cars on the current phase are impacted by the number of cars on the pre-phase.

Table 5.8 Pattern of traffic flow during each phase at Chonlaprathan intersection based on conventional controller.

Cycle	Phase	No. Green	No. Red	Length(sec)
1	1	0-16	2-32	20
	2	20-50	14-49	25
	3	29-44	22-58	25
2	1	33-55	27-70	20
	2	52-84	20-57	25
	3	41-57	18-61	25
3	1	32-48	31-66	20
	2	48-66	20-61	25
	3	40-61	23-63	25
4	1	42-58	23-52	20
	2	40-61	14-58	25
	3	34-47	26-75	25
5	1	41-53	36-69	20
	2	53-74	18-51	25
	3	34-52	19-55	25
6	1	34-49	23-55	20
	2	37-64	20-53	25
	3	37-51	18-55	25
7	1	35-47	22-47	20
	2	39-65	10-35	25
	3	18-30	19-63	25
8	1	35-53	30-64	20
	2	47-73	19-51	25
	3	37-56	16-53	25
9	1	29-37	26-56	20
	2	36-56	23-60	25
	3	38-55	24-64	25
10	1	33-54	33-70	20
	2	52-74	20-45	25
	3	34-49	13-43	25
11	1	24-35	21-45	20
	2	34-55	13-40	25
	3	29-42	13-47	25
12	1	30-46	19-51	20
	2	39-70	14-50	25
	3	32-50	20-50	25
13	1	33-47	19-58	20
	2	40-66	20-60	25
	3	45-62	17-50	25
14	1	29-40	23-47	20
	2	36-61	13-54	25
	3	37-54	19-58	25

15	1	37-48	23-53	20
	2	41-65	14-56	25
	3	36-57	22-64	25
16	1	42-51	24-59	20
	2	47-69	14-52	25
	3	24-45	30-68	25
17	1	45-58	25-57	20
	2	44-67	15-49	25
	3	33-45	18-48	25
18	1	27-42	23-53	20
	2	36-67	17-54	25
	3	34-54	22-53	25
19	1	35-49	20-37	20
	2	33-45	6-34	25
	3	23-41	13-42	25
20	1	33-39	21-47	20
	2	27-53	12-45	25
	3	26-41	21-50	25
21	1	33-46	19-55	20
	2	47-69	10-43	25
	3	24-35	21-58	25
22	1	44-79	12-56	20
	2	40-58	18-56	25
	3	40-58	18-56	25
23	1	35-46	23-67	20
	2	45-66	24-55	25
	3	45-62	22-60	25
24	1	43-56	19-58	20
	2	38-57	22-56	25
	3	43-60	15-49	25
25	1	30-40	21-52	20
	2	40-64	14-41	25
	3	28-43	15-55	25
26	1	30-43	27-69	20
	2	-	-	
	3	-	-	
Average Standarderviation		$\overline{\boldsymbol{X}} = 53.75$ S = 11.786	$\overline{\boldsymbol{X}} = 54.0526$ S = 8.7115	

Table 5.8 shows that the average of the number of cars behind the green light and the red light on each phase are 54 cars.The number of cars behind the green and the red light on all phases are moderate. In detail of cycle 1 (see Figure 4.5), each figure shows that there are 16 cars passing the green light and there are 32 cars stopping behind the red light, it uses 20 seconds for this phase. On phase 2 of cycle 1, 20 cars from 32 cars on phase 1 pass the green light , the number of all cars that pass the green light on this phase are 50 cars. There are 12 cars from the 32 cars on phase 1 still stopping behind the red light including the other 2 cars.The last cars that stops behind the red light on

this phase is 49th, so that the number of all cars that stop behind the red light on this phase are 49 cars. The phase uses 25 secconds. On phase 3 of cycle 1, 29 cars from 49 cars on phase 2 passing the green light, the last car that passes the green light on this pase is 44th , so that the number of all cars that pass the green light on this phase are 44 cars. There are 20 cars from 49 cars on phase 2 still stopping behind the red light including the other 2 cars, the number of all cars that stop behind the red light on this phase are 58 cars. The phase uses 25 seconds. The description of other cycles are similar to the description of cycle 1 in which the number of cars on the current phase are impacted by the number of cars on the
pre-phase.

Table 5.9 Pattern of traffic flow during each phase at Airport intersection based on conventional controller.

Cycle	Phase	No. Green	No. Red	Length(sec)
1	1	0-17	2-45	25
	2	13-18	34-61	20
	3	24-28	39-76	20
	4	35-54	43-81	25
2	1	38-52	45-83	25
	2	52-57	33-62	20
	3	23-33	41-60	20
	4	37-51	25-67	25
3	1	31-47	38-79	25
	2	39-47	42-71	20
	3	36-45	37-78	20
	4	39-55	41-69	25
4	1	39-53	32-75	25
	2	27-39	50-79	20
	3	40-54	41-73	20
	4	47-56	28-72	25
5	1	34-47	40-78	25
	2	37-49	43-87	20
	3	38-51	51-86	20
	4	45-64	43-77	25
6	1	44-59	35-84	25
	2	37-49	49-85	20
	3	41-52	46-74	20
	4	41-57	35-80	25
7	1	37-47	45-83	25
	2	38-48	47-73	20
	3	40-49	35-66	20
	4	42-57	26-69	25
8	1	28-39	43-79	25
	2	40-51	41-74	20
	3	44-48	32-68	20

8	4	42-50	38-76	25
9	1	38-54	40-92	25
	2	43-49	51-88	20
	3	37-42	53-91	20
	4	49-56	44-85	25
10	1	34-48	53-89	25
	2	50-55	41-80	20
	3	34-38	48-72	20
	4	44-59	30-82	25
11	1	39-52	45-87	25
	2	46-54	43-74	20
	3	39-43	37-75	20
	4	37-54	40-76	25
12	1	34-51	44-90	25
	2	43-52	49-74	20
	3	34-38	42-75	20
	4	44-60	33-72	25
13	1	30-44	44-72	25
	2	44-54	33-69	20
	3	28-37	43-72	20
	4	40-54	34-81	25
14	1	43-55	40-96	25
	2	56-71	42-81	25
	3	43-54	40-74	20
	4	43-62	33-64	20
15	1	33-46	33-69	25
	2	35-45	36-66	20
	3	29-36	39-66	20
	4	39-49	29-75	25
16	1	31-39	46-82	25
	2	37-49	47-73	20
	3	41-51	34-69	20
	4	35-46	36-72	25
17	1	35-41	39-75	25
	2	40-54	37-87	20
	3	34-49	55-88	20
	4	46-56	44-84	25
18	1	53-65	33-81	25
	2	39-50	44-79	20
	3	29-41	52-96	20
	4	60-84	38-74	25
19	1	34-51	42-86	25
	2	41-54	47-82	20
	3	44-51	40-78	20
	4	43-57	37-72	25
20	1	39-54	35-75	25
	2	36-45	41-71	20
	3	37-48	36-64	20
Average Standarderviation		$\overline{X} = 46.6203$ S = 9.7261	$\overline{X} = 76.519$ S = 8.7616	

From Table 5.9 shows that the average of the number of cars behind the green light and the red light on each phase are respectively 47 and 77 cars. The number of cars behind the green and the red light on all phases are moderate. In detail of cycle 1 (see Figure 4.6), each figure shows that there are 17 cars behind the green light and there are 45 cars behind the red light on phase 1; it uses 25 seconds for this phase. On phase 2 of cycle 1, there are 13 cars from 45 cars on phase 1 passing the green light , the number of all cars passing the green light on this phase is 18 cars while 32 cars from 45 cars on phase 1 still stop behind the red light including the other 2 cars , the number of all cars stopping behind the red light on this phase is 61 cars. The phase uses 20 secconds. On phase 3 of cycle 1, 24 cars from 61 cars on phase 2 passing the green light and the last car passing the green light on this phase is the 28th, the number of all cars passing the green light on this phase is 28 cars. There are 37 cars from 61 cars on phase 2 still stopping behind the red light including the other 2 cars, the number of all cars that behind the red light on this phase are 76 cars. The phase uses 20 seconds. On phase 4 of cycle 1, 35 cars from 76 cars on phase 3 passing the green light and the last cars passing the green light on this phase is the 54th, the number of all cars behind the green light on this phase are 54 cars. There are 41 cars from 76 cars on phase 3 still stopping behind the red light including the another 2 cars; the number all cars that behind the red light on this phase is 81 cars. The phase uses 25 seconds. The description of the other cycles are similar to the description of cycle 1 in which the number of cars on the current phase are impacted by the number of cars on the pre phase.

5.3 The controller performance comparison

The cost function provides a means of comparing the traffic flow performance of the fuzzy controller against the conventional controller. The lower the cost function is the better the perfomance. The controller performance comparisons are as illustrated in Figures 5.1-5.4 .

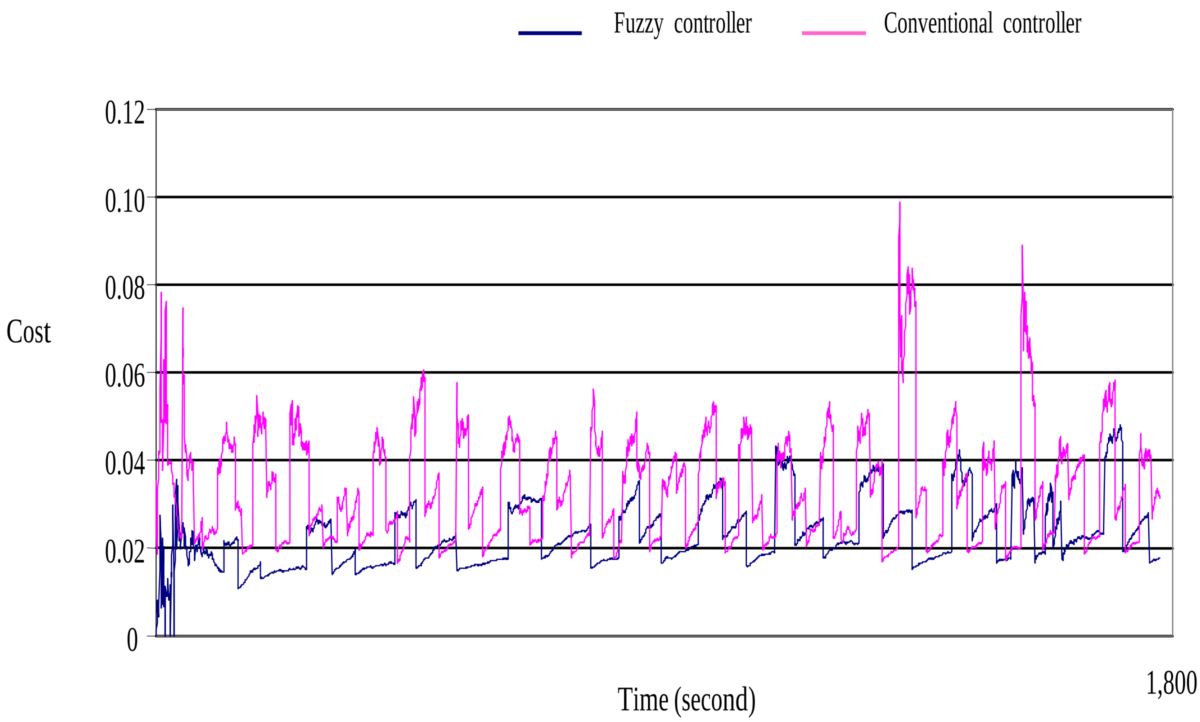


Figure 5.1 Controller performance comparison at Uboncharearnsri intersection.

Figure 5.1 shows that on average, the cost function based on the fuzzy controller is lower than the cost function based on the conventional controller.

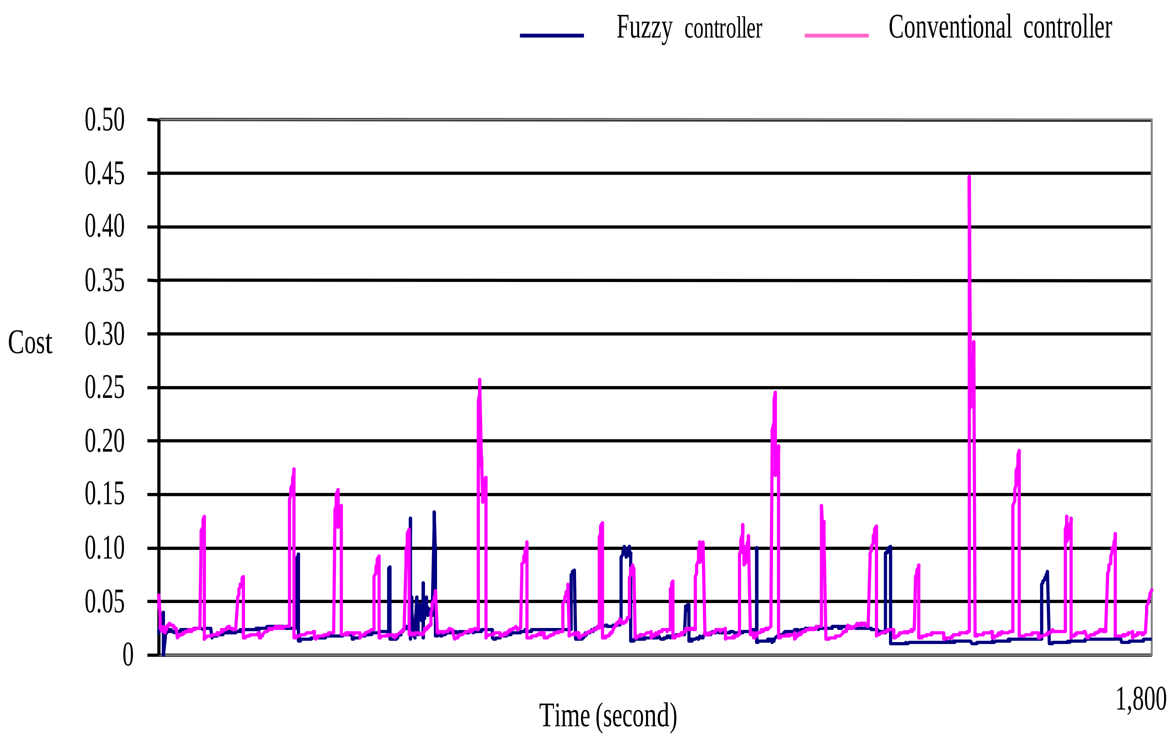


Figure 5.2 Controller performance comparison at Clock Hall intersection.

Figure 5.2 shows that on average, the cost function based on the fuzzy controller is lower than the cost function based on the conventional controller.

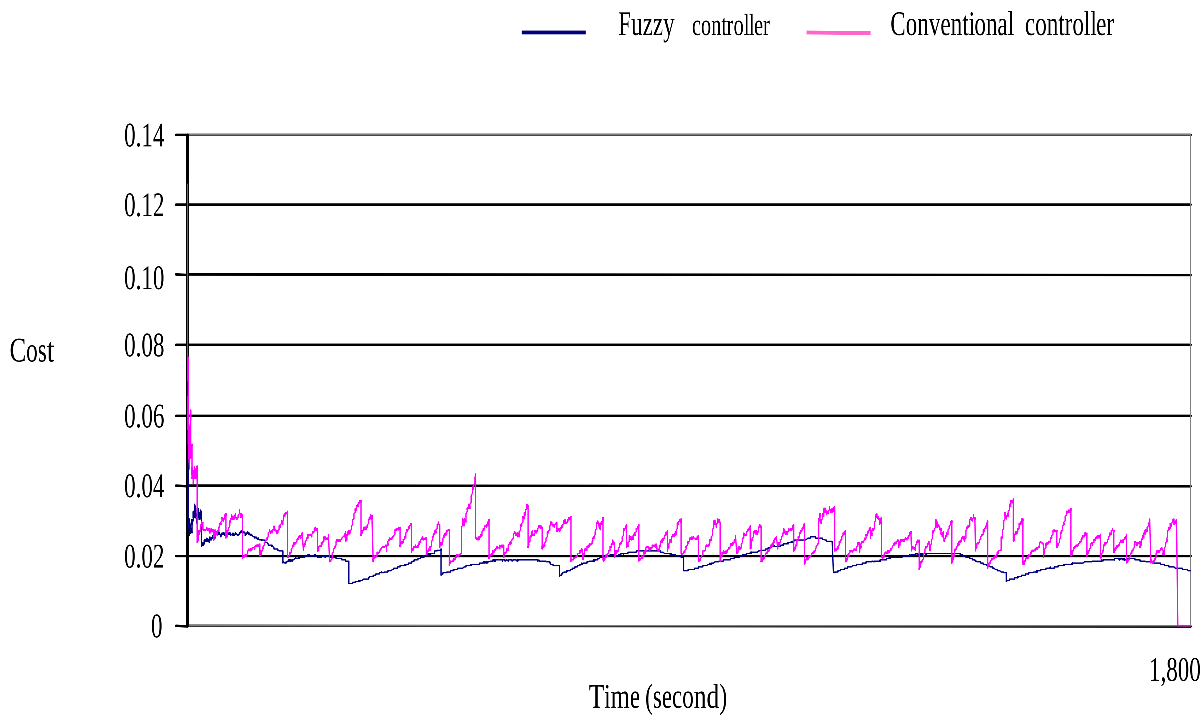


Figure 5.3 Controller performance comparison at Chonlaprathan intersection.

Figure 5.3 shows that on average, the cost function based on the fuzzy controller is lower than the cost function based on the conventional controller.

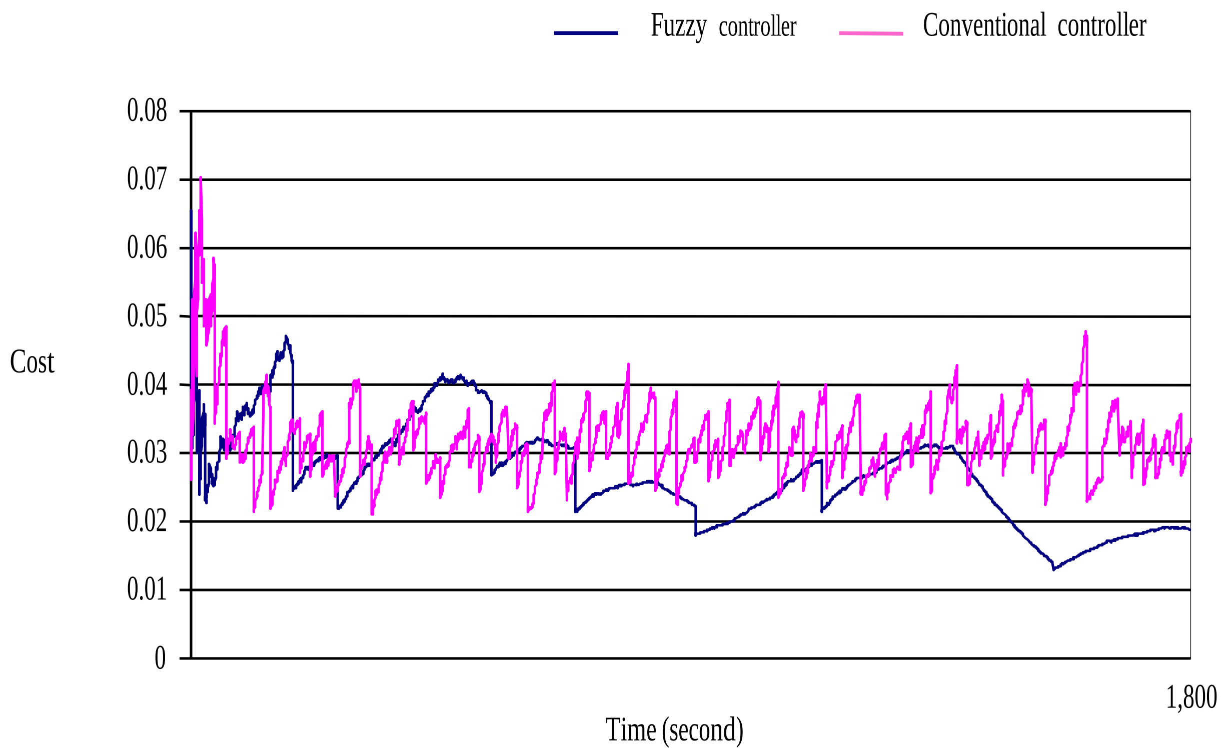


Figure 5.4 Controller performance comparison at Airport intersection.

Figure 5.4 shows that on average, the cost function based on the fuzzy controller is lower than the cost function based on the conventional controller.

Chapter 6
Conclusion and Discussion

6.1 Conclusion

This study aims at computing the optimal lengths of traffic signal lights on each phase of four intersections in the inner city of Ubon rachathani Province namely Ubonchareanrsri , Clock Hall , Chonlaprathan and Airport intersections. The expected outcomes consist of the method to calculate the traffic signal timing at the targeted intersections during rush hour and the traffic signal timing that is relevant to the number of vehicles at the intersections.

To estimate the number of cars that arrive at or depart from the intersections, the study uses a mixed model of maximum likelihood (Vardi, 1996) and Bayesian inference (Tebaldi & West, 1998). The process started with a survey at the intersections of the traffic system under study path. Let each intersection be a node and treat the traffic system as a network. The path that connects any two nodes was called a direct route and a direct link that refers to the path that have no nodes between the two ends. There are 72 direct routes and 18 direct links in the network. This enables the researcher to observe the number of cars passing on any direct link but not on the direct route.

A relation between the number of cars passing on a direct link and direct route are presented by an equation as follows:

$$\boldsymbol{Y} = \boldsymbol{AX}$$

$\boldsymbol{Y}$: direct link vector

$\boldsymbol{X}$: direct route vector

$\boldsymbol{A}$: routing matrix

In the process of data collection, the number of cars were observed on 20 days and the EM iteration was used to solve the equation to estimate the mean (l) of the number of cars on all links. Now knowing $\boldsymbol{Y}$ and l from observation and EM iteration, the next step was to estimate $\boldsymbol{X}$. Bayesian inference was used to achieve the goal; the illustrated distribution is as follows:

$$p\left(\boldsymbol{x} \mid \boldsymbol{y}, l\right)$$

The Gibbs sampler (Casella & George,1992) is used to establish the algorithm of the software to generate $\boldsymbol{X}$, and support starting point of the algorithm with the mean (l). As previously mentioned the study mixed the two methods of EM iteration and the Gibbs sampler to estimate the number of cars on all links.

The statistical inference shows the number of cars behind the green light and behind the red light. In addition, queuing system theory is used to generate the length of current cycle time.The length derived from summation of interarrival time. The interarrival time is generated from an exponential distribution.

The outputs from the estimation, the number of cars behind the green light, the number of cars behind the red light and the length of current cycle time are used as the fact for a Fuzzy logic system that consists of four components.

1. Fuzzyfier
2. Fuzzy rule based
3. Fuzzy inference engine, and
4. Defuzzifier

The Fuzzyfier component defines membership values of Fuzzy sets according to Kelsey and Bisset (1993), and also the rule based in the Fuzzy rule base component that are composed of 26 rules, which are different from those of rules based on Pappis and Mamdani (1977) who use a set of five rules in their fuzzy logic system.

The Fuzzy inference engine component is based on the product-sum-gravity method presented by Kandel and Langholz (1994). It was used to combine the Fuzzy rules in the fuzzy rules base into a mapping from fuzzy set to fuzzy set . The Defuzzifier component, is based on the center average defuzzifier that was presented by Kandel and Langholz (1994) and is used to perform a mapping from fuzzy set to crisp point.

The crisp point from fuzzy logic is the degree of change. The degree of change has a value between 0 to 1. If the degree of change converges to 0 then the state of the light (phase) remain the same, whereas the state will change to next state if the degree converges to 1.

From the conclusion, as previously mentioned, we can generate traffic flow in a certain time. The traffic flow is composed of the number of cars behind the green light and the number of cars behind the red light at the current moment of time. In addition, the estimation delivers the length of the current cycle time. Finally the optimal length of each phase of the cycle is the length of current cycle time.

The traffic flow outputs under fuzzy controller at each intersections are different. At Ubonchareamsri intersection, it is found that the optimal length of each phase at early cycles (cycle 1-cycle 7) is very short. At late cycles (cycle 8 and beyond) the optimal length of each phase is moderate. The optimal length of phase 2 is likely to be longer than the others. There are few cars behind both the green light and the red light in the early cycle. There is a moderate number of cars behind both the green light and the red light in the late cycle.

At Clock Hall intersection, the traffic flow outputs at early cycles (cycle 1-cycle 8) is found to be not stable. At cycle 1, there are a few cars behind both the green and the red light. In addition, the optimal length is very short. For cycle 2 and cycle 3, the number of cars and the optimal length are moderate. The traffic flow outputs at cycle 4-cycle 8 is just the same as the cycle 1. At late cycles (cycle 9 and beyond), the most optimal lengths are long. The most number of cars behind the green and red light are found many.

For Chonlaprathan intersection, there are only five cycles during the specified time. There are a few cars and very short optimal length on all phases at cycle 1 and cycle 2. For cycle 3,4 and 5 there are many cars behind the green and the red light, and the optimal length is very long on all phases. The traffic flow at Airport intersection is similar to that at Chonlaprathan intersection.

For the traffic flow under the conventional controller, the length of traffic lights on each phase of all cycles are fixed. The results at all intersections are similar; the number of cars are moderate and there are approximate 22 cycles on specific period of time.

This study employs the cost function to evaluate the traffic flow. The cost function involves the average of wait time and drive time, the number of cars exiting and entering the intersection. The efficiency of a traffic controller can be judged from the value of the cost function. The lower the cost function the better performance of the controller.

The comparison of controller performances shows that cost function under the suggested traffic controller is lower than the cost function from conventional controller. This shows that the output of the comparison illustrating the fuzzy controller is more efficient than the conventional controller.

6.2 Discussion

From the literature review, there are many ways to attempt to solve traffic problems. This study concentrates on solving a part of the traffic problem, congestion at intersection. The study accords with these of many authors such as Kotsopoulos (1999), Lan (2002) and so on. A major factor that influences traffic congestion is poor timing. The study improves traffic signal timing at intersections by using mathematical and statistical methods similar to those of Schutter's study (2002) and Yi, Xin and Zhao's study (2001). Fuzzy logic is applied in a way similar to the work of many authors such as Zhenyang's study (2004), Ande'study (1996), Edid's study (1999),Seongho's study (1994), Adeli and Karim's study (2000) , Lee, Krammes and Yen's study (1998) and Cabrera and Ivan's study (2000). The present study ignored the development for the software or hardware of traffic signals. The study did not use high technology tools because of these high cost and the traffic control was unavailable for traffic control in the area of study. The main contribution of the study is the provision of an alternative means to improve the suitable signal timing for traffic controller at the intersections studied by using the optimal length computed by using computer programming by the Fortran language which the police and authorities can apply to solve the traffic problems. The algorithm of computer programming is based on EM algorithm and the Gibbs sample in Markov Chain Monte Carlo, in which demonstrated on many articles such as Herring and Ibrahim (2002), Karlis (2003), Kim and Taylor (1995), Lee and Shi (2001), Carlin, Stern, and Rubin (1995) and so on. The objective of the algorithms is to estimate traffic intensity based on the coordination of the idea of Vardi (1996) and Tebaldi and West (1998). Moreover the study applied queuing theory to identify waiting time, length of queue and the length of the current cycle time similar to the work done by Cheng and Allam (1992), Cruz, Smith and Mediros (2005), Dewees (1979), Das and Levinson (2004), and Omari,Masaeid and Shawaeid (2004). Queuing application in such report papers is mainly based on simulation that is different from this study in that this study only applied queuing to generate interarrival time to calculate waiting time and queue length and the length of the current cycle time. This study also applied fuzzy logic system for traffic control similar to the work of many authors such as Zhenyang' study (2004), Ande;s study (1996), Enid's study (1999), Cabrera and Ivan' study (2000) and so on. Fuzzy logic system designs the algorithm of decision process. The algorithm was designed to change traffic intensity estimator and the length of the current cycle time to degree of change just the same as of the study done by Kelsey and Bisset (1993). The degree of change decided whether to change the state of the traffic light or remain in the same state. In addition, the algorithm was dependent upon an expert traffic control and the membership function that need to be adapted with the observation data (Wang,1994).

The likelihood of the output of traffic flow performance under fuzzy controller at Ubon Charernsri intersection and the performance at Clock Hall intersection derived from the two intersections are close to each other. Additionally, these intersections are in the same traffic environment. The optimal length of traffic signal light on each phase of the late cycles are moderate, because the number of cars that exit and enter the intersections are moderate. This is likely because there are a few cars that exit and enter the intersections at the early cycles, the optimal length of traffic signal light are very short.

The traffic flow performance at Airport and Chonlaprathan intersection gave a similar result in both the number of cars and optimal length of traffic signal light due to their proximity. The optimal length of traffic signal light on all phases is likely to be very long because the Airport intersection has more traffic congestion than the others whereas Chonlaprathan intersection has fewer cars than the others.

Theoretically, the fuzzy logic application to control traffic signal light in other research reports was based on the simulation and the controller installed on the equipment of traffic signal controller differentiated this study from the previous studied. Such difference is that the output from this study do not apply to control traffic flow at the moment. Instead, the process needs data collection and computation by computer programming and then apply traffic timing to control in the next time.

Interestingly, the evaluation by using cost function generation shows that the procedure of this study is very helpful to decrease waiting time and queue length as done by other methods that use high technology equipment.

Summing up, this study presents the mixed method between maximum likelihood estimation and Bayesian estimation to estimate the number of cars that pass all links in the studied traffic system. Moreover this study also let the estimator in the fuzzy logic system to infer the optimal length on each phase at each intersection. The problem and obstacles of this study is that the observation is probably incorrect in some situations, and the study does not cover the improvement of the optimum length in the real situation. Another problem is that this study independently calculated the optimal length at each intersection which may not correspond to the real situation. The future study should link data between each intersection to calculate the optimal length.

References

Abdel-Aty, M. A., & Abdelwahab, H. T. (2004). Predicting injury severity levels in traffic crashes: A modeling comparison. *Journal of Transportation Engineering, 130*(2), 204.

Abutaled, A. S., & Papaioannou, M. G. (2000). Maximum likelihood estimation of time-varying parameters: An application to the athens stock exchange index. *Applied, 32*(10), 1323.

Adeli, H., & Jiang, X. (2003). Neuro-fuzzy logic model for freeway work zone capacity estimation. *Journal of Transportation Engineering, 129*(5), 484.

Adeli, H., & Karim, A. (2000). Fuzzy-wavelet rbfnn model for freeway incident detection. *Journal of Transportation Engineering, 126*(6), 464.

Ait-Sahalia, Y. (2002). Maximum likelihood estimation of discretely sampled diffusions: A closed-form approximation approach. *Econometrica, 70*(1), 223.

Ament, J. M. (1980). Change a queue system from passive to active. *Industrial Engineering, 12*(4), 40.

Ande, M. M. (1996). *Design of fuzzy logic based adaptive traffic signal controller.* University of Nevada, Las Vegas, Las Vegas.

Aquilar-Iqartua, M., Postiqo-Boix, M., & Garcia-Haro, J. (2002). Atm inverse multiplexing. Fundamentals and markovian single-server queue analysis for performance evaluation and validation purposes. *Telecommunication Systems, 21*(1), 103.

Banks, J., Carson, J.S., Nelson, B.L., & Nicol, D. M. (2001). *Discrete-event system simulation.* New Jersey: Prentice-hall.

Basu, S., Banerjee, M., & Sen, A. (2000). Bayesian inference for kappa from single and multiple studies. *Bioinformatics, 56*(2), 577.

Bera, A. K., & Bilias, Y. (2002). The mm, me, ml el, ef and gmm approaches to estimation: A synthesis. *Journal of Applied Econometrics, 107*(1,2), 51.

Beynon, M. J., Pee, M. J., & Tang, Y.-C. (2004). The application of fuzzy decision tree analysis in an exposition of the antecedents of audit fees. *Omega, 32*(3), 231.

Blackwell, P. (2001). Bayesian inference for a random tessellation process. *Biometrics, 57*(2), 502.

Blackwell, P. G. (2003). Bayesian inference for markov processes with diffusion and discrete components. *Biometrika, 90*(3), 613.

Brown, L., Gans, N., Mandelbaum, A., & Sakov, A. (2005). Statistical analysis of a telephone call center: A queueing-science perspective. *Journal of American Statistical Association, 100*(469), 36.

Cabrera, G., & Ivan, L. (2000). *A methodology to design traffic signal controllers based on fuzzy logic.* A methodology to design traffic signal controllers based on fuzzy logic, Puerto Rico.

Carey, V., Baker, C., & Platt, R. (2001). Bayesian inference on protective antibody levels using case-control data. *Biometrics, 57*(1), 135.

Carlin, B. P., & Louis, T. A. (1996). *Bayes and empirical Bayes methods for data analysis.* London: Chapman & Hall.

Casella, G., & George, E.I. (1992) Explaining the Gibbs sampler. *The American Statistician, 46,* 167-174.

Chan, F., & McAleer, M. (2002). Maximum likelihood estimation of star and star-garch models: Theory and monte carlo evidence. *Journal of Applied Econometrics, 17*(5), 509.

Chen, M., & Ibrahim, J. (2001). Maximum likelihood methods for cure rate models with missing covariates. Biometrics, *57*(1), 43.

Chen, M.-H., G.Ibrahim, J., & R.Lipsitz, S. (2002). Bayesian methods for missing covariates in cure rate models. *Lifetime Data Analysis, 8*(2), 117.

Chen, S.-P. (2004). Parametric nonlinear programming for analyzing fuzzy queues with finite capacity. *European Journal of Operational Research, 157*(2), 429.

Cheng, T. C. E., & Allam, S. (1992). A review of stochastic modeling of delay and capacity at unsignalized priority intersections. *European Journal of Operational Research, 60*(3), 247.

Cho, H., & Yi, S.-J. (2004). Vehicle trajectory control using the fuzzy logic controller. *Journal of automobile engineering, 218*(1), 21.

Chopin, N., & Pelqrin, F. (2004). Bayesian inference and state number determination for hidden markov models: An application to the information content of the yield curve about inflation. *Journal of Econometrics, 123*(2), 327.

Corander, J., & Villani, M. (2004). Bayesian assessment of dimensionality in reduced rank regression. *Statistica Neerlandica, 58*(3), 255.

Cruz, F. R. B., MacGreqor, J., & Queiroz, D. C. (2005a). Service and capacity allocation in m/g/c/c state-dependent queuing networks. *Computer & Operations Research, 32*(6), 1545.

Cruz, F. R. B., Smith, J. M., & Medeiros, R. O. (2005b). An m/g/c/c state-dependent network simulation model. *Computer & Operations Research, 32*(4), 919.

Das, S., & Levinson, D. (2004). Queuing and statistical analysis of freeway bottleneck formation. *Journal of Transportation Engineering, 130*(6), 787.

David, w. (1992). How to design fuzzy logic controllers. *Machine Design, 64*(23), 92.

Demey, P., Jean-Frederic, Roget, C., & Poncalli, T. (2004). Maximum likelihood estimate of default correlations. *Risk, 17*(11), 104.

Deschamps, P. J. (1998). Full maximum likelihood estimation of dynamic demand models. *Journal of Applied Econometrics, 82*(2), 335.

Dewees, D. N. (1979). Estimating the time cost of highway congestion. *Econometrica, 47*(6), 1499.

Drekic, S., & Woolford, D. G. (2005). A preemptive priority queue with balking. *European Journal of Operational Research, 164*(2), 387.

Dunson, D., & Herring, A. (2003). Bayesian inferences in the cox model for order-restricted hypotheses. *Biometrics, 59*(4), 916.

Durtham, G. B., Gallant, A. R., Ait-Sahalia, Y., & Brandt, M. W. (2002). Numerical techniques for maximum likelihood estimation of continuous-time diffusion processes / comment. *Journal of Business & Economic, 20*(3), 297.

Duson, D., & Neelon, B. (2003). Bayesian inference on order-constrained parameters in generalized linear models. *Bioinformatics, 59*(2), 286.

Ellis, R. L., & Durgee, D. H. (1982). Coupling queuing with route-advance (part 1). *Telephony, 202*(19), 34.

Ellson, S. R., (1993). *Maximum likelihood estimation logic and practice.* United state of America: Sage publication.

Enid, M. M. (1999). *Fuzzy logic-based control and coordination of traffic signals along an arterial street.* University of Puerto Rico, Mayaguez (Puerto Rico).

Eqorov, A. V., Li, H., & Xu, Y. (2003). Maximum likelihood estimation of time-inhomogeneous diffusions. *Journal of Econometrics, 114*(1), 107.

Erkanli, A., Soyer, R., & Costello, E. (1999). Bayesian inference for prevalence in longitudinal two-phase studies. *Biometrics, 55*(4), 1145.

Fakinos, D. (1982). The expected remaining service time in a single server queue. *Operations Research, 30*(5), 1014.

Fisher, D. (2004). Teaching machines to think fuzzy. *The technology teacher, 64*(2), 13.

Fouqere, D., & Kamionka, T. (2003). Bayesian inference for the mover-stayer model in continuous time with an application to labour market transition data. *Journal of Applied Econometrics, 18*(6), 697.

Frehlich, R., & Sharman, R. (2005). Maximum likelihood estimates of vortex parameters from simulated coherent doppler lidar data. *Journal of Atmospheric and Oceanic Technology, 22*(2), 117.

Fridman, M., & Harris, L. (1998). A maximum likelihood approach for non-gaussian stochastic volatility models. *Journal of Business & Economic Statistics, 16*(3), 284.

Fu, M. C., Hu, J.-Q., & Naqi, R. (1995). Comparison of gradient estimation techniques for queues with non-identical servers. *Computers & Operations Research, 22*(7), 715.

Gelfand, A.E., & Smith, A.F.M. (1990). Sampling-Based approaches to calculating marginal densities. *Journal of the American Statistical Association,* 85, (398-409)

Geweke, J. (1989). Baysian inference in econometric models using monte carlo integration. *Econometrica, 57*(6), 1317.

Geweke, J., Gowrisankaran, G., & Town, R. J. (2003). Bayesian inference for hospital quality in a selection model. *Econometrica, 71*(4), 1215.

Ghitany, M. E., & Al-Awadhi, S. (2002). Maximum likelihood estimation of burr xii distribution parameters under random censoring. *Journal of Applied Statistics, 29*(7), 955.

Gill, R. (2004). Maximum likelihood estimation in generalized broken-line regress. *The Canadian Journal of Statistics, 32*(3), 227.

Glenn, A. (1994). Fuzzy logic: What it is; what it does; what it can do. *Production, 106*(10), 38.

Gorney, L. (1979). Queuing theory, the science of wait control part 2: System types gorney, len. Byte. Peterborough: May 1979.Vol.4, iss. 5; pg. 176. *Byte, 4*(5), 176.

Halachmi, B. (1978). The fokker-planck equation as an approximating formula for the g/m/k queing system. *Computers & Operations Research, 5*(3), 179.

Hagger, C., Janss, L., Kadarmideen, H., & Stranzinger, G. (2004). Bayesian inference on major loci in related multigeneration selection lines of laying hens. *Poultry Science, 83*(12), 1932.

Harb, A. M., & Smadi, I. A. (2004). On fuzzy control of chaotic systems. *Journal of vibration and control, 10*(7), 979.

Herring, A. H., & Ibrahim, J. G. (2002). Maximum likelihood estimation in random effects cure rate models with nonignorable missing covariates. *Biostatistics, 3*(3), 387.

Horton, N., & Laird, N. (2001). Maximum likelihood analysis of logistic regression models with incomplete covariate data and auxiliary information. *Biometrics, 57*(1), 34.

Hsiao-Ching, & Denver, C. (1998, Nov 17). Moving, moving, moving traffic-signal timing plan makes inroads; [rockies edition]. *Denver Post,* p. F.07.

Hsiao, C., Pesaran, M. H., & Tahmiscioqlu, A. K. (2002). Maximum likelihood estimation of fixed effects dynamic panel data models covering short time periods. *Journal of Applied Econometrics, 109*(1), 107.

Huelsenbeck, J. P., Ronguist, F., Nielsen, R., & Bollback, J. P. (2001). Bayesian inference of phylogeny and its impact on evolutionary biology. *Science, 294*(5550), 2310.

Hunter, D. R., & lange, K. (2004). A tutorial on mm algorithms. *The American Statistician, 58*(1), 30.

Jaiimsin, A. (2004, Nov 29). Overhauled mass transit system for bangkok to include rapid routes. *Knight Ridder Tribune Business New,* p. 1.

Jaqannathan, R., & Khan, A. M. (2001). Methodology for the assessment of traffic adaptive control systems. *Institue of Transportation Engineers.ITE Journal, 71*(6), 28.

Jensen, M. J. (2004). Semiparametric bayesian inference of long-memory stochastic volatility models. *Journal ot Time Series Analysis, 25*(6), 895.

Jewell, N. P. (2004). Maximum likelihood estimation of ordered multinomial parameters. *Biostatistics, 5*(2), 291.

Jonker, M. A. (2003). Maximum likelihood estimation of life-span based on censored and passively registered historical data. *Lifetime Data Analysis, 9*(1), 35.

Kendel, A., & Langholz, G. (1994). *Fuzzy control systems.* United State of America: CRC Press.

Karlis, D. (2001). A general em approach for maximum likelihood estimation in mixed poisson regression models. *Statistica Modelling, 1*(4), 305.

Karlis, D. (2003). An em algorithm for multivariate poisson distribution and related models. *Journal of Applied Statistics, 30*(1), 63.

Keats, J. B., Lawrence, F. P., & Wang, F. K. (1997). Weibull maximum likelihood parameter estimates with censored data. *Journal of Quality Technology, 29*(1), 105.

Kelsey, R. L., & Bisset, K. R. (1993). Simulation of traffic flow and control using fuzzy and conventional methods. In M. Jamshidi (Ed.), *Fuzzy and control* (pp. 262-278). New Jersey: Prentrice-Hall.

Kerbache, L., & Smith, J. M. (2004). Queueing networks and the topological design of supply chain systems. *Interational journal of Production Economics, 91*(3), 251.

Kikuchi, S., & Tanaka, N. U. M. (2003). Impacts of shorter perception-reaction time of adapted cruise controlled vehicles on traffic flow and safety. *Journal of Transportation Engineering, 129*(146), 146.

Kim, D., & Taylor, J. M. G. (1995). The restricted em algorithm for maximum likelihood estimation under linear restrictions on the parameters. *Journal of American Statistical Association, 90*(430), 708-715.

Kirawanich, P., & O'Connell, R. M. (2004). Fuzzy logic control of an active power line conditioner. *IEEE Trasactions on power electronics, 19*(6), 1574.

Kleiberqen, F. (2004). Invariant bayesian inference in regression models that is robust against the jeffreys-lindley's paradox. *Journal of Econometrics, 123*(2), 227.

Kleinrock, L. (1976). *Queuing systems.* United State of America: John Wiley & Son.

Kotsopoulos, N. (1999, Jul 2 1999). Signal upgrade at rice square smoothes 4-street traffic flow; [final edition]. *Telegram&Gazette*, p. B.1.

Kotz, S., Kozubowski, T. J., & Podqorski, K. (2002). Maximum likelihood estimation of asymmetric laplace parameters. *Annals of the Institute of Statistical Mathematics, 54*(4), 816.

Lan, C.-J. (2004). New optimal cycle length formulation for pretimed signals at isolated intersections. *Journal of Transportation Engineering, 130*(5), 637.

Lazar, N. A. (2003). Bayesian empirical likelihood. *Biometrika, 90*(2), 319.

Ledford, J. (2002). Well-timed signals cut delays, fuel use; [home edition]. *The Atlanta Journal-Constitution,* C.2.

Lee, S., Krammes, R. A., & Yen, J. (1998). Fuzzy-logic-based incident detection for signalized diamond interchanges. *Transportation Research Part C.*

Lee, S., Messer, C. J., Oh, Y., & lee, C. (2004). Assessment of three simulation models for diamond interchange analysis. *Journal of Transportation Engineering, 130*(3), 304.

Lee, S., & Shi, J. (2001). Maximum likelihood estimation of two-level latent variable models with mixed continuous and polytomous data. *Biometrics, 57*(3), 787.

Leonard, D, J., Rodegerdts, & A, L. (1998). Comparison of alternate signal timing policies. *Journal of Transportation Engineering, 124*(6), 510.

Liu, J., & Sabatti, C. (2000). Generalised gibbs sampler and multigrid monte carlo for bayesian computation. *Biometrika, 87*(2), 353.

Liu, J. S., & Lawrence, C. E. (1999). Bayesian inference on biopolymer models. *Bioinformatics, 15*(1), 38.

Liu, L., Liu, X., & Yao, D. D. (2004). Analysis and optimization of a multistage inventory-queue system. *Management Science, 50*(3), 365.

Lo, H. K., & Chow, A. H. F. (2004). Control strategies for oversaturated traffic. *Journal of Transportation Engineering, 130*(4), 466.

Lynch, T. B., Nkouka, J., Huebschmann, M. M., & Guldin, J. M. (2003). Maximum likelihood estimation for predicting the probability of obtaining variable shortleaf pine regeneration densities. *Forest Science, 49*(4), 577.

Mahmoud, M. E.-D., & EL-Araby, K. (1999). A robust dynamic highway traffic simulation model. *Computers & Industrial Engineering, 37*(1,2), 189.

Maqlaras, C., & Mieqhem, J. A. V. (2005). Queueing systems with leadtime constraints: A fluid-model approach for admission and sequencing control. *European Journal of Operation Research, 167*(1), 179.

Mamdani, E.H. (1974) Applications of fuzzy algorithms for control of a simple dynamic Plant. *Proc. of IEEE,*121, (1585-1588)

Martin, A. D. (2003). Bayesian inference for heterogeneous event counts. *Sociological Methods and Research, 32*(1), 30.

Masalonis, A. J., & Parasuraman, R. (2003). Fuzzy signal detection theory: Analysis of human and machine performance in air traffic control, and analytic considerations. *Ergonomics, 46*(11), 1045.

Milescu, L. S., Akk, G., & Sachs, F. (2005). Maximum likelihood estimation of ion channel kinetics from macroscopic currents. *Biophysical Journal, 88*(4), 2494.

Milligan, B. G. (2003). Maximum-likelihood estimation of relatedness. *Genetics, 163*(3), 1153.

Miranda, M. J., & Rui, X. (1997). Maximum likelihood estimation of the nonlinear rational expectations asset pricing model. *Journal of Economic Dynamics & Control, 21*(8,9), 1493.

Moore, D. S. (1997). Bayes for beginners? Some reasons to hesitate. *The American Statistician, 51*(3), 254.

Nair, V. N., Tang, B., & Xu, L.-A. (2001). Bayesian inference for some mixture problems in quality and reliability. *Journal of Quality Technology, 33*(1), 16.

Nam, D. H., & R., D. D. (1998). Analyzing freeway traffic under congestion: Traffic dynamics approach. *Journal of Transportation Engineering, 124*(3), 208.

Ning-Zhong, Zneng, S.-R., & Guo, J. (2005). The restricted em algorithm under inequality restrictions on the parameters. *Journal of Multivariate Analysis, 92*(1), 53.

O'Loughlin, C., & Coenders, G. (2004). Estimation of the european customer satisfaction index: Maximum likelihood versus partial least squares. Application to postal services. *Total Quality Management & Business Excellence, 15*(9,10), 1231.

Odejar, M. A. E., & S.McNulty, M. (2001). Bayesian analysis of the stochastic switching regression model using markov chain monte carlo methods. *Computational Economics, 17*(2-3), 265.

Oh, M.-S., Choi, J. W., & Kim, D.-G. (2003). Bayesian inference and model selection in latent class logit models with parameter constraints: An application to market segmentation. *Journal of Applied Statistics, 30*(2), 191.

Paiqe, R. L., & Butler, R. W. (2001). Bayesian inference in neural networks. *Biometrika, 88*(3), 623.

Pasquale, F., Barone, P., Sebstiani, G., & Stander, J. (2004). Bayesian analysis of dynamic magnetic resonance breast images. *Applied Statistics, 53*(3), 475.

Piles, M., Gianola, D., Varona, L., & Blasco, A. (2003). Bayesian inference about parameters of a longitudinal trajectory when selection operates on a correlated trait1. *Journal of Animal Science, 81*(11), 2714.

Porter, J. (2002). Efficiency of covariance matrix estimators for maximum likelihood estimation. *Journal of Business & Econometrics Statistics, 20*(3), 431.

Rakha, H., & Zhang, Y. (2004). Sensitive analysis of transit signal priority impacts on operation of a signalized intersection. *Journal of Transportation Engineering, 130*(6), 796.

Ramasamy, N. R., & Selladurai, V. (2004). Fuzzy logic approach to prioritise engineering characteristics in quality function deployment (fl-qfd). *The international journal of quality & Reliability management, 21*(9), 1012.

Roberts, G. O., Papaspiliopoulos, O., & Dellaportas, P. (2004). Bayesian inference for non-gaussian ornstein-uhlenbeck stochastic volatility processes. *Journal of the Royal Statistical Society. Series B. Statistical Methodology, 66*(2), 369.

Rolls, D. A., Michailidis, G., & Hernandez-Campos, F. (2005). Queueing analysis of network traffic: Methodology and visualization tools. *Computer Networks, 48*(3), 447.

Ross, A. M., & Shanthikumar, J. G. (2005). Estimating effective capacity in erlang loss systems under competition. *Queueing Systems, 49*(1), 23.

Rous, J. J., Jewell, R. T., & Brown, R. W. (2004). The effect of prenatal care on birthweight: A full-information maximum likelihood approach. *Health Economics, 13*(3), 251.

Rovers, M., Vander, W. G., Vander, B. S., Straatman, H., Ingels, K., & Zielhuis, G. (2005). Bayes' theorem: A negative example of a rct on grommets in children with glue ear. *European Journal of Epidemiology, 20*(1), 23.

Rubinstein, R. Y. (1981). *Simulation and the Monte Carlo method.* United State of American: John Wiley & Sons.

Sarkar, D., & Zangwill, W. I. (1992). File and work transfers in cyclic queue systems. *Management Science, 38*(10), 1510.

Scheike, T. H., & Martinussen, T. (2004). Maximum likelihood estimation for cox's regression model under case-cohort sampling. *Scandinavian Journal of Statistics, 31*(2), 283.

Schutter, B. D. (2002). Optimizing acyclic traffic signal switching sequences through an extended linear complementarity problem formulation. *European Journal of Operational Research, 139*(2), 400.

Seongho, K. (1994). *Applications of petri networks and fuzzy logic to advanced traffic management systems.* Polytechnic University, New York.

Soud, D., & Kazemian, H. B. (2004). A fuzzy approach to active usage parameter control in ieee 802.11b wireless networks. *Expert Sytem, 21*(5), 269.

Stewart, D., Cheraghi, S., & Malzahn, D. (2004). Fuzzy defect avoidance system (fdas) for product defect control. *Interational journal of production research, 42*(16), 3159.

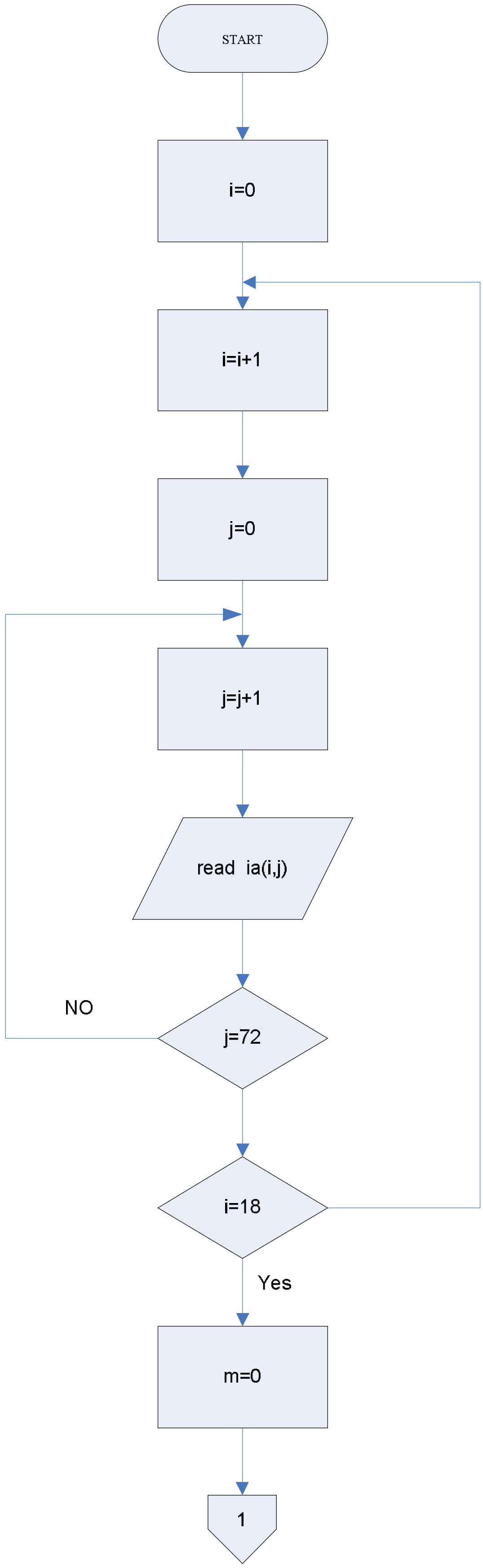
Sunkari, S. (2004). The benefits of retiming traffic signals. *74*(4), 26.

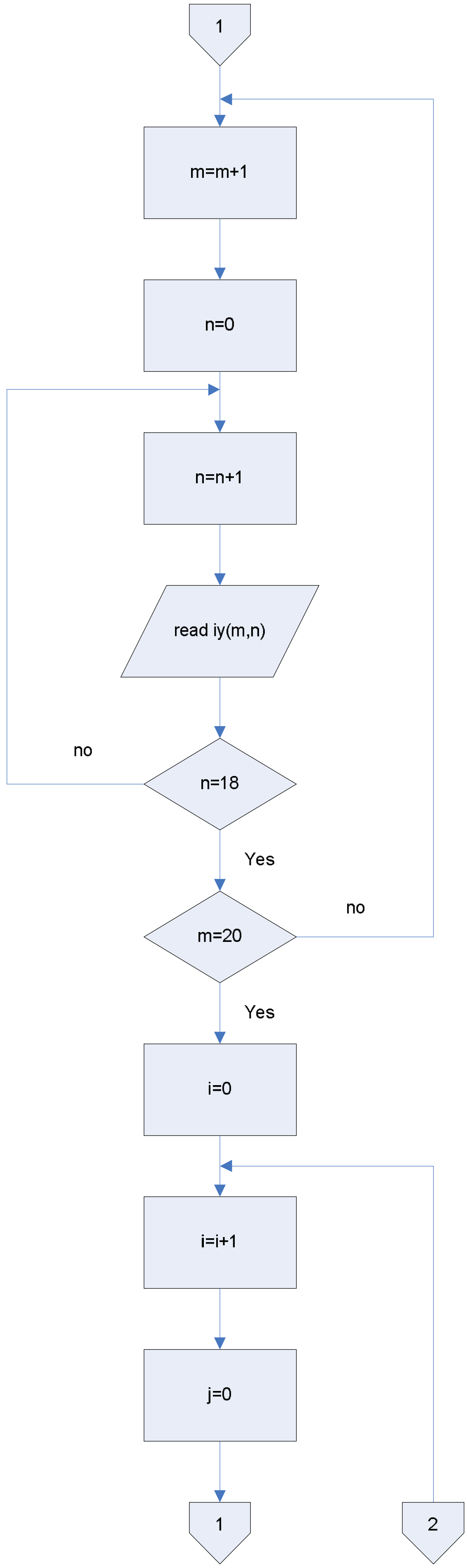
Swann, C. A. (2002). Maximum likelihood estimation using parallel computing: An introduction to mpi. *Computational Economics, 19*(2), 145.

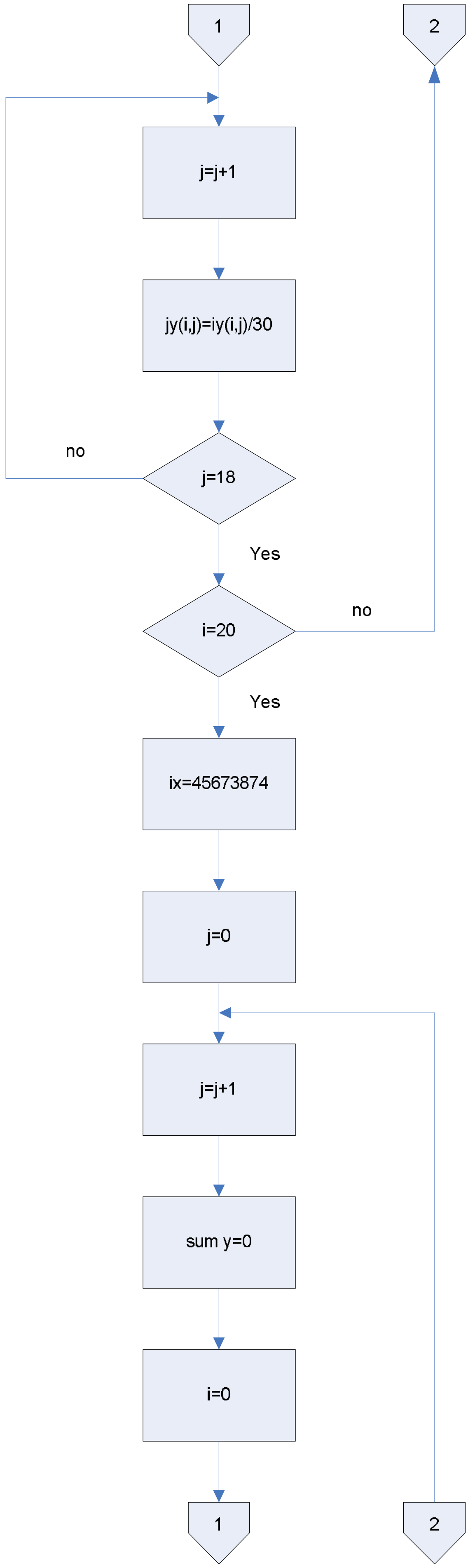
Takine, T. (2005). Single-server queues with markov-modulated arrivals and service speed. *Queueing Systems, 49*(1), 7.
Tebaldi, C., & West, M. (1998). Bayesian inference on network traffic using link count data. *American statistical association journal of American statistical association, 93,* 557-573.
Teodorovic, D., & Vukadinovic, K. (1998). *Traffic control and transport plannig.* Massachusetts: Kluwer academic publishers.
Toshinori, M., & Yashvant, J. (1994). Fuzzy systems: An overview. *Association for Computing Machinery. Communications of the ACM, 37*(3), 68.
Vardi, Y. (1996). Work tomography: Estimating source-destination traffic intensity from link data. *Journal of American Statistical Association, 91,* 365-377.
VerevKa, O. V., & Parasyuk, I. N. (2002). Mathematical fundamentals of constructing fuzzy bayesian inference techniques. *Cybernetics and Systems Analysis, 38*(1), 89.
Wang, L.-X. (1994). *Fuzzy systems and control. New Jersey:* United States of America.
Wang, X., He, C. Z., & Sun, D. (2005). Bayesian inference on the patient population size given list mismatches. *Statistics in Medicine, 24*(2), 249.
Waqner, K., & Gill, J. (2005). Bayesian inference in public administration research: Substantive differences from somewhat different assumptions. *Interational Journal of Public Administration, 28*(1,2), 5.
Yang, T., Lee, R.-S., Chen, M.-C., & Chen, P. (2005). Queueing network model for a single-operator machine interference problem with external operations. *European Journal of Operation Research, 167*(1), 163.
Yi, P., Xin, C., & Zhao, Q. (2001). Implementation and field testing of characteristics-based intersection queue estimation model. *Networks and Spatial Economics, 1*(1-2), 205.
Yu, Q., & Wong, G. Y. C. (2005). Modified semiparametric maximum likelihood estimator in linear regression analysis with complete data or right-censored data. *Technometrics, 47*(1), 34.
Zhang, H., & Tam, C. (2004). Fuzzy simulation of flexible construction processes. *Interational journal of computer applications in technology, 20*(1-3), 15.
Zhenyang, L. (2004). *Dynamic left-turn phase optimization using fuzzy logic control.* The University of Tennessee, Tennessee.
Zhu, Y., & Zhang, Z. G. (2004). M/gi/1 queues with services of both positive and negative customers. *Journal of Applied Probability, 41*(4), 1157.

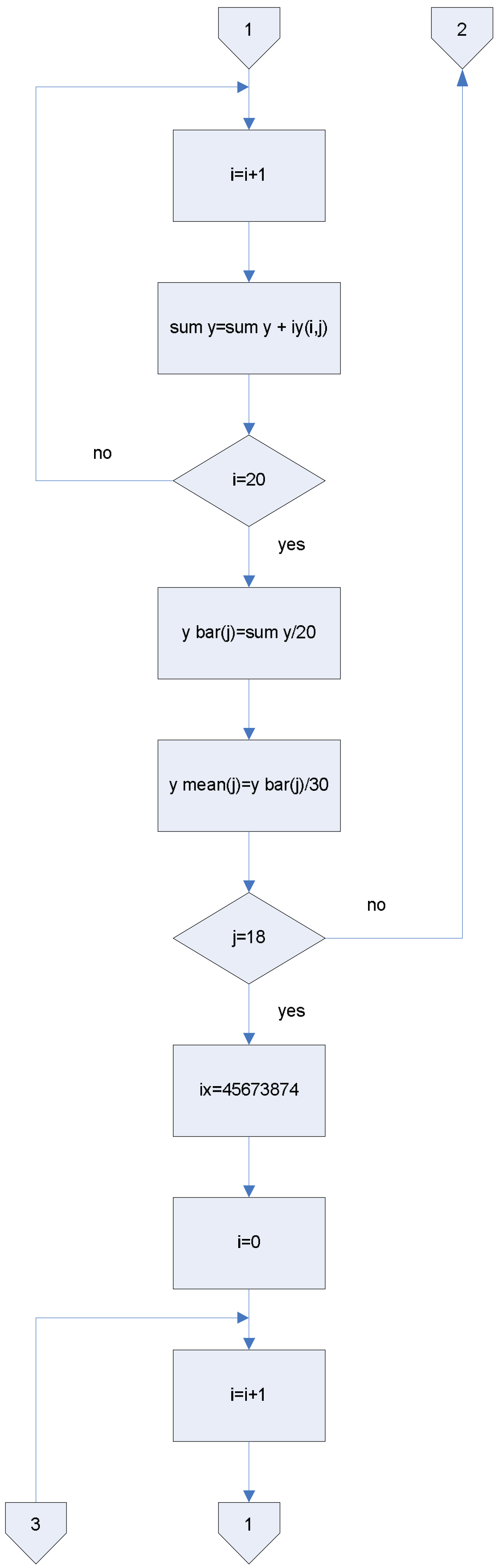
Appendix

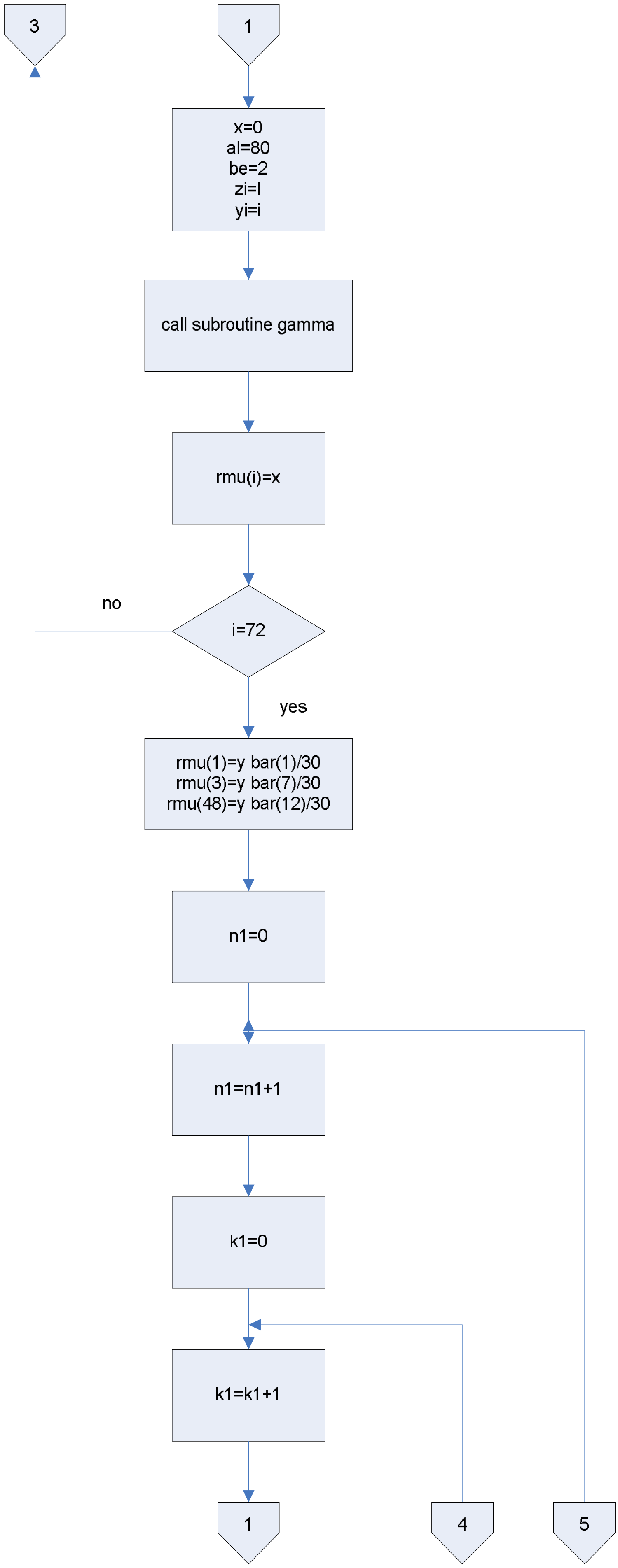
The flowchart for main program

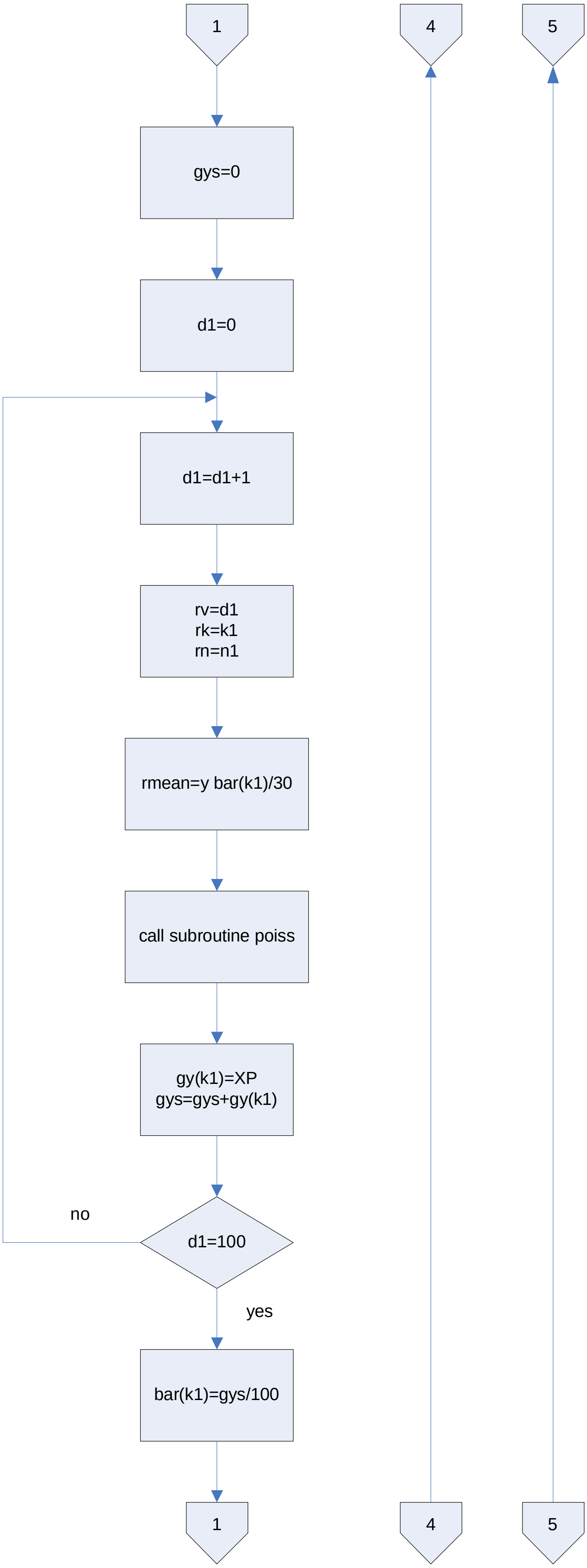


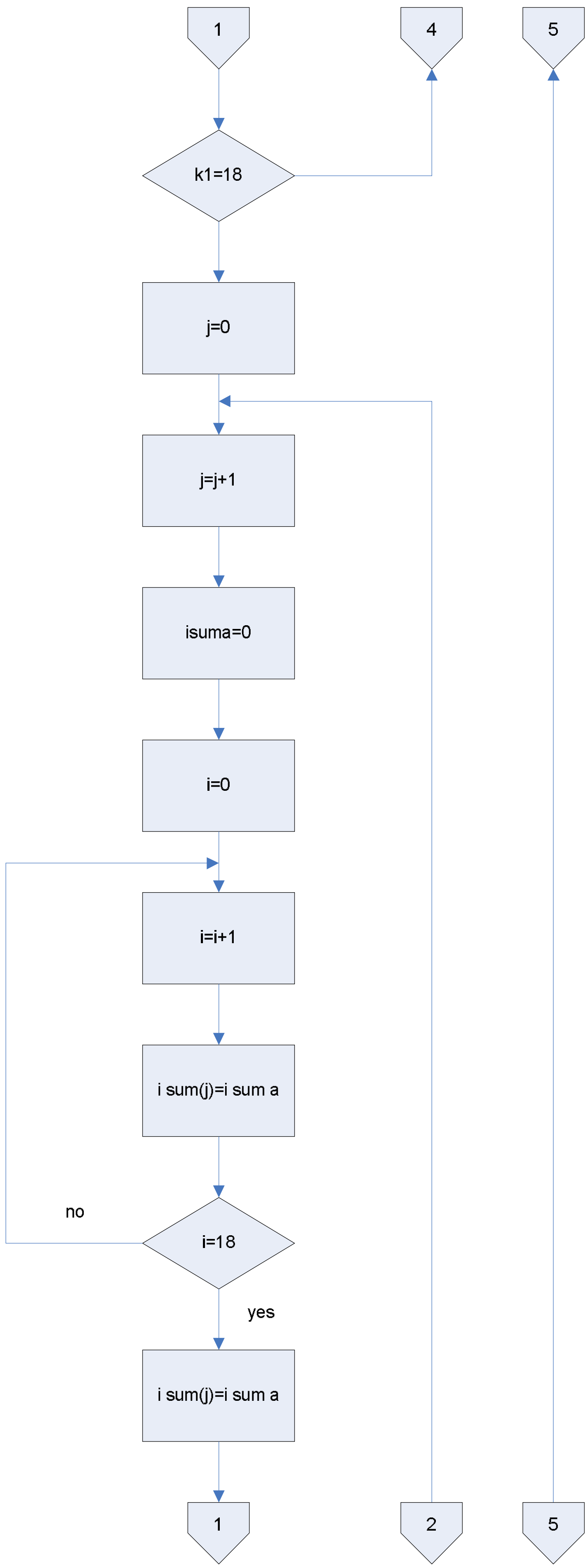

1
m=m+1
n=0
n=n+1
read iy(m,n)
no
n=18
Yes
no
m=20
Yes
i=0
i=i+1
j=0
1
2

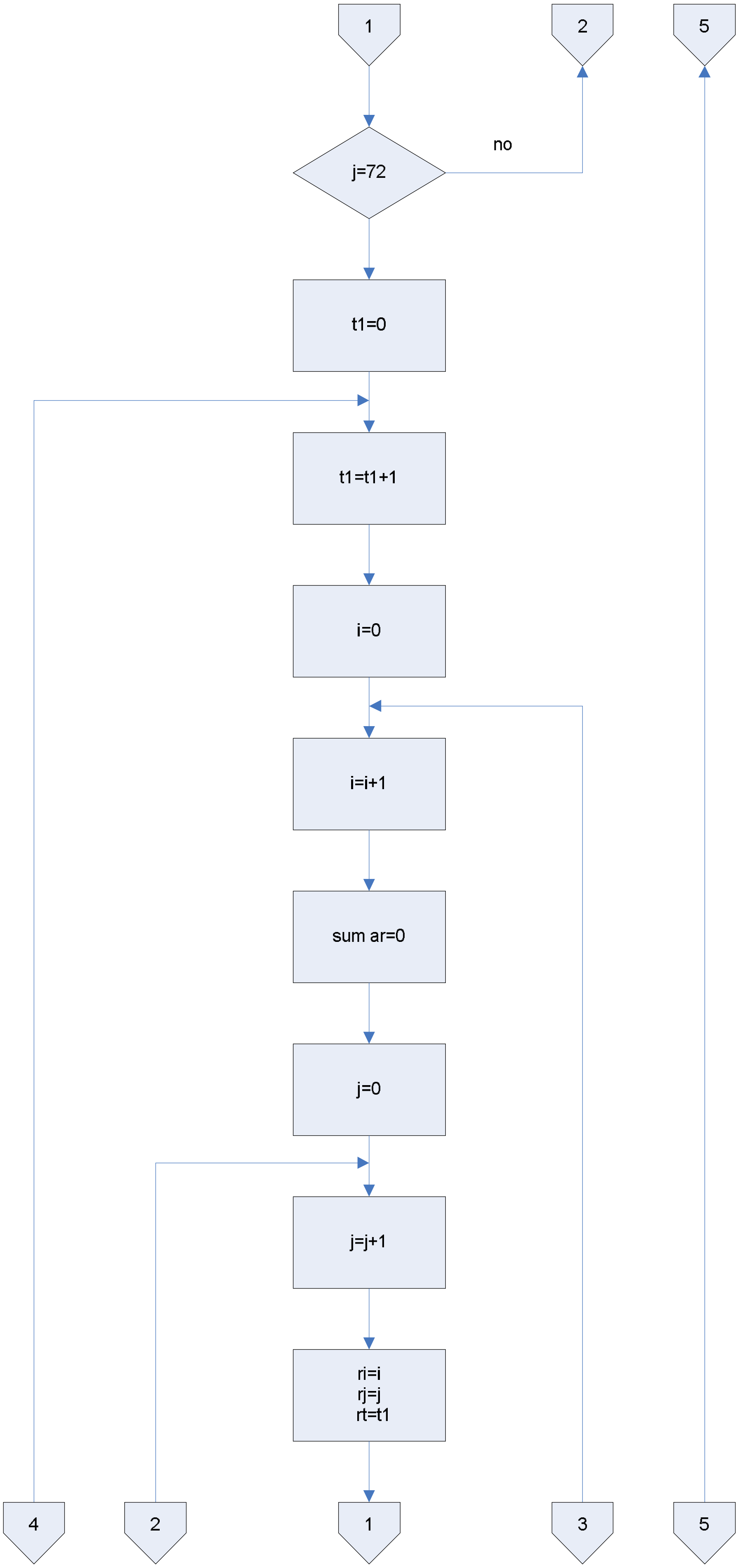

1
2
j=j+1
jy(i,j)=iy(i,j)/30
no
j=18
Yes
no
i=20
Yes
ix=45673874
j=0
j=j+1
sum y=0
i=0
1
2

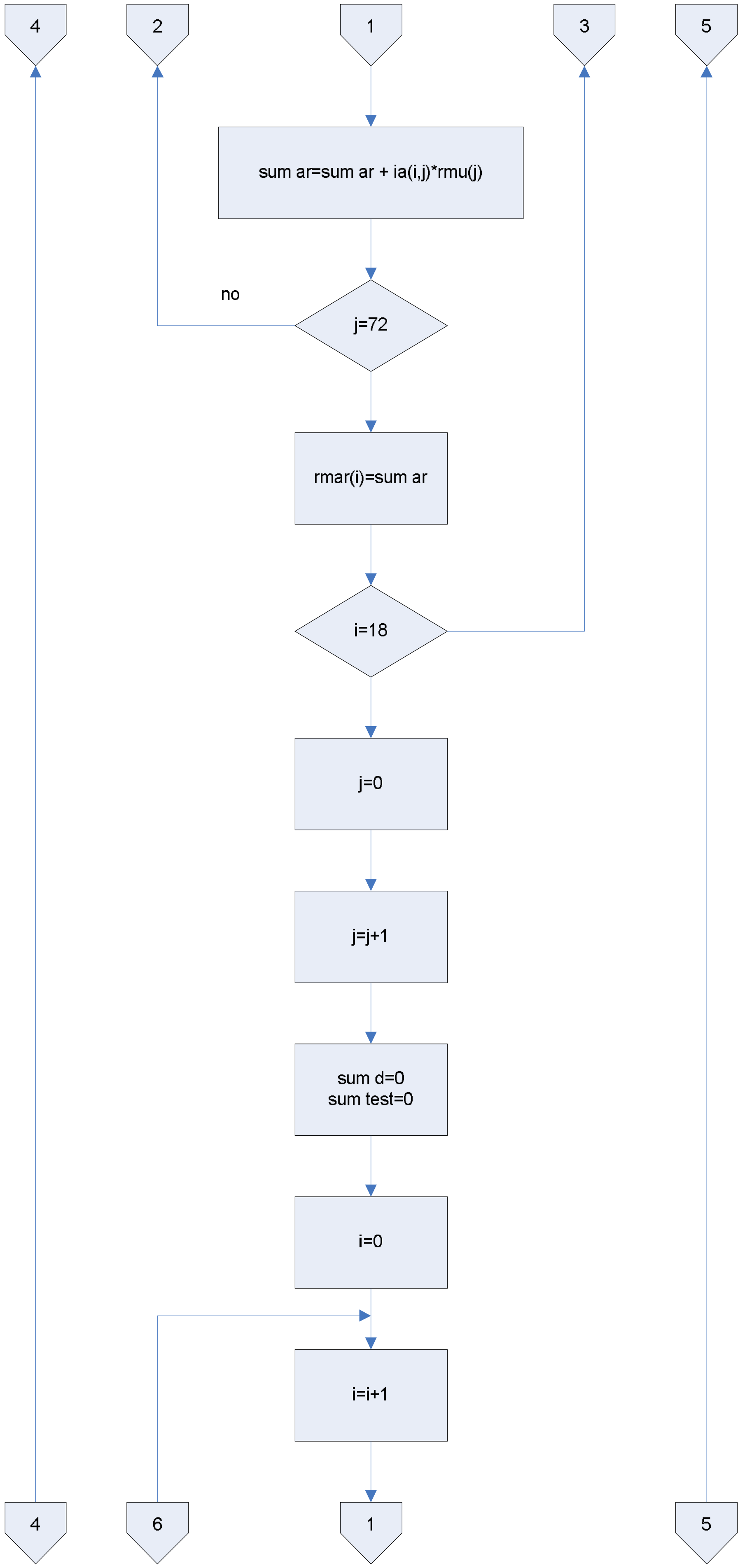

1
2
i=i+1
sum y=sum y + iy(i,j)
no
i=20
yes
y bar(j)=sum y/20
y mean(j)=y bar(j)/30
no
j=18
yes
ix=45673874
i=0
i=i+1
3
1

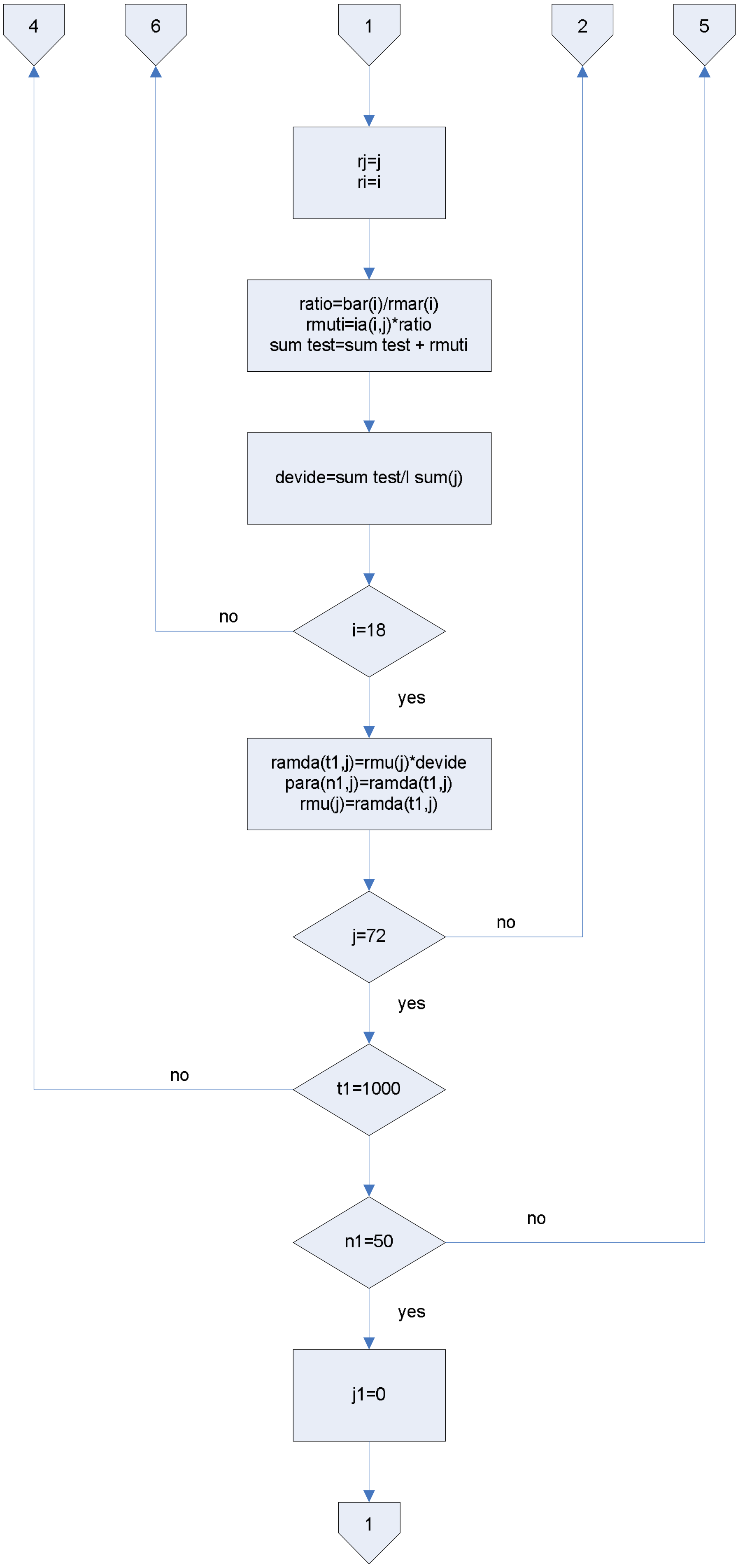

3
1
x=0
al=80
be=2
zi=l
yi=i
call subroutine gamma
rmu(i)=x
no
i=72
yes
rmu(1)=y bar(1)/30
rmu(3)=y bar(7)/30
rmu(48)=y bar(12)/30
n1=0
n1=n1+1
k1=0
k1=k1+1
1
4
5

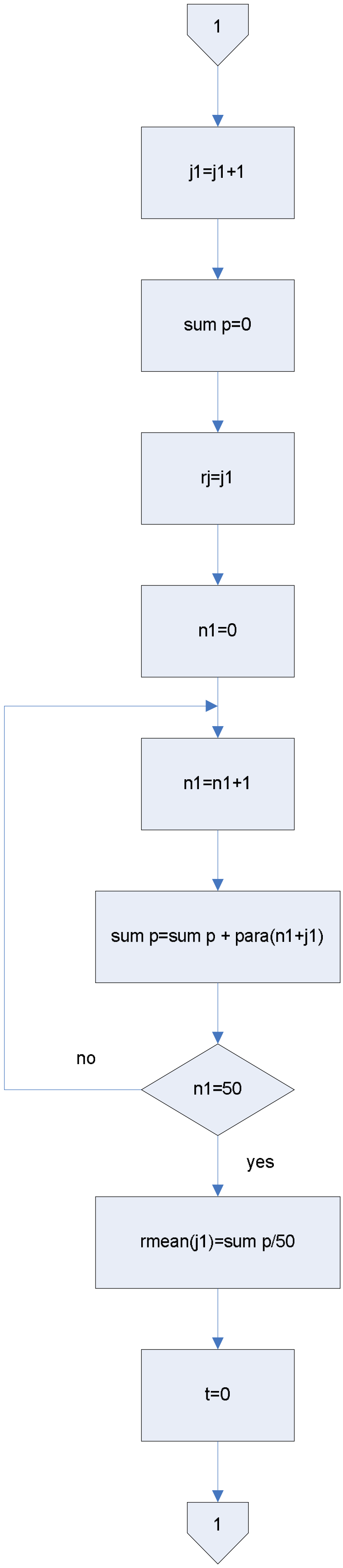

1
4
5
gys=0
d1=0
d1=d1+1
rv=d1
rk=k1
rn=n1
rmean=y bar(k1)/30
call subroutine poiss
gy(k1)=XP
gys=gys+gy(k1)
no
d1=100
yes
bar(k1)=gys/100
1
4
5

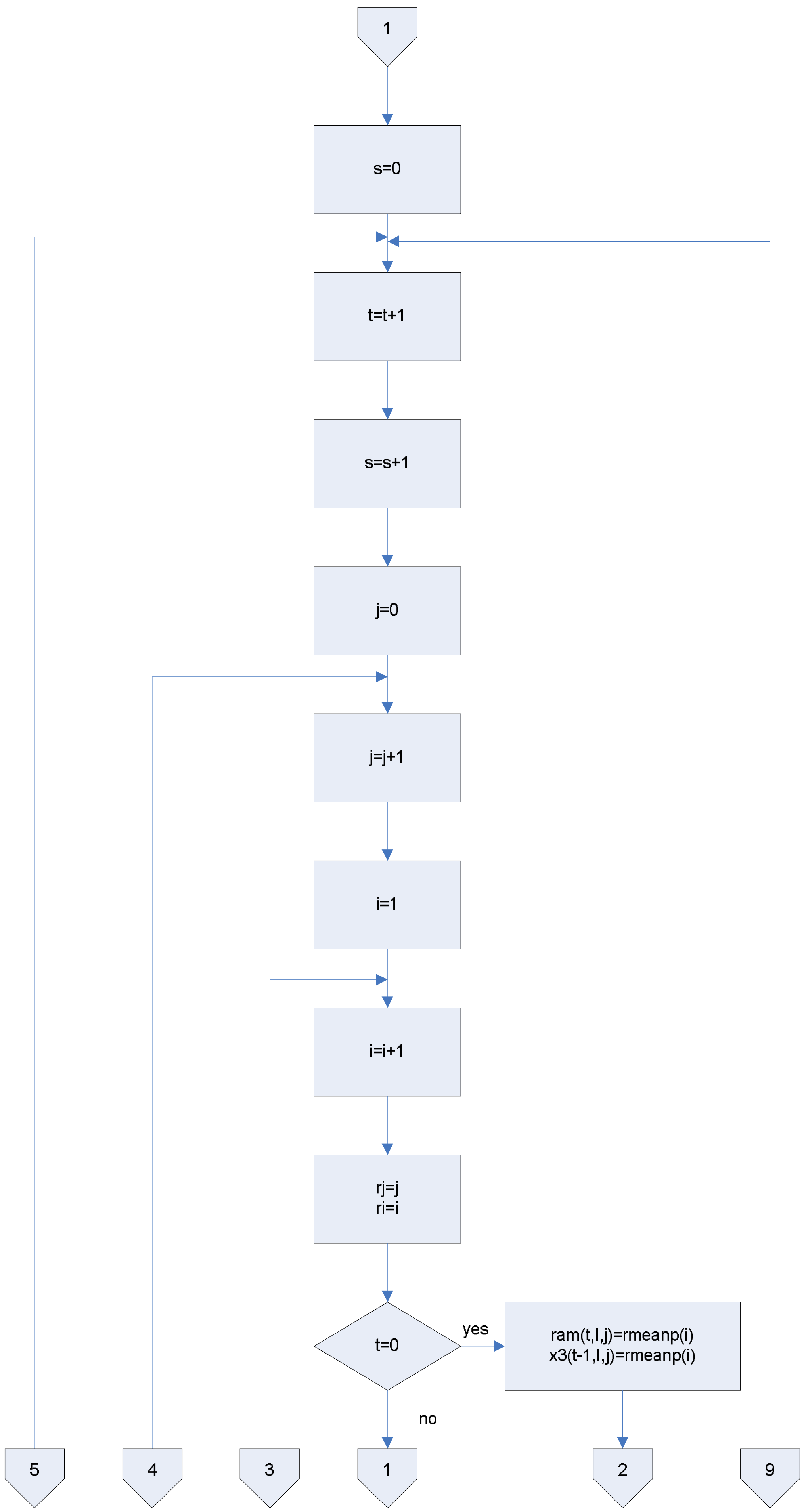

1
4
5
k1=18
j=0
j=j+1
isuma=0
i=0
i=i+1
i sum(j)=i sum a
no
i=18
yes
i sum(j)=i sum a
1
2
5

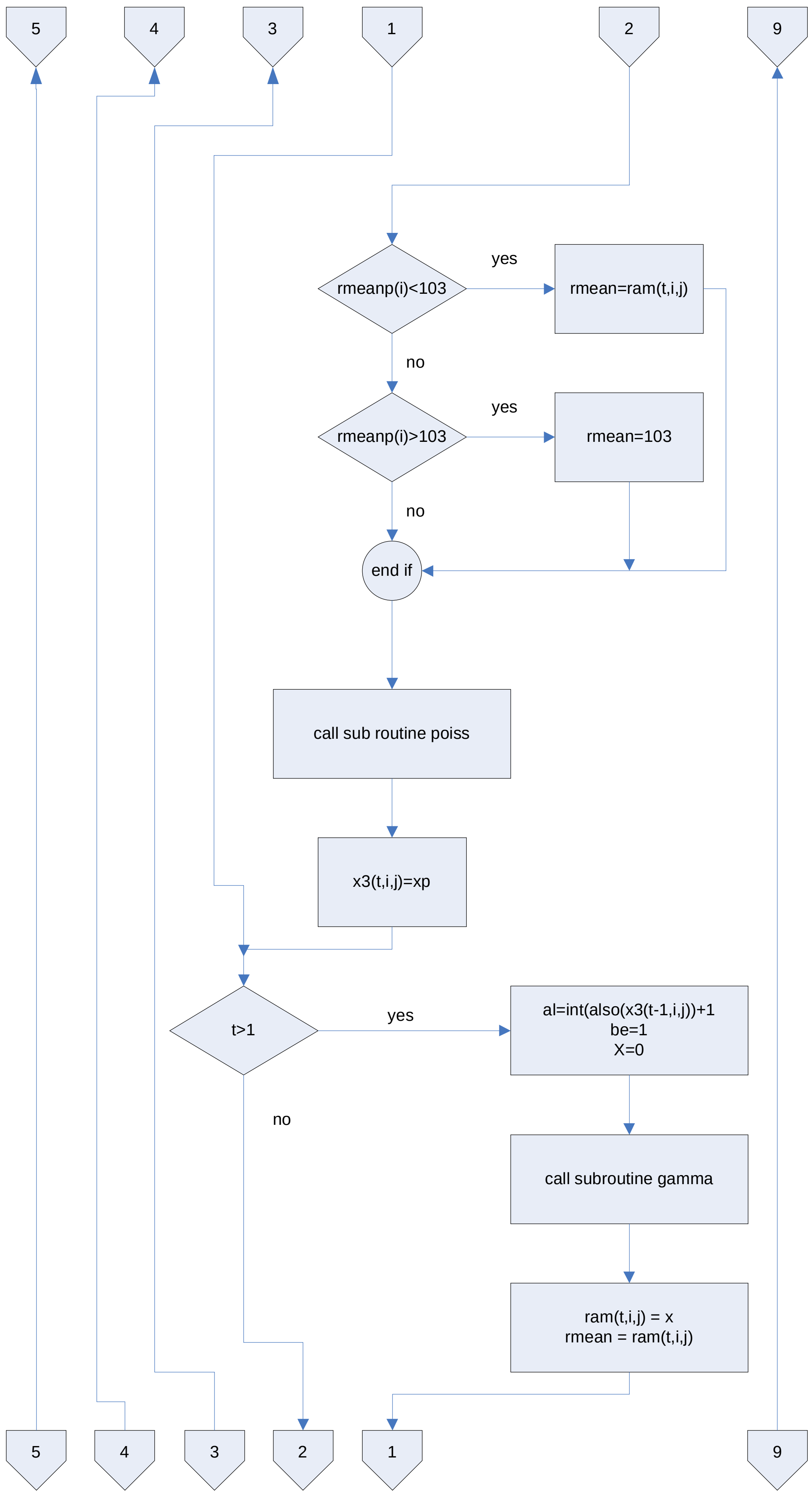

1
2
5
j=72
no
t1=0
t1=t1+1
i=0
i=i+1
sum ar=0
j=0
j=j+1
ri=i
rj=j
rt=t1
4
2
1
3
5

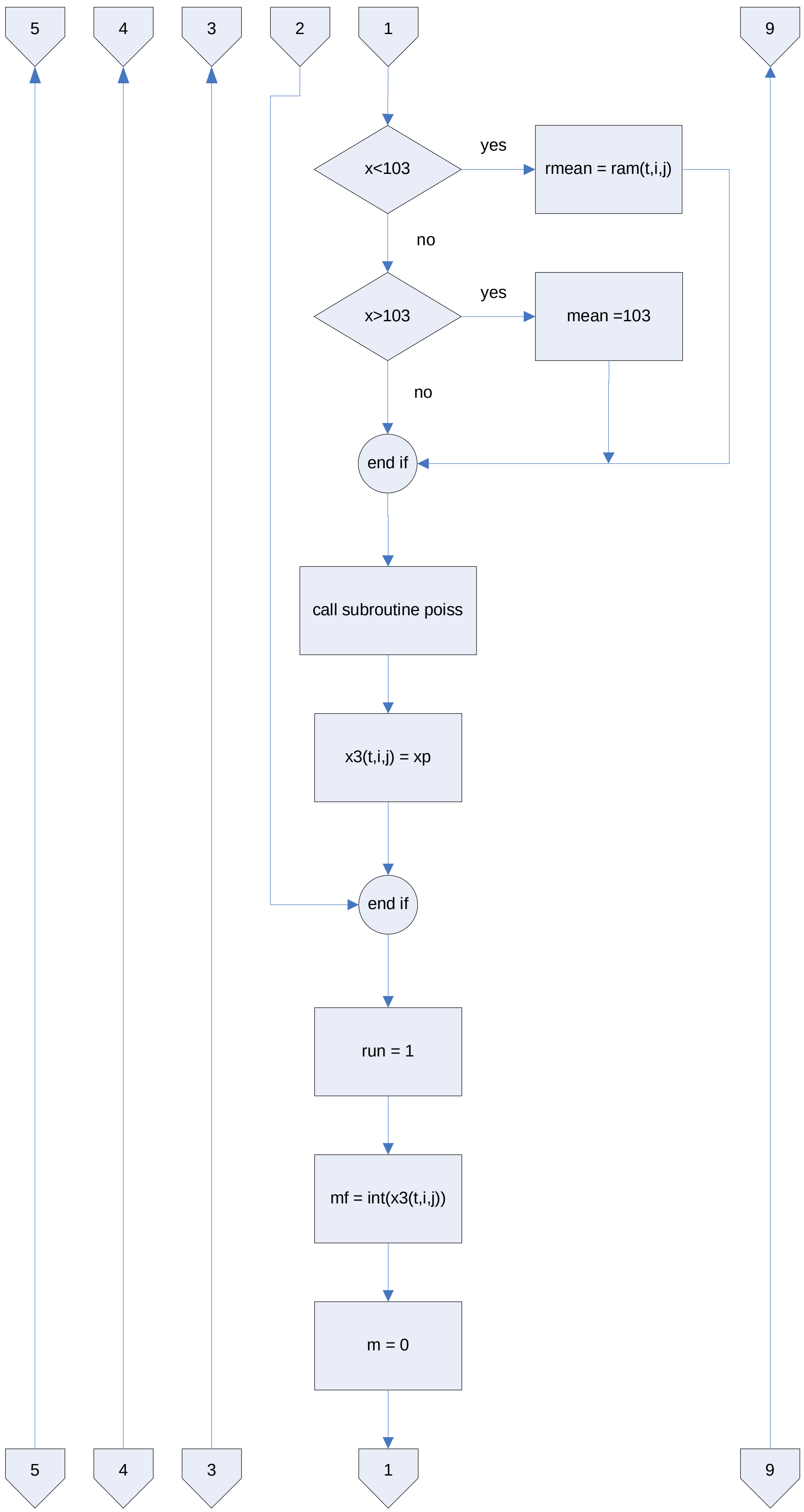

4
2
1
3
5
sum ar=sum ar + ia(i,j)*rmu(j)
no
j=72
rmar(i)=sum ar
i=18
j=0
j=j+1
sum d=0
sum test=0
i=0
i=i+1
4
6
1
5

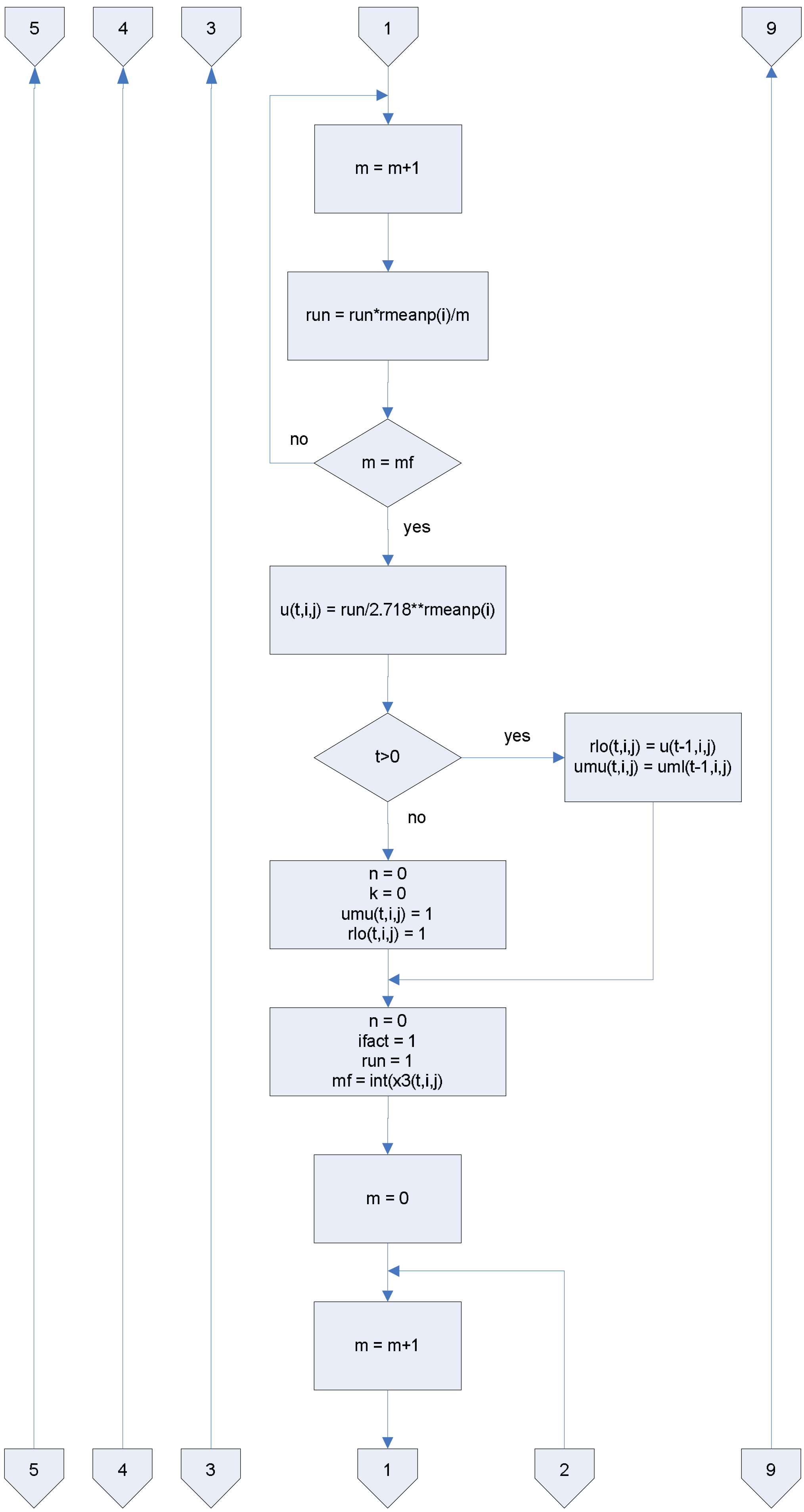

4
6
1
2
5
rj=j
ri=i
ratio=bar(i)/rmar(i)
rmuti=ia(i,j)*ratio
sum test=sum test + rmuti
devide=sum test/l sum(j)
no
i=18
yes
ramda(t1,j)=rmu(j)*devide
para(n1,j)=ramda(t1,j)
rmu(j)=ramda(t1,j)
j=72
no
yes
no
t1=1000
n1=50
no
yes
j1=0
1

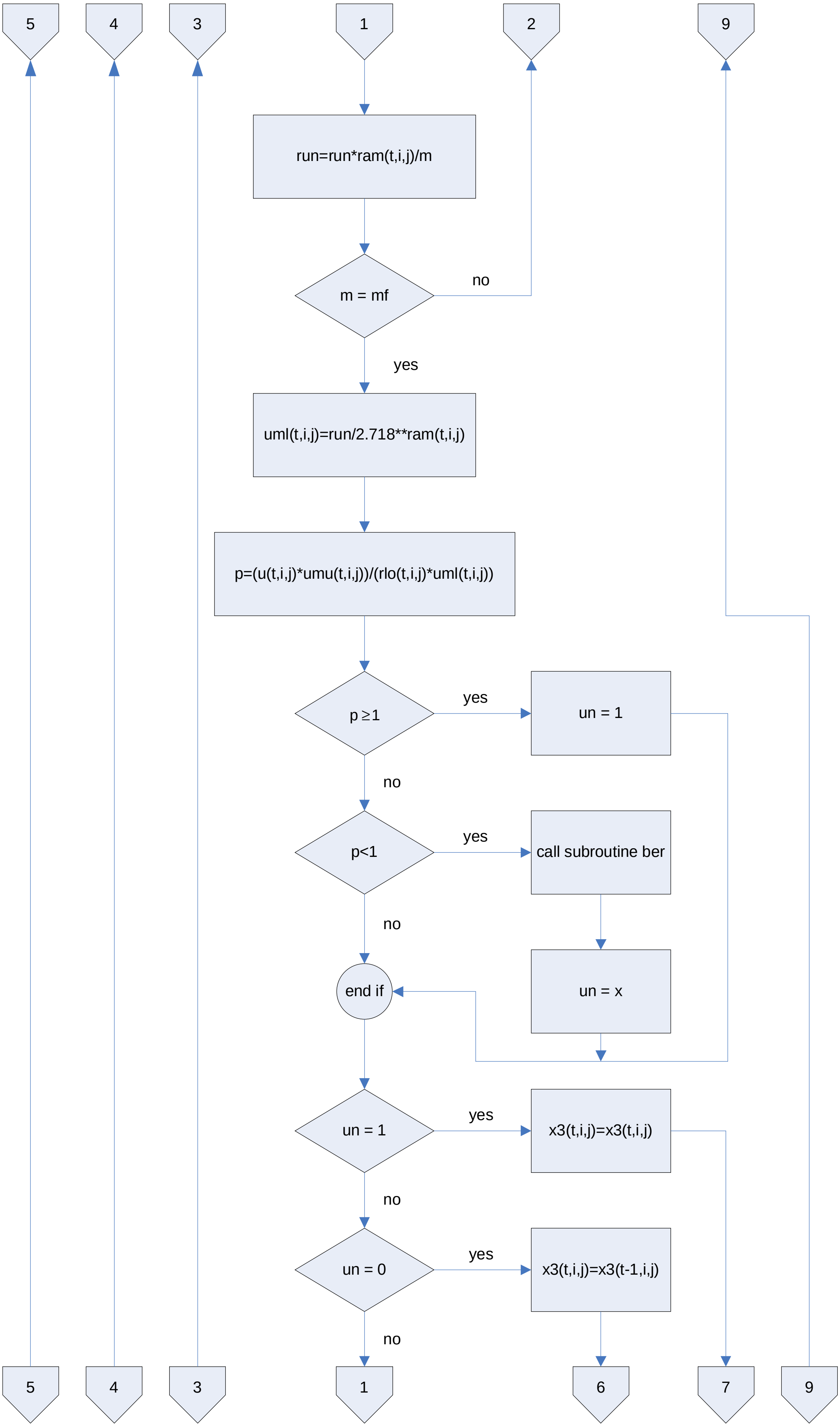

1
j1=j1+1
sum p=0
rj=j1
n1=0
n1=n1+1
sum p=sum p + para(n1+j1)
no
n1=50
yes
rmean(j1)=sum p/50
t=0
1

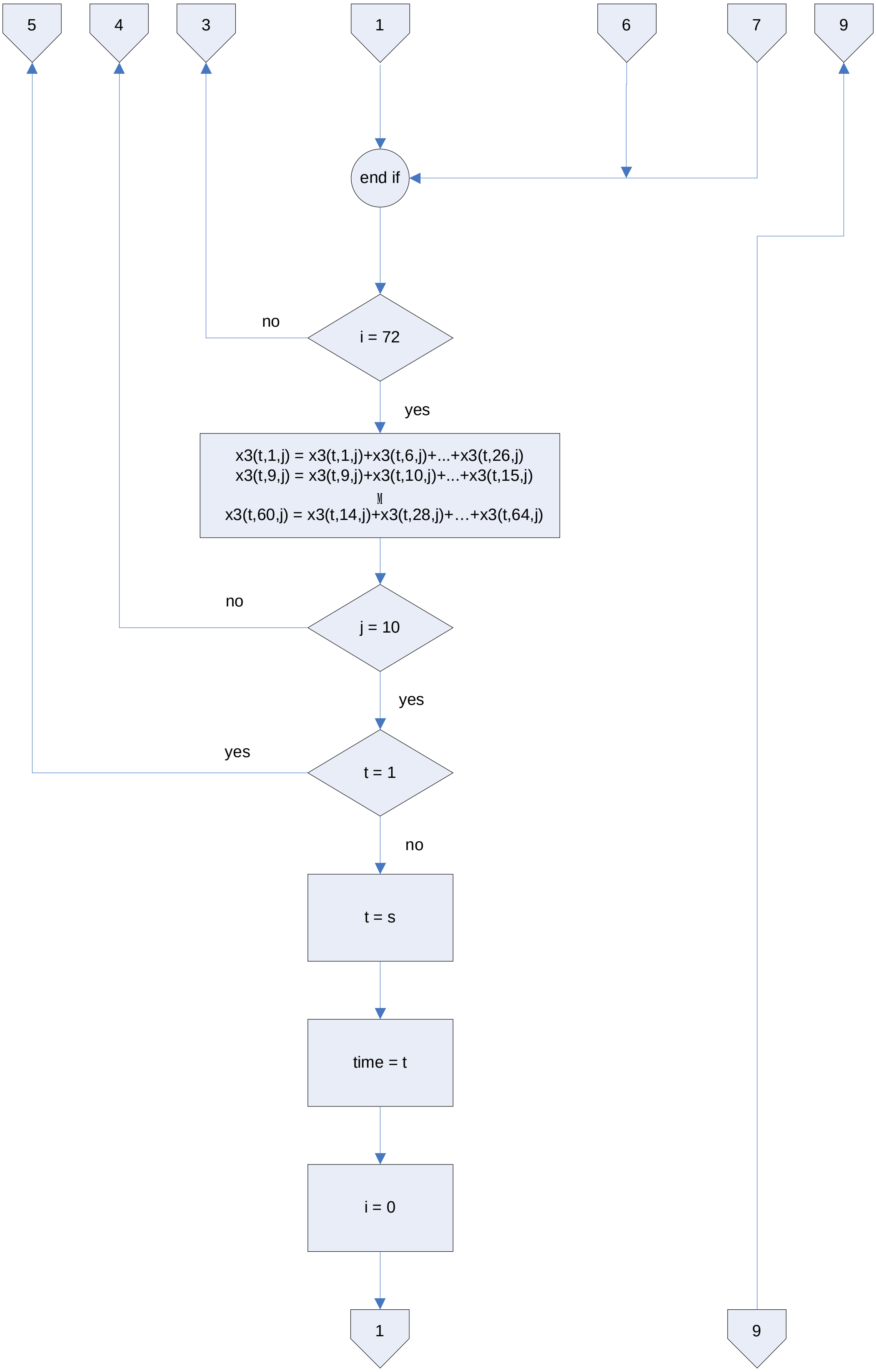

1
s=0
t=t+1
s=s+1
j=0
j=j+1
i=1
i=i+1
rj=j
ri=i
t=0
yes
ram(t,l,j)=rmeanp(i)
x3(t-1,l,j)=rmeanp(i)
no
5
4
3
1
2
9

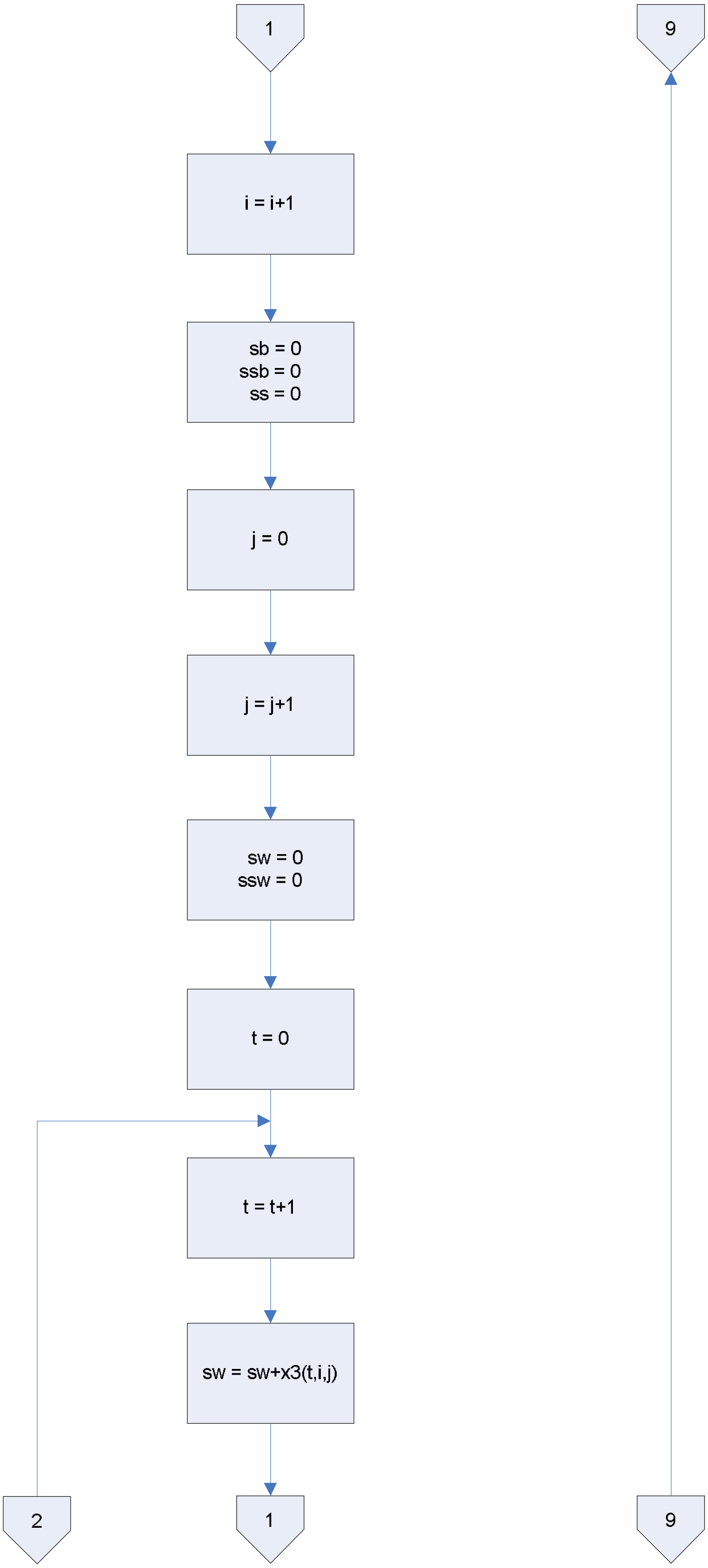

5
4
3
1
2
9
rmeanp(i)<103
yes
rmean=ram(t,i,j)
no
rmeanp(i)>103
yes
rmean=103
no
end if
call sub routine poiss
x3(t,i,j)=xp
t>1
yes
al=int(also(x3(t-1,i,j))+1
be=1
X=0
no
call subroutine gamma
ram(t,i,j) = x
rmean = ram(t,i,j)
5
4
3
2
1
9

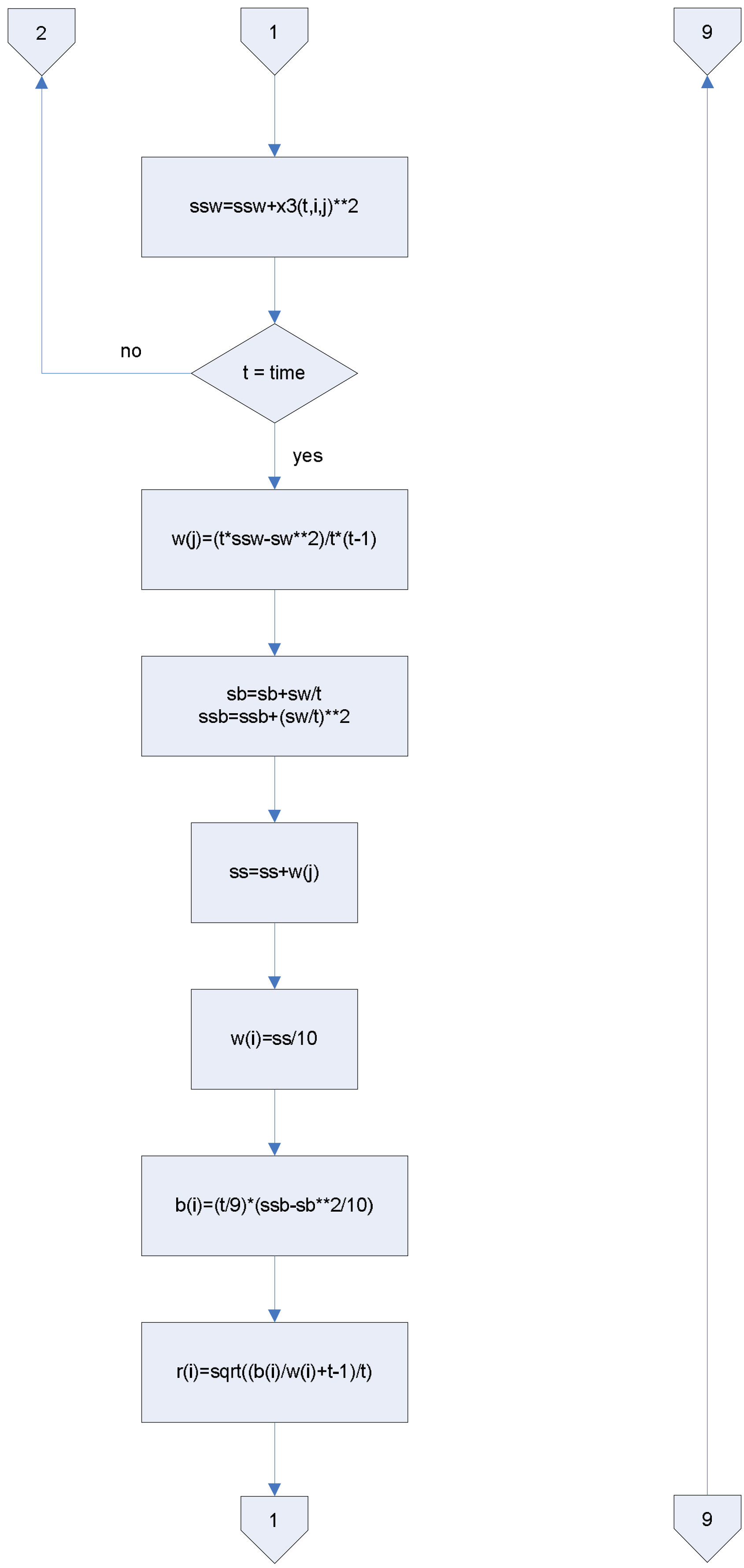

5
4
3
2
1
9
x<103
yes
rmean = ram(t,i,j)
no
x>103
yes
mean =103
no
end if
call subroutine poiss
x3(t,i,j) = xp
end if
run = 1
mf = int(x3(t,i,j))
m = 0
5
4
3
1
9

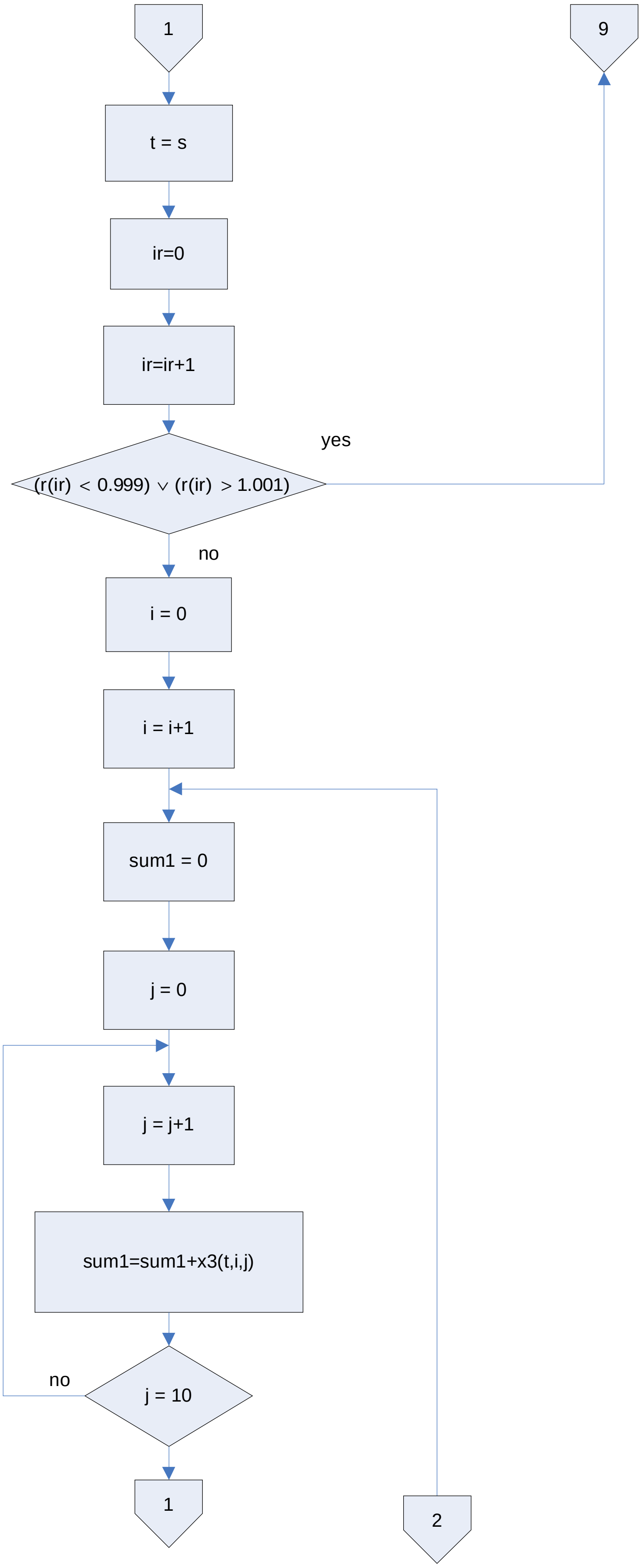

5
4
3
1
9
m = m+1
run = run*rmeanp(i)/m
no
m = mf
yes
u(t,i,j) = run/2.718**rmeanp(i)
t>0
yes
rlo(t,i,j) = u(t-1,i,j)
umu(t,i,j) = uml(t-1,i,j)
no
n = 0
k = 0
umu(t,i,j) = 1
rlo(t,i,j) = 1
n = 0
ifact = 1
run = 1
mf = int(x3(t,i,j)
m = 0
m = m+1
5
4
3
1
2
9

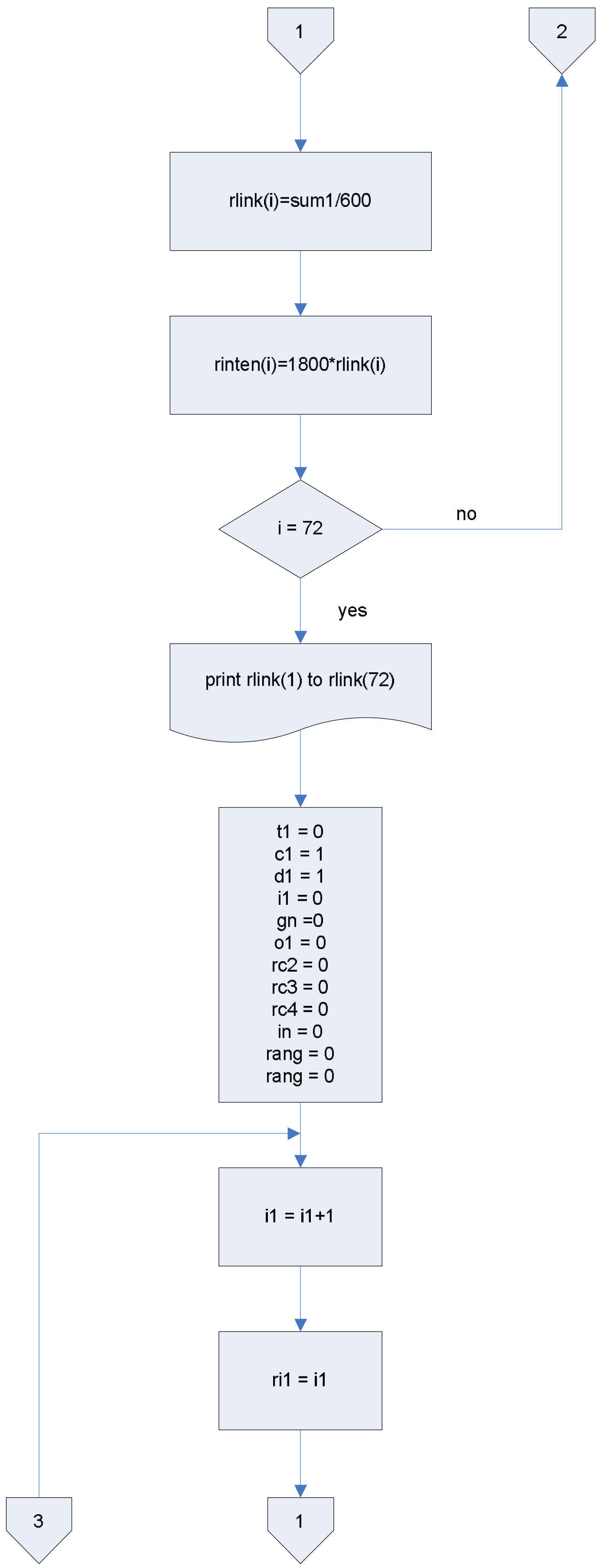

5
4
3
1
2
9
run=run*ram(t,i,j)/m
m = mf
no
yes
uml(t,i,j)=run/2.718**ram(t,i,j)
p=(u(t,i,j)*umu(t,i,j))/(rlo(t,i,j)*uml(t,i,j))
p ≥1
yes
un = 1
no
p<1
yes
call subroutine ber
no
end if
un = x
un = 1
yes
x3(t,i,j)=x3(t,i,j)
no
un = 0
yes
x3(t,i,j)=x3(t-1,i,j)
no
5
4
3
1
6
7
9

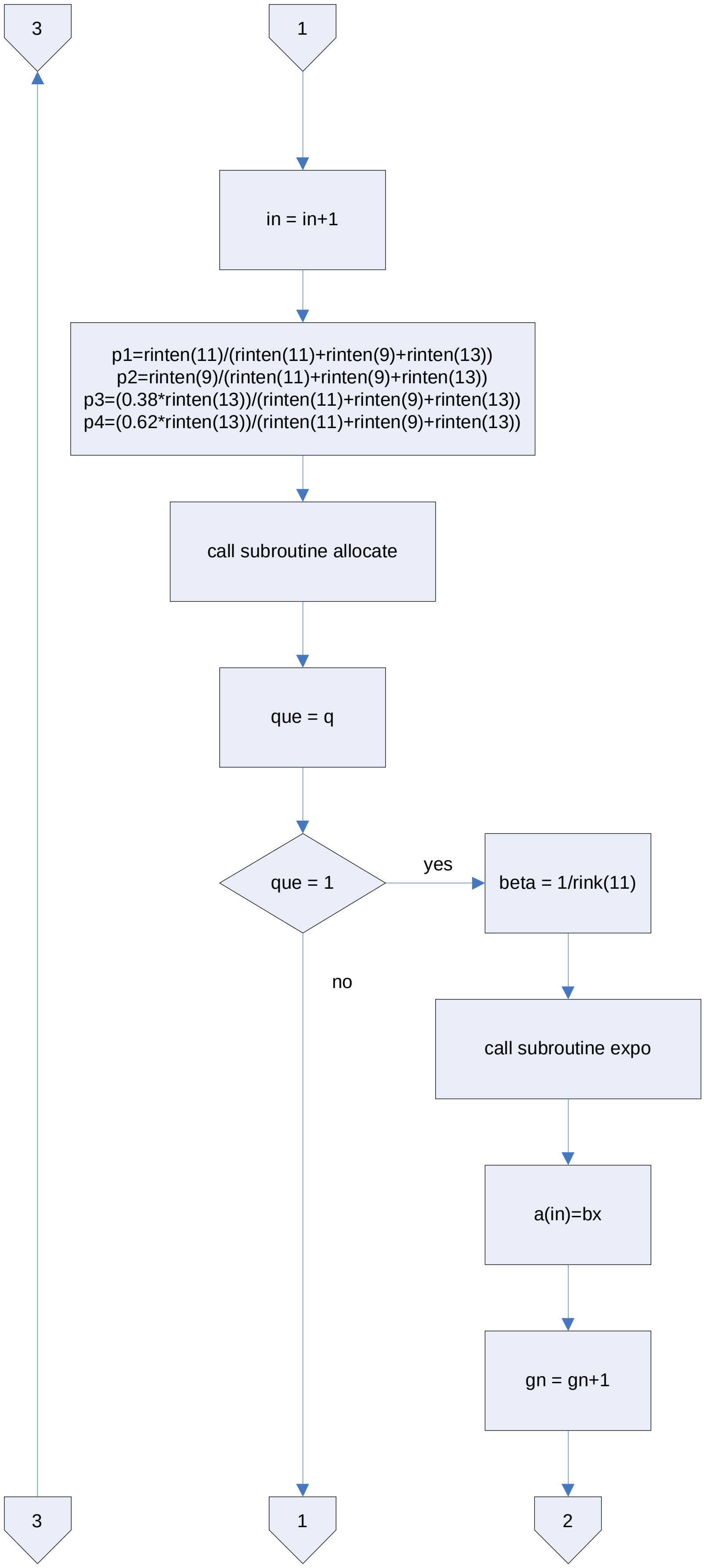

5
4
3
1
6
7
9
end if
no
i = 72
yes
x3(t,1,j) = x3(t,1,j)+x3(t,6,j)+...+x3(t,26,j)
x3(t,9,j) = x3(t,9,j)+x3(t,10,j)+...+x3(t,15,j)
x3(t,60,j) = x3(t,14,j)+x3(t,28,j)+…+x3(t,64,j)
no
j = 10
yes
yes
t = 1
no
t = s
time = t
i = 0
1
9

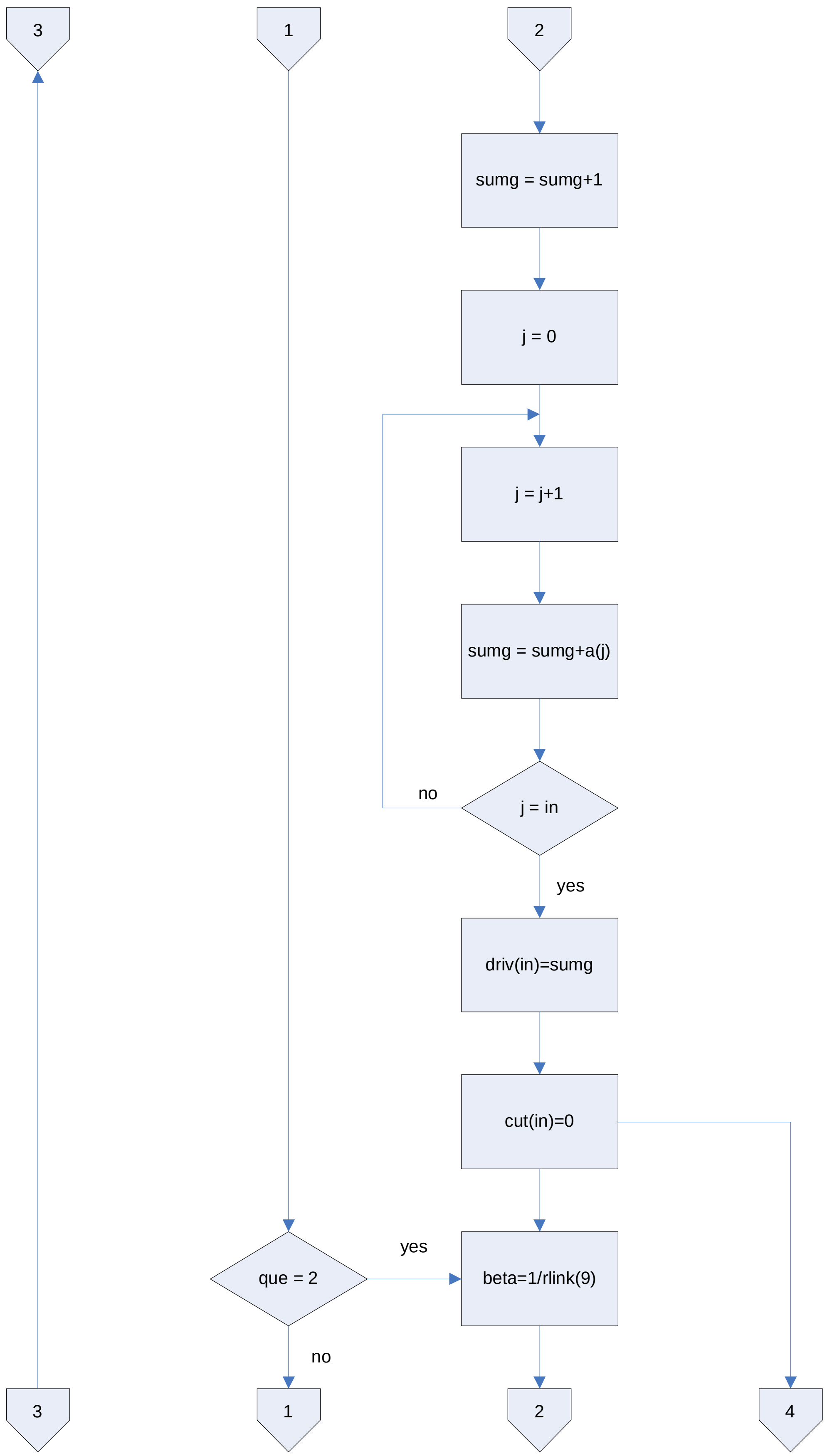

1
9
i = i+1
sb = 0
ssb = 0
ss = 0
j = 0
j = j+1
sw = 0
ssw = 0
t = 0
t = t+1
sw = sw+x3(t,i,j)
2
1
9

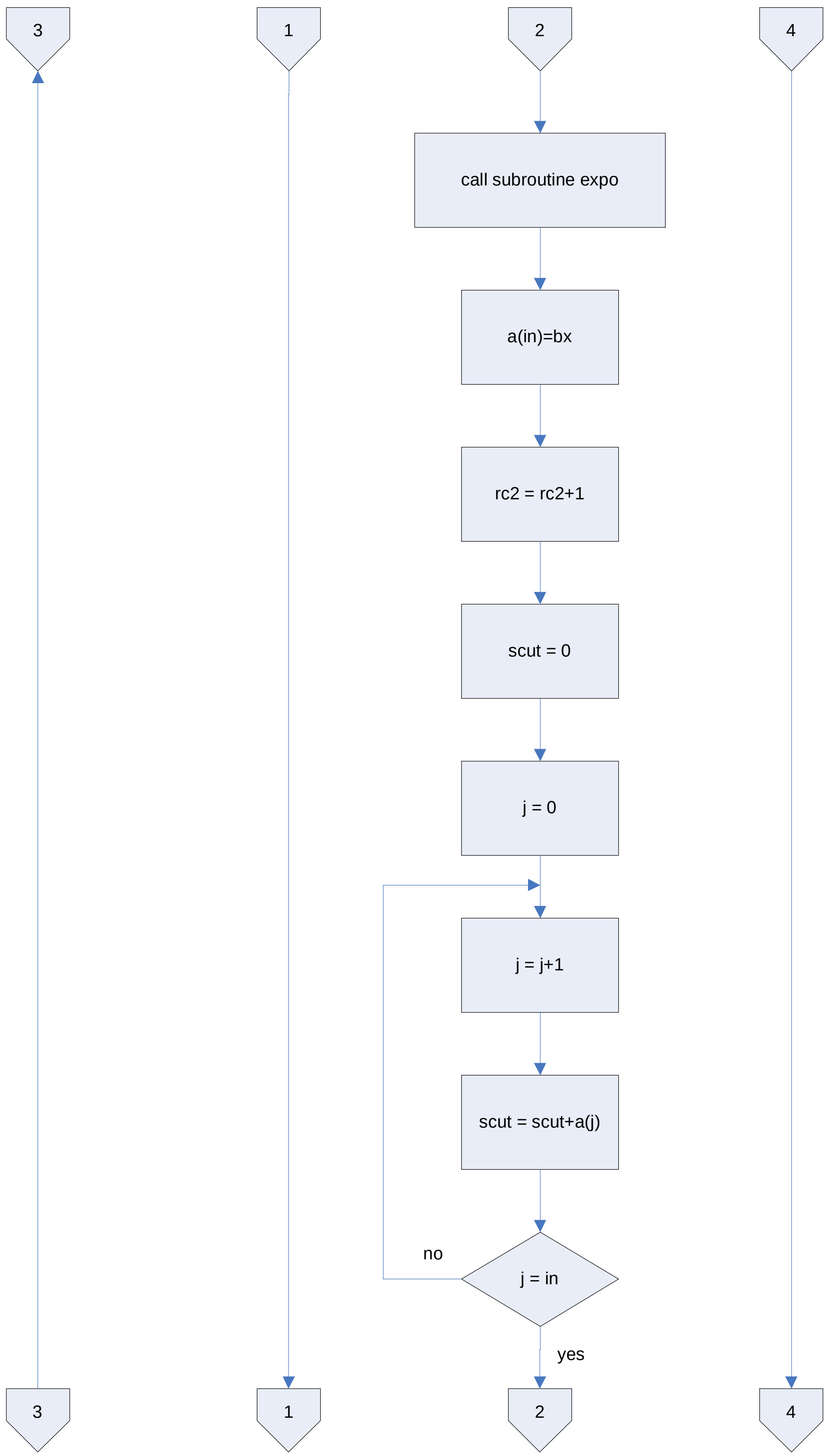

2
1
9
ssw=ssw+x3(t,i,j)**2
no
t = time
yes
w(j)=(t*ssw-sw**2)/t*(t-1)
sb=sb+sw/t
ssb=ssb+(sw/t)**2
ss=ss+w(j)
w(i)=ss/10
b(i)=(t/9)*(ssb-sb**2/10)
r(i)=sqrt((b(i)/w(i)+t-1)/t)
1
9

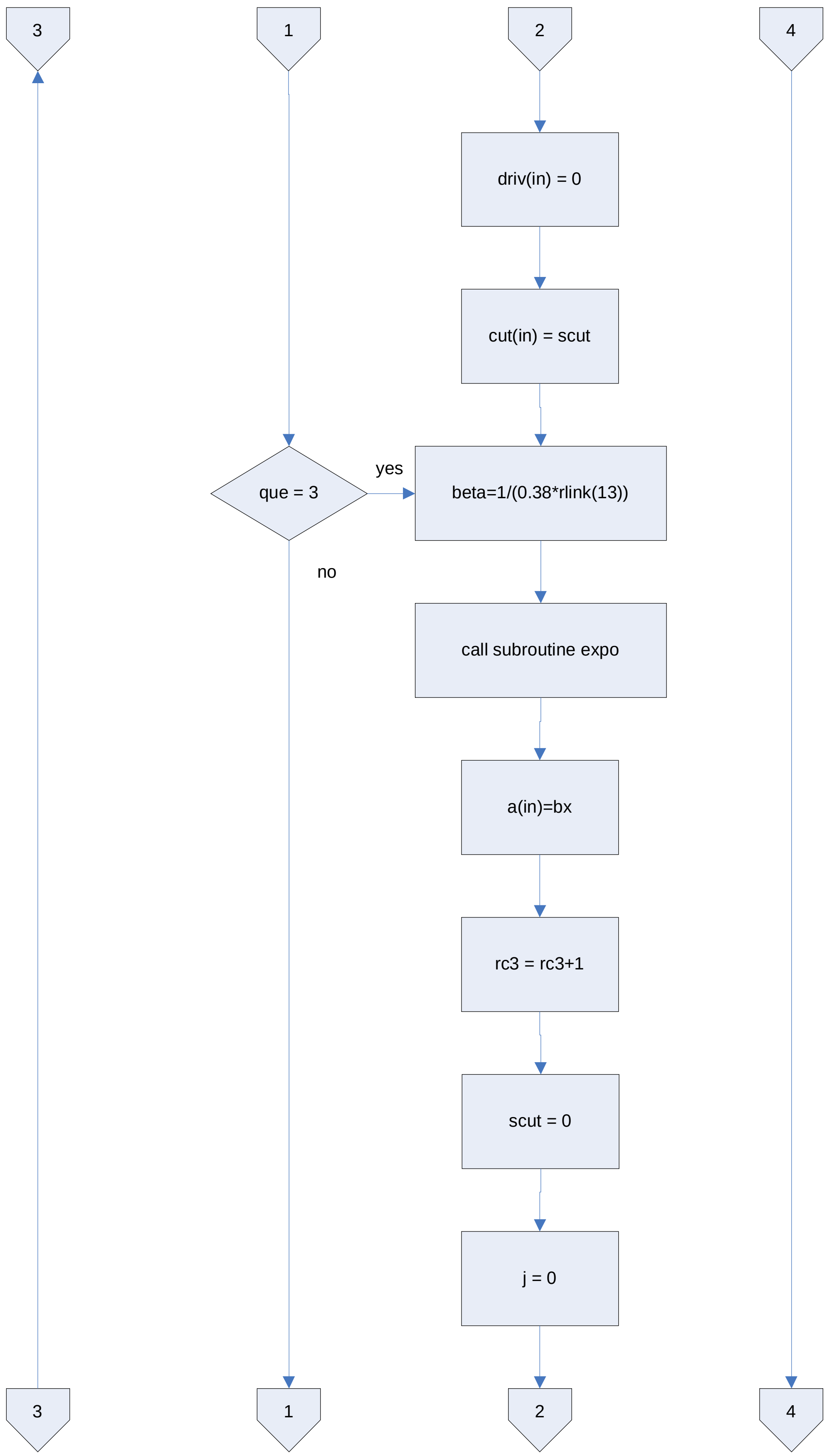

1
9
t = s
ir=0
ir=ir+1
yes
(r(ir) < 0.999) ∨ (r(ir) > 1.001)
no
i = 0
i = i+1
sum1 = 0
j = 0
j = j+1
sum1=sum1+x3(t,i,j)
no
j = 10
1
2

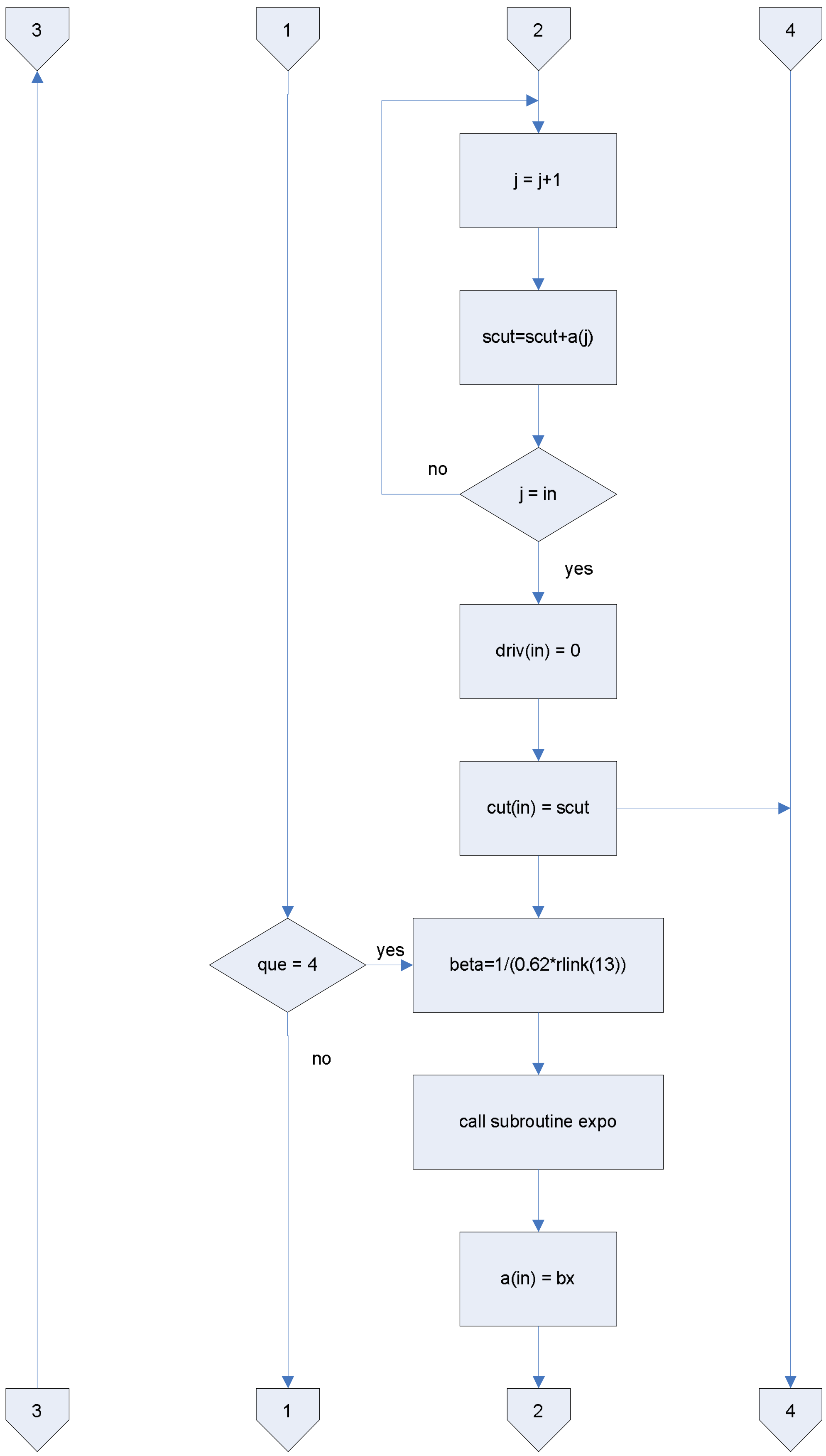

1
2
rlink(i)=sum1/600
rinten(i)=1800*rlink(i)
i = 72
no
yes
print rlink(1) to rlink(72)
t1 = 0
c1 = 1
d1 = 1
i1 = 0
gn =0
o1 = 0
rc2 = 0
rc3 = 0
rc4 = 0
in = 0
rang = 0
rang = 0
i1 = i1+1
ri1 = i1
3
1

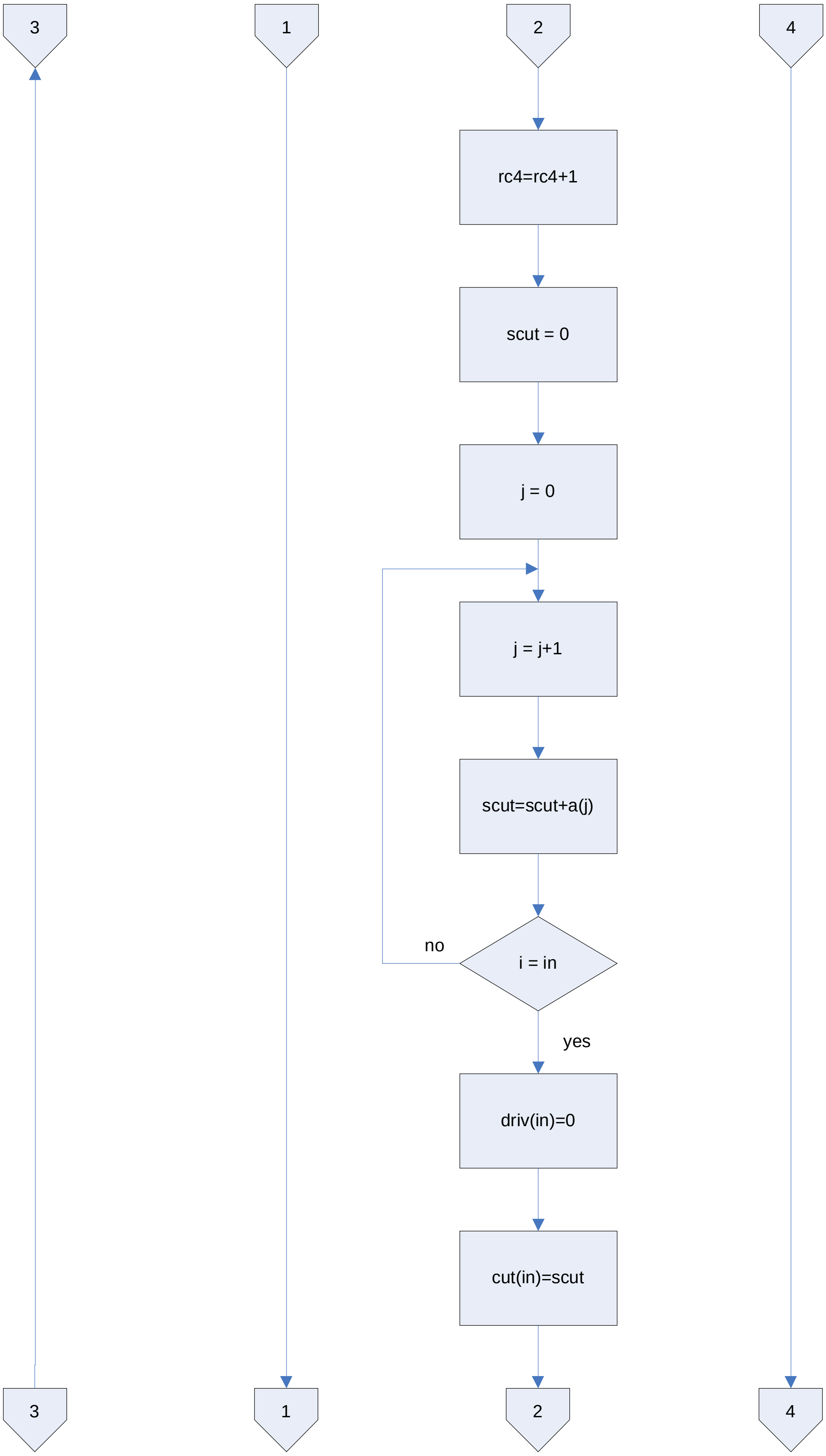

3
1
in = in+1
p1=rinten(11)/(rinten(11)+rinten(9)+rinten(13))
p2=rinten(9)/(rinten(11)+rinten(9)+rinten(13))
p3=(0.38*rinten(13))/(rinten(11)+rinten(9)+rinten(13))
p4=(0.62*rinten(13))/(rinten(11)+rinten(9)+rinten(13))
call subroutine allocate
que = q
que = 1
yes
beta = 1/rink(11)
no
call subroutine expo
a(in)=bx
gn = gn+1
3
1
2

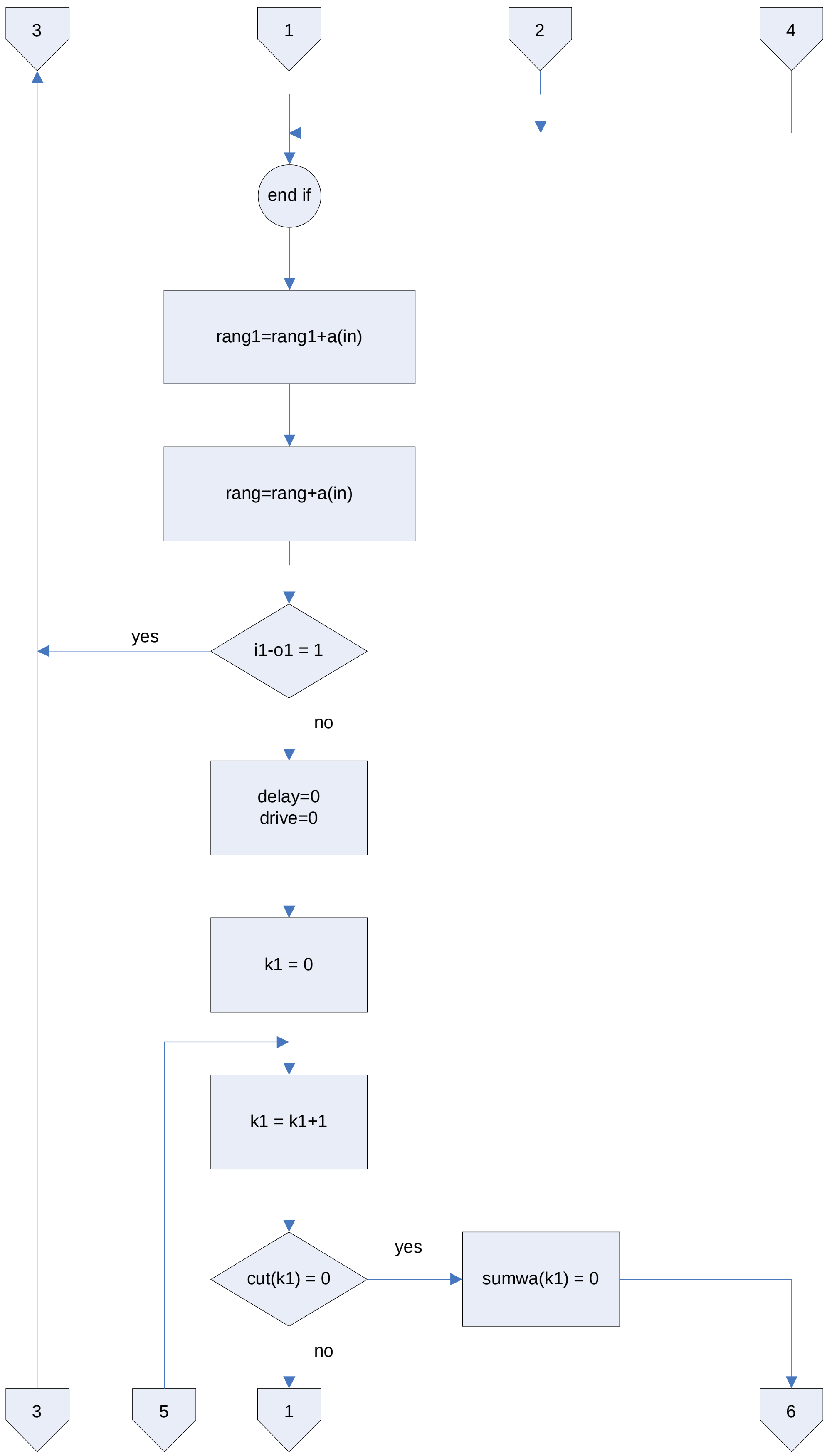

3
1
2
sumg = sumg+1
j = 0
j = j+1
sumg = sumg+a(j)
no
j = in
yes
driv(in)=sumg
cut(in)=0
yes
que = 2
beta=1/rlink(9)
no
3
1
2
4

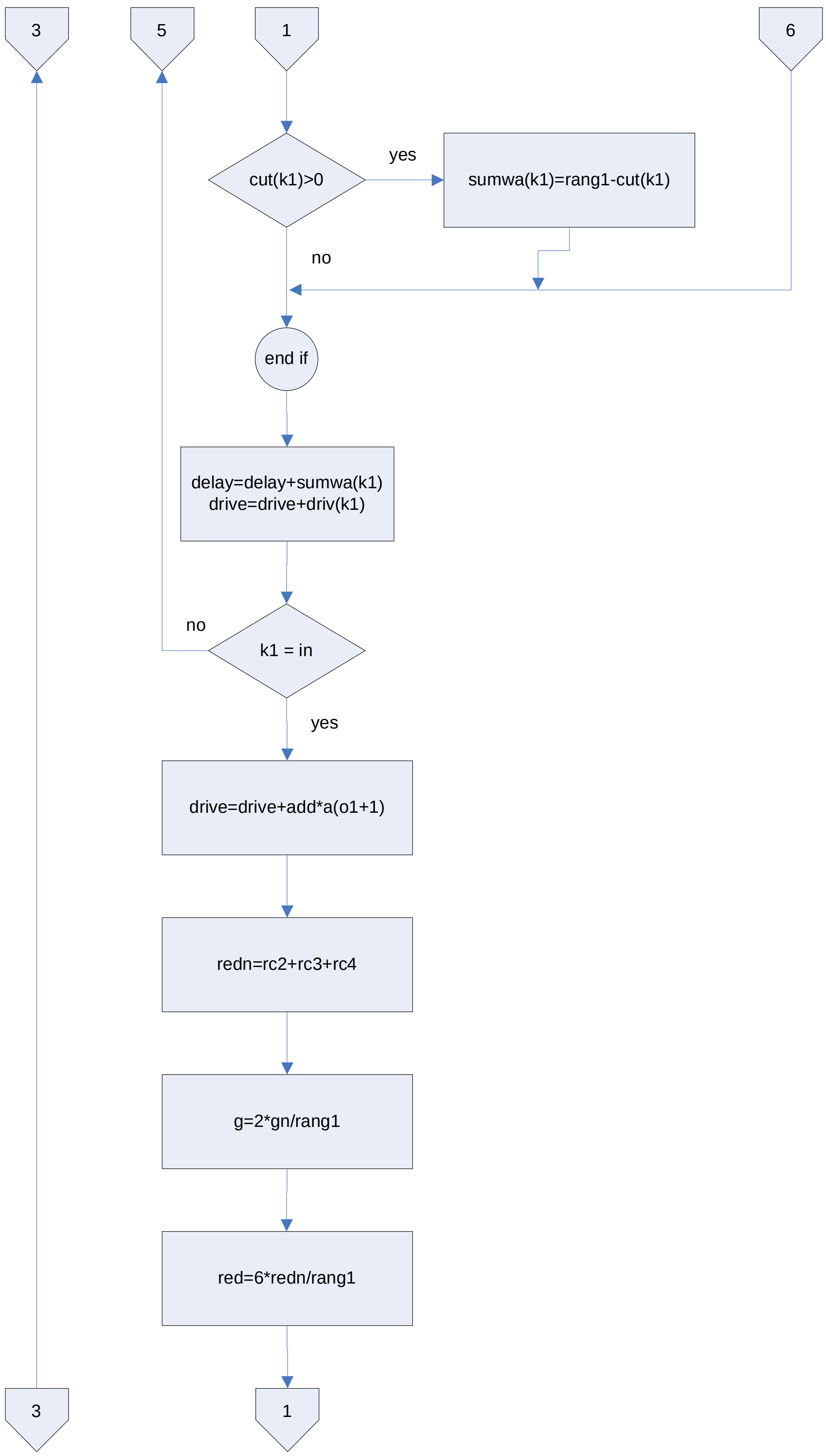

3
1
2
4
call subroutine expo
a(in)=bx
rc2 = rc2+1
scut = 0
j = 0
j = j+1
scut = scut+a(j)
no
j = in
yes
3
1
2
4

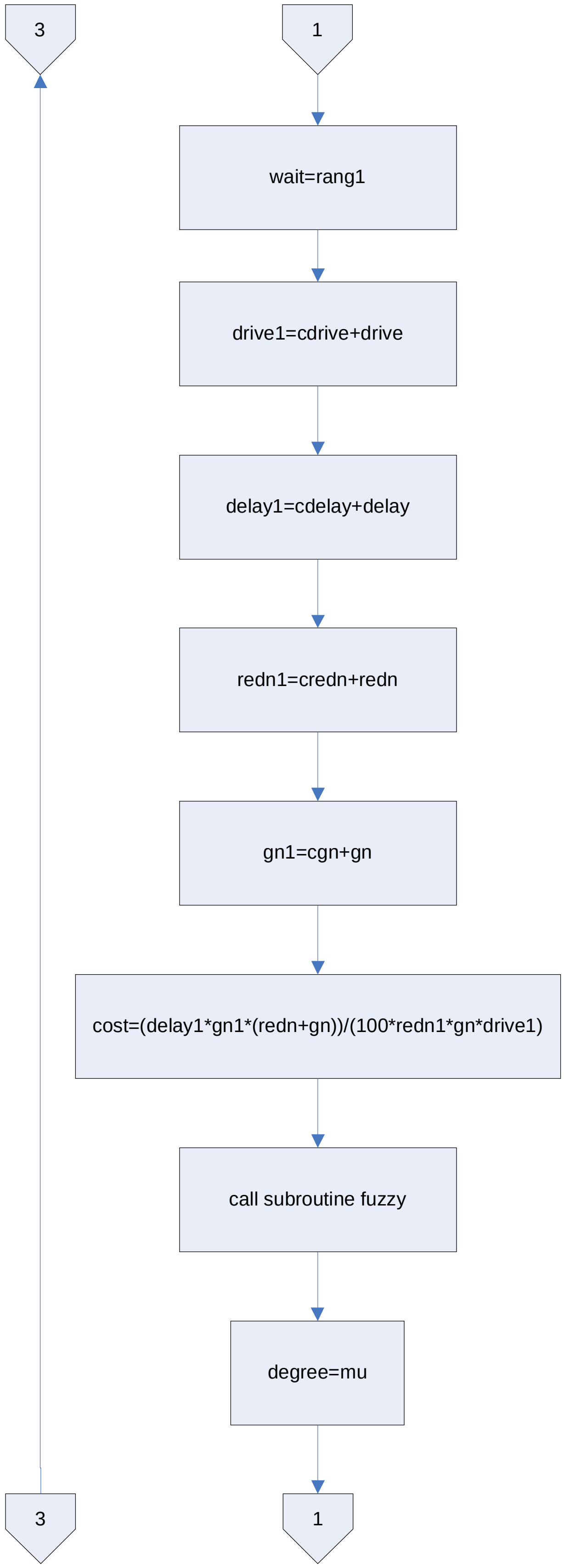

3
1
2
4
driv(in) = 0
cut(in) = scut
que = 3
yes
beta=1/(0.38*rlink(13))
no
call subroutine expo
a(in)=bx
rc3 = rc3+1
scut = 0
j = 0
3
1
2
4

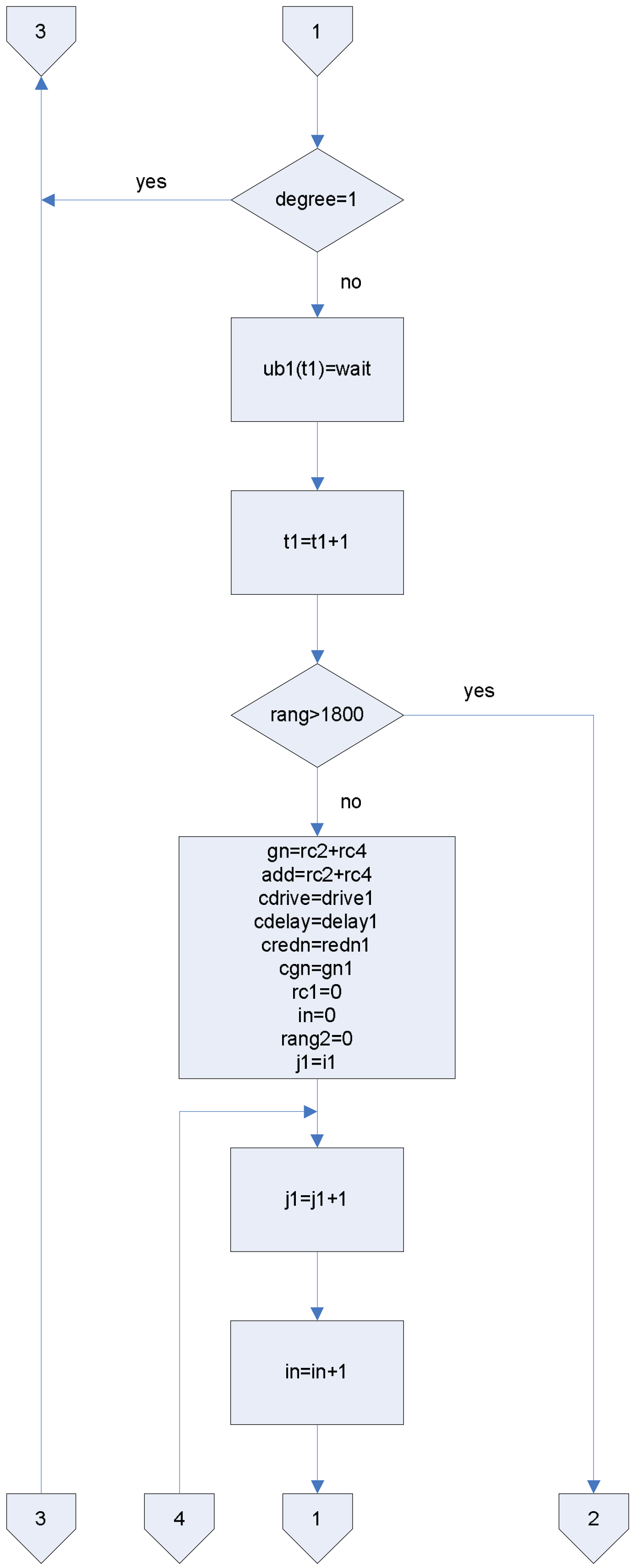

3
1
2
4
j = j+1
scut=scut+a(j)
no
j = in
yes
driv(in) = 0
cut(in) = scut
que = 4
yes
beta=1/(0.62*rlink(13))
no
call subroutine expo
a(in) = bx
3
1
2
4

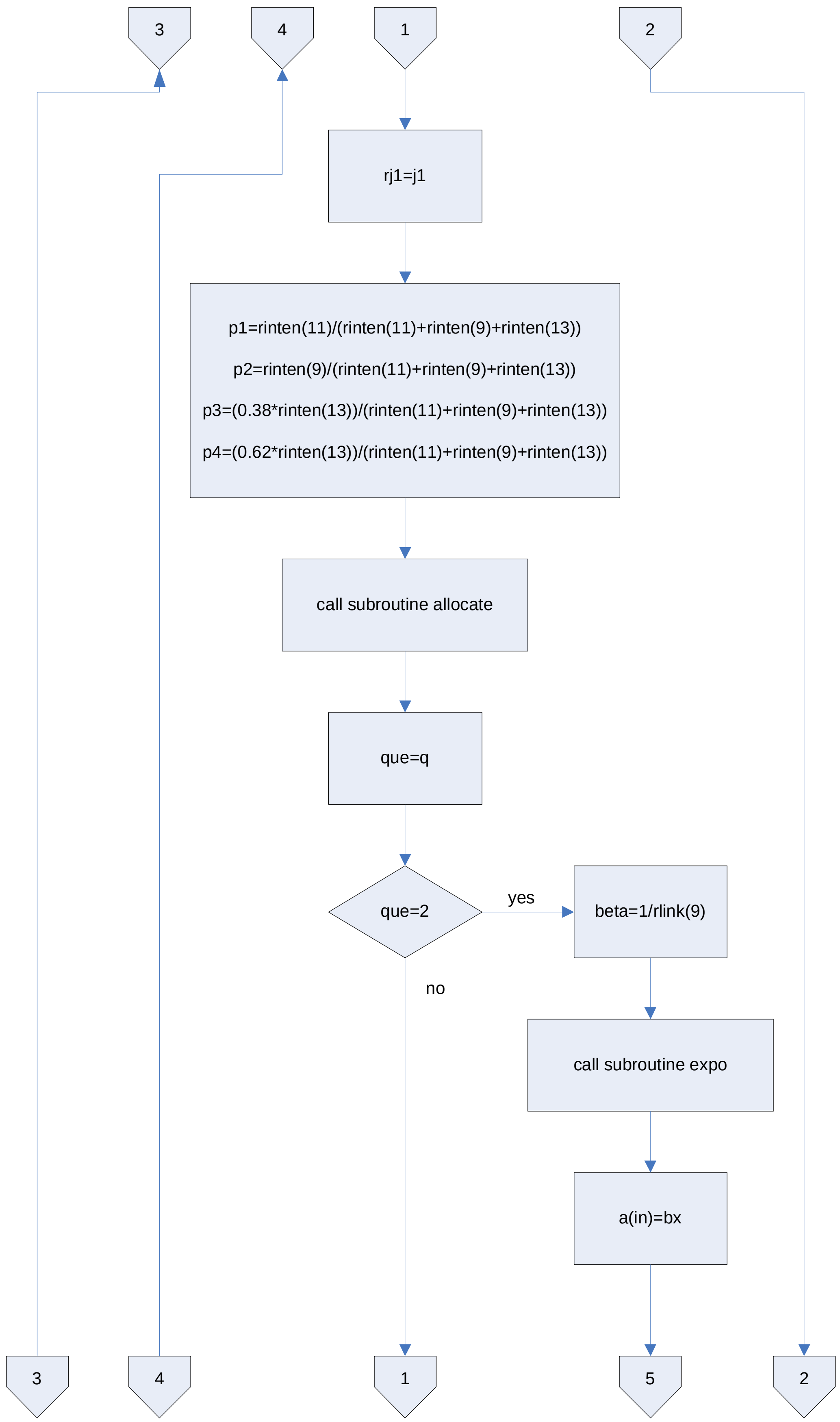

3
1
2
4
rc4=rc4+1
scut = 0
j = 0
j = j+1
scut=scut+a(j)
no
i = in
yes
driv(in)=0
cut(in)=scut
3
1
2
4

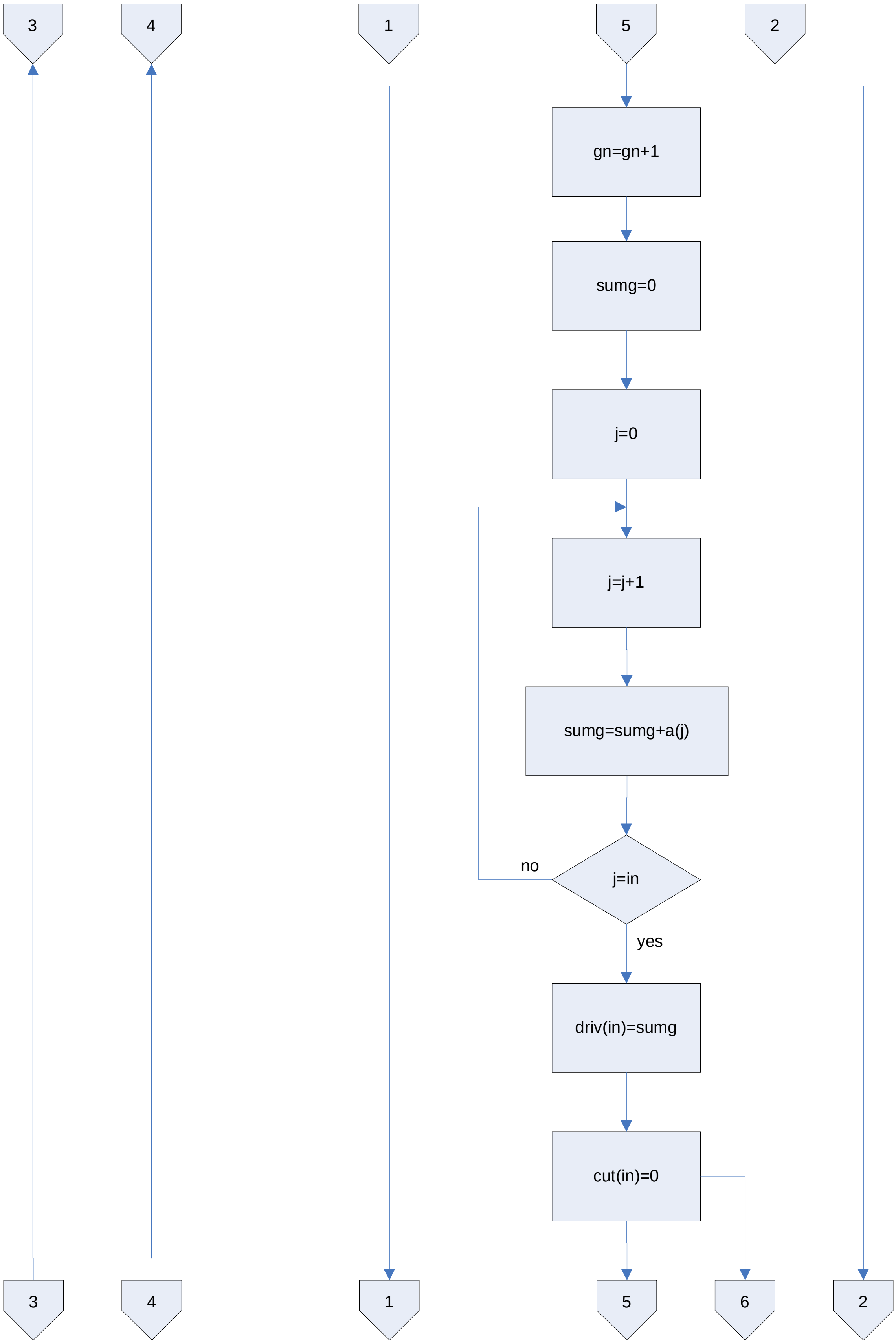

3
1
2
4
end if
rang1=rang1+a(in)
rang=rang+a(in)
yes
i1-o1 = 1
no
delay=0
drive=0
k1 = 0
k1 = k1+1
yes
cut(k1) = 0
sumwa(k1) = 0
no
3
5
1
6

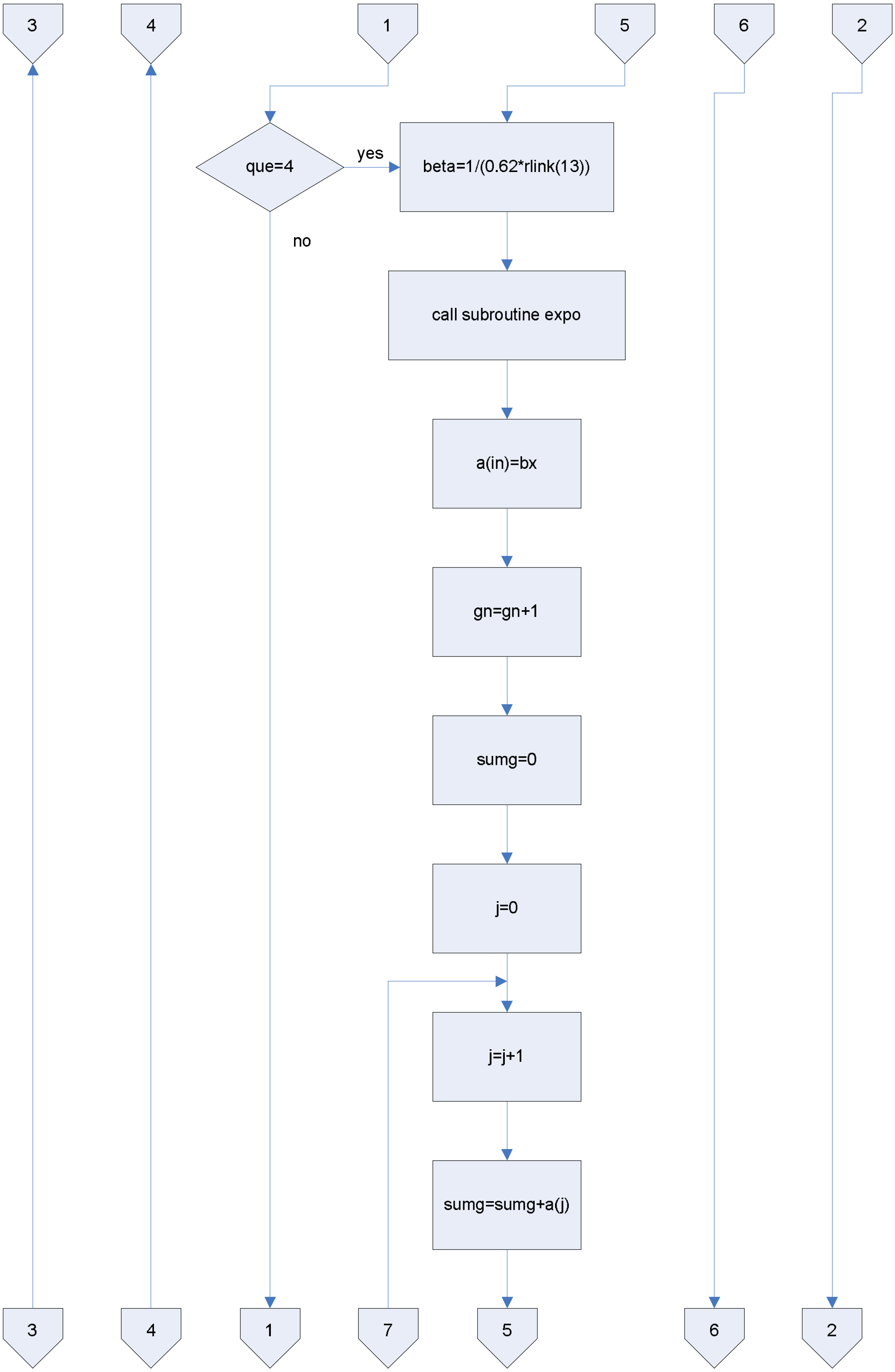

3
5
1
6
cut(k1)>0
yes
sumwa(k1)=rang1-cut(k1)
no
end if
delay=delay+sumwa(k1)
drive=drive+driv(k1)
no
k1 = in
yes
drive=drive+add*a(o1+1)
redn=rc2+rc3+rc4
g=2*gn/rang1
red=6*redn/rang1
3
1

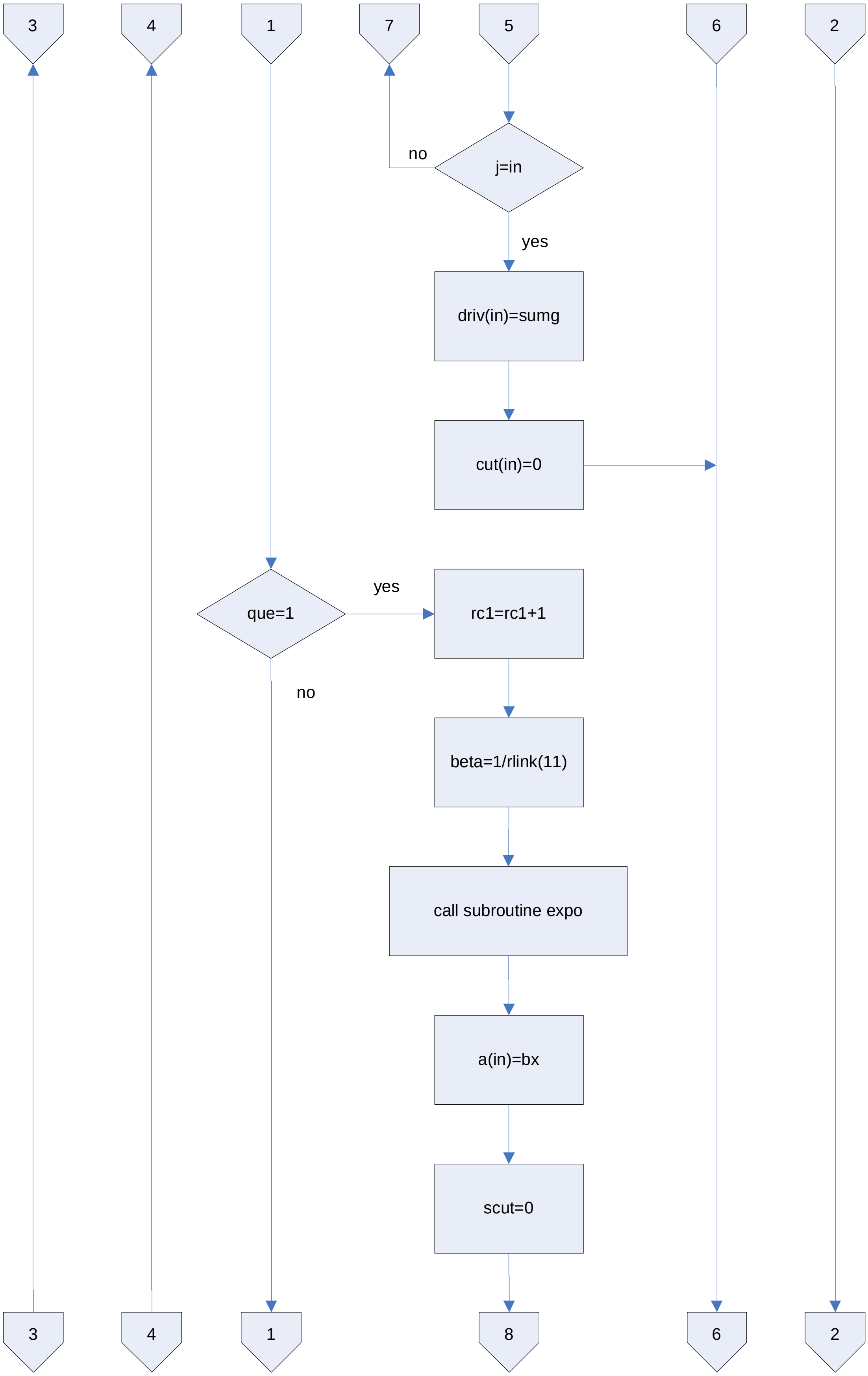

3
1
wait=rang1
drive1=cdrive+drive
delay1=cdelay+delay
redn1=credn+redn
gn1=cgn+gn
cost=(delay1*gn1*(redn+gn))/(100*redn1*gn*drive1)
call subroutine fuzzy
degree=mu
3
1

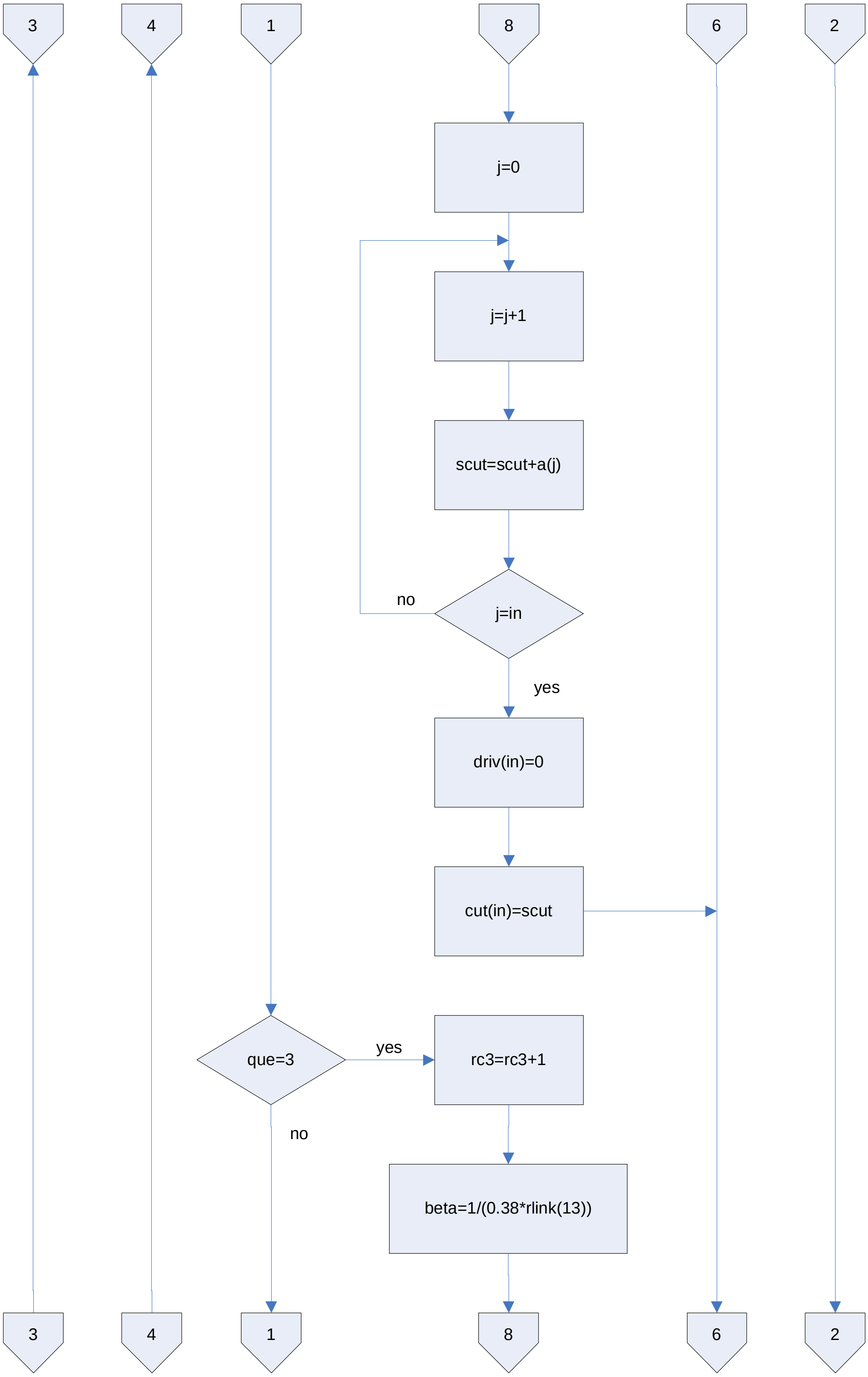

3
1
degree=1
yes
no
ub1(t1)=wait
t1=t1+1
rang>1800
yes
no
gn=rc2+rc4
add=rc2+rc4
cdrive=drive1
cdelay=delay1
credn=redn1
cgn=gn1
rc1=0
in=0
rang2=0
j1=i1
j1=j1+1
in=in+1
3
4
1
2

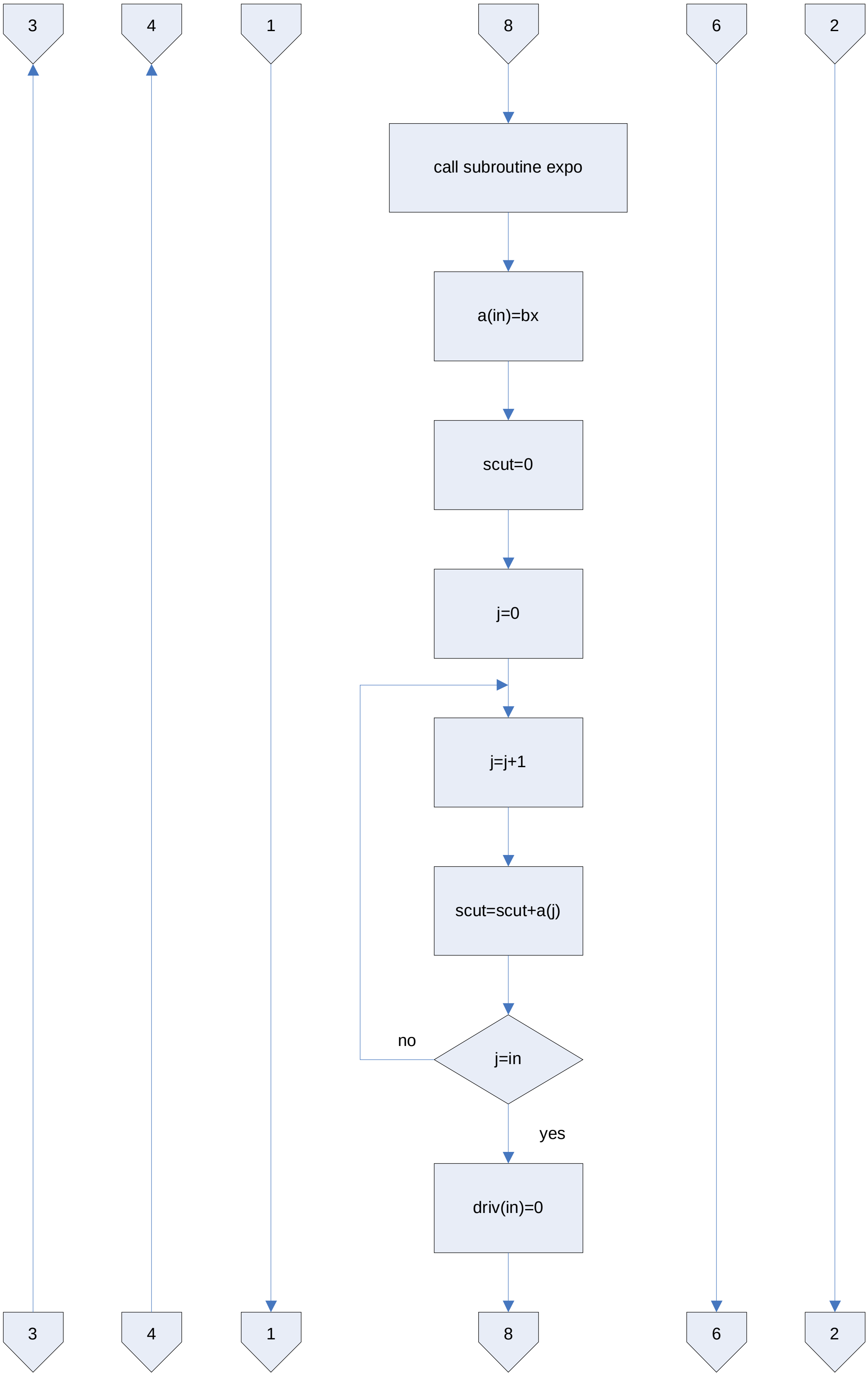

3
4
1
2
rj1=j1
p1=rinten(11)/(rinten(11)+rinten(9)+rinten(13))
p2=rinten(9)/(rinten(11)+rinten(9)+rinten(13))
p3=(0.38*rinten(13))/(rinten(11)+rinten(9)+rinten(13))
p4=(0.62*rinten(13))/(rinten(11)+rinten(9)+rinten(13))
call subroutine allocate
que=q
que=2
yes
beta=1/rlink(9)
no
call subroutine expo
a(in)=bx
3
4
1
5
2

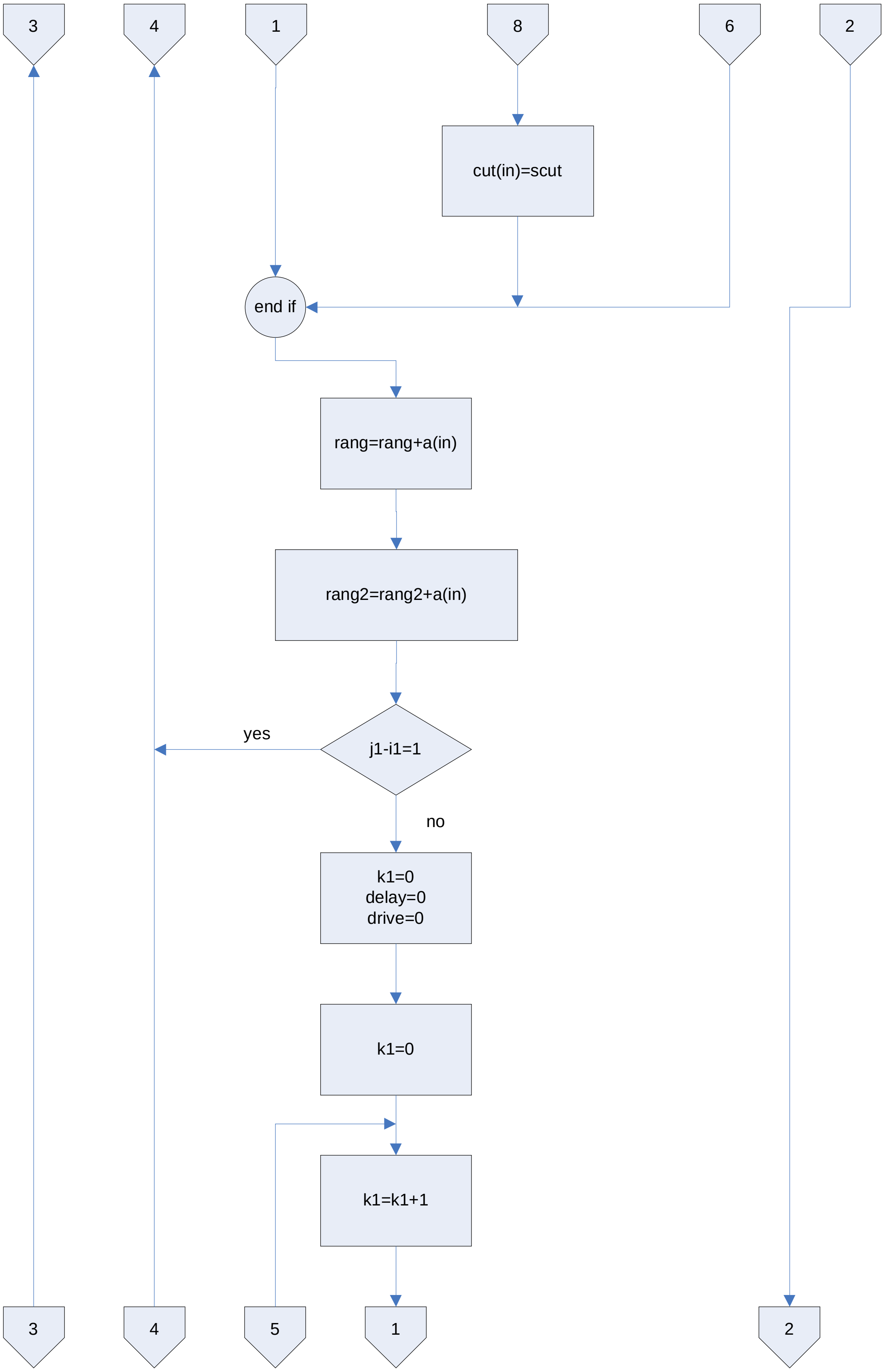

3
4
1
5
2
gn=gn+1
sumg=0
j=0
j=j+1
sumg=sumg+a(j)
no
j=in
yes
driv(in)=sumg
cut(in)=0
3
4
1
5
6
2

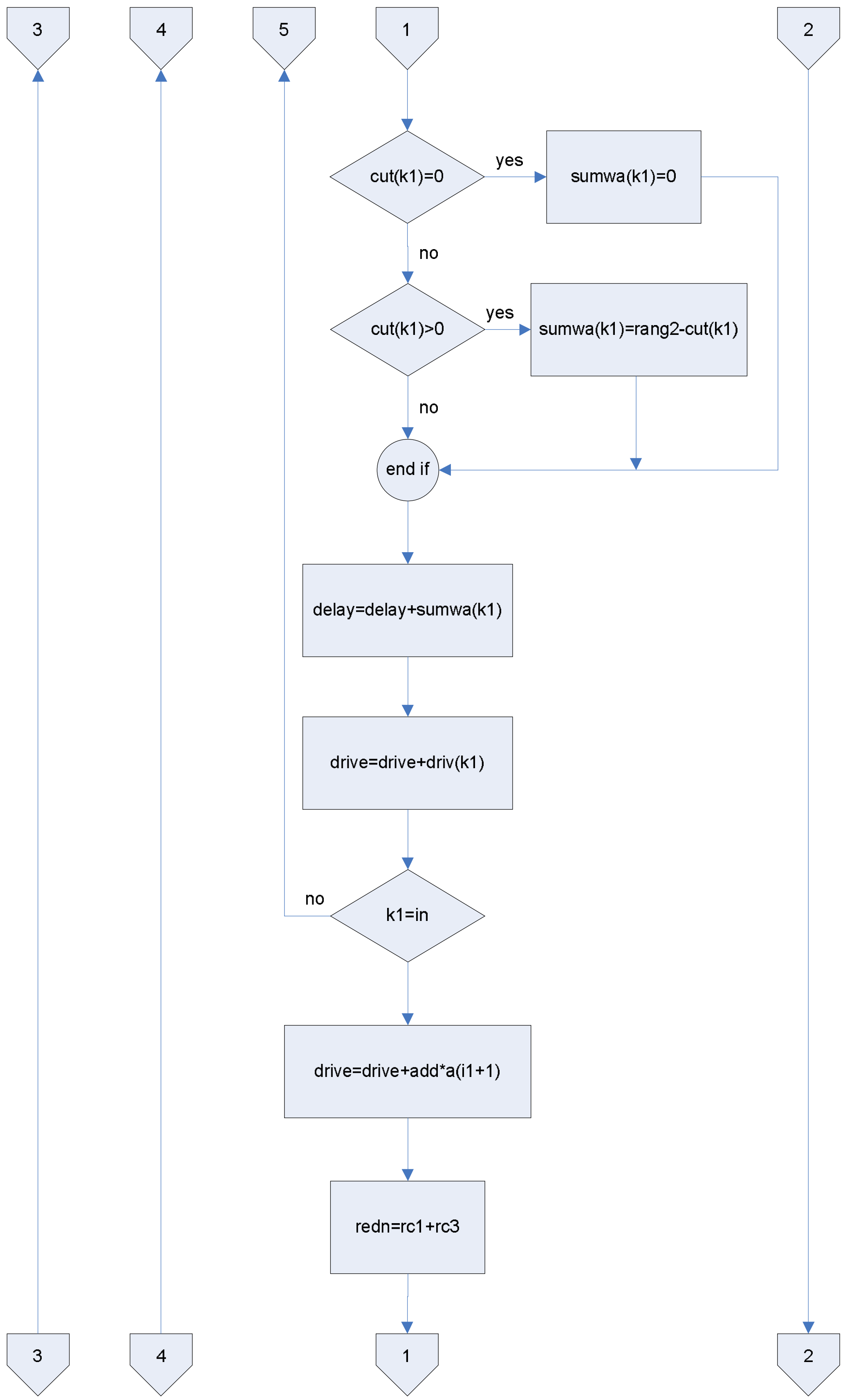

3
4
1
5
6
2
que=4
yes
no
beta=1/(0.62*rlink(13))
call subroutine expo
a(in)=bx
gn=gn+1
sumg=0
j=0
j=j+1
sumg=sumg+a(j)
3
4
1
7
5
6
2

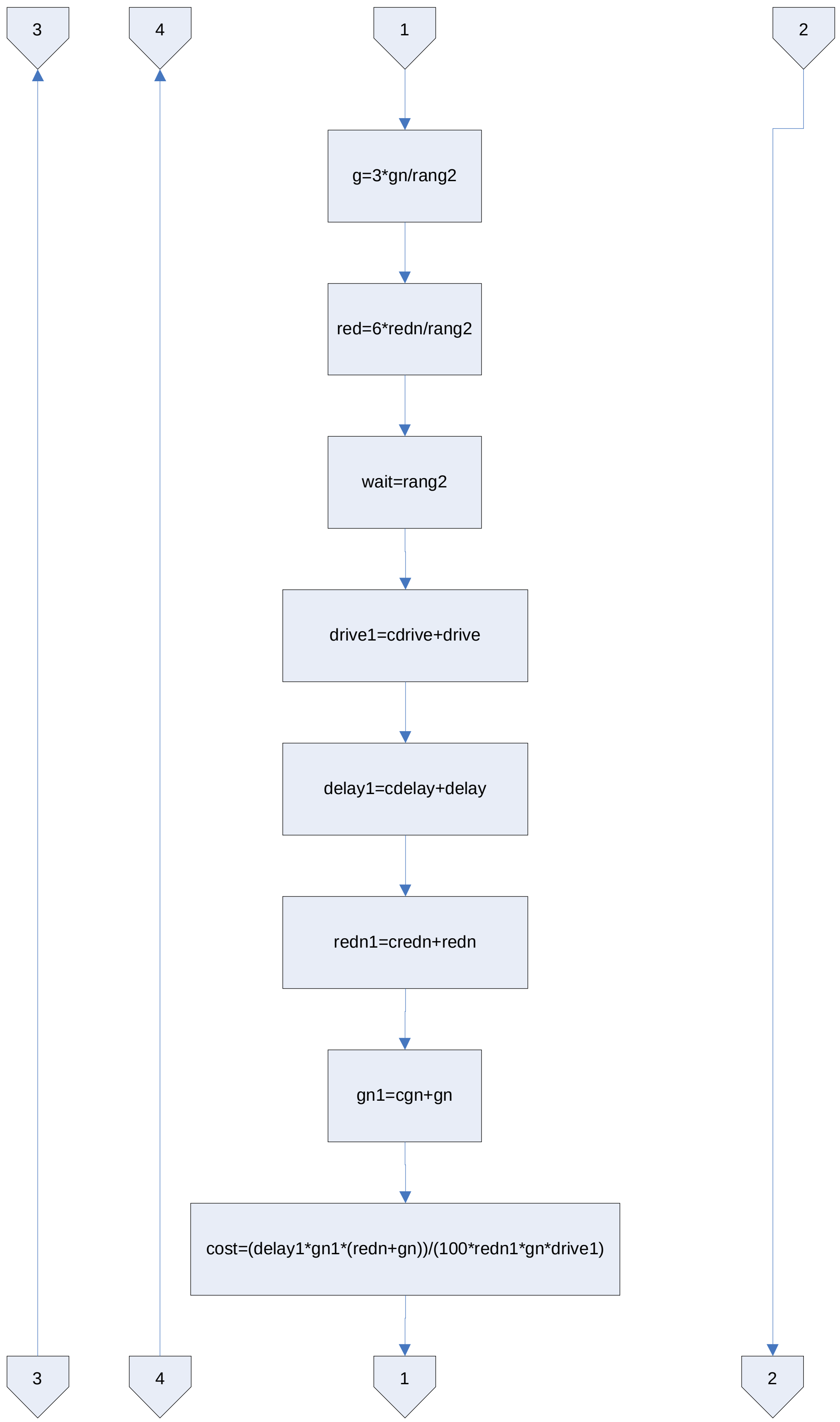

3
4
1
7
5
6
2
j=in
no
yes
driv(in)=sumg
cut(in)=0
que=1
yes
no
rc1=rc1+1
beta=1/rlink(11)
call subroutine expo
a(in)=bx
scut=0
3
4
1
8
6
2

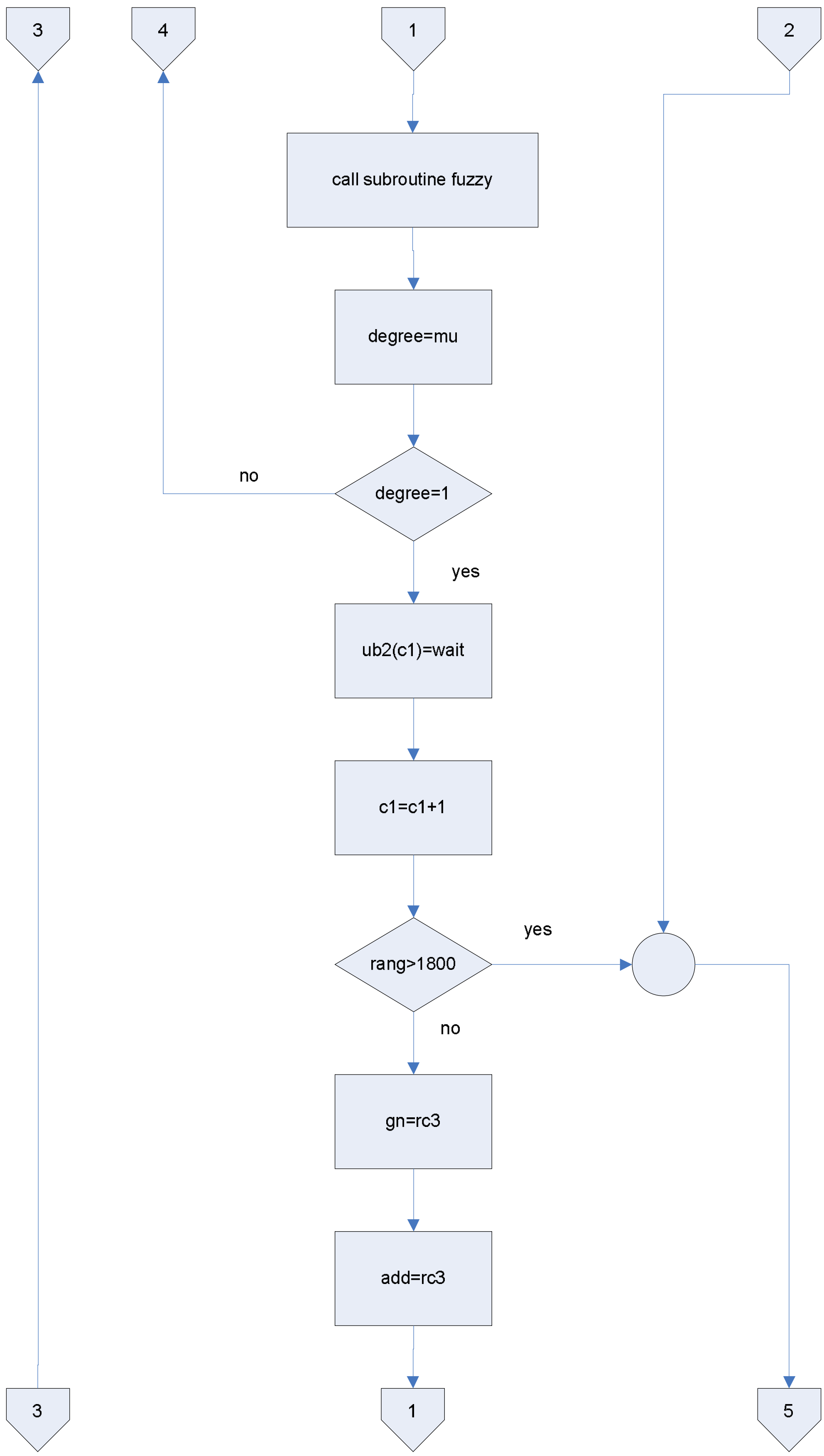

3
4
1
8
6
2
j=0
j=j+1
scut=scut+a(j)
j=in
no
yes
driv(in)=0
cut(in)=scut
que=3
yes
rc3=rc3+1
no
beta=1/(0.38*rlink(13))
3
4
1
8
6
2

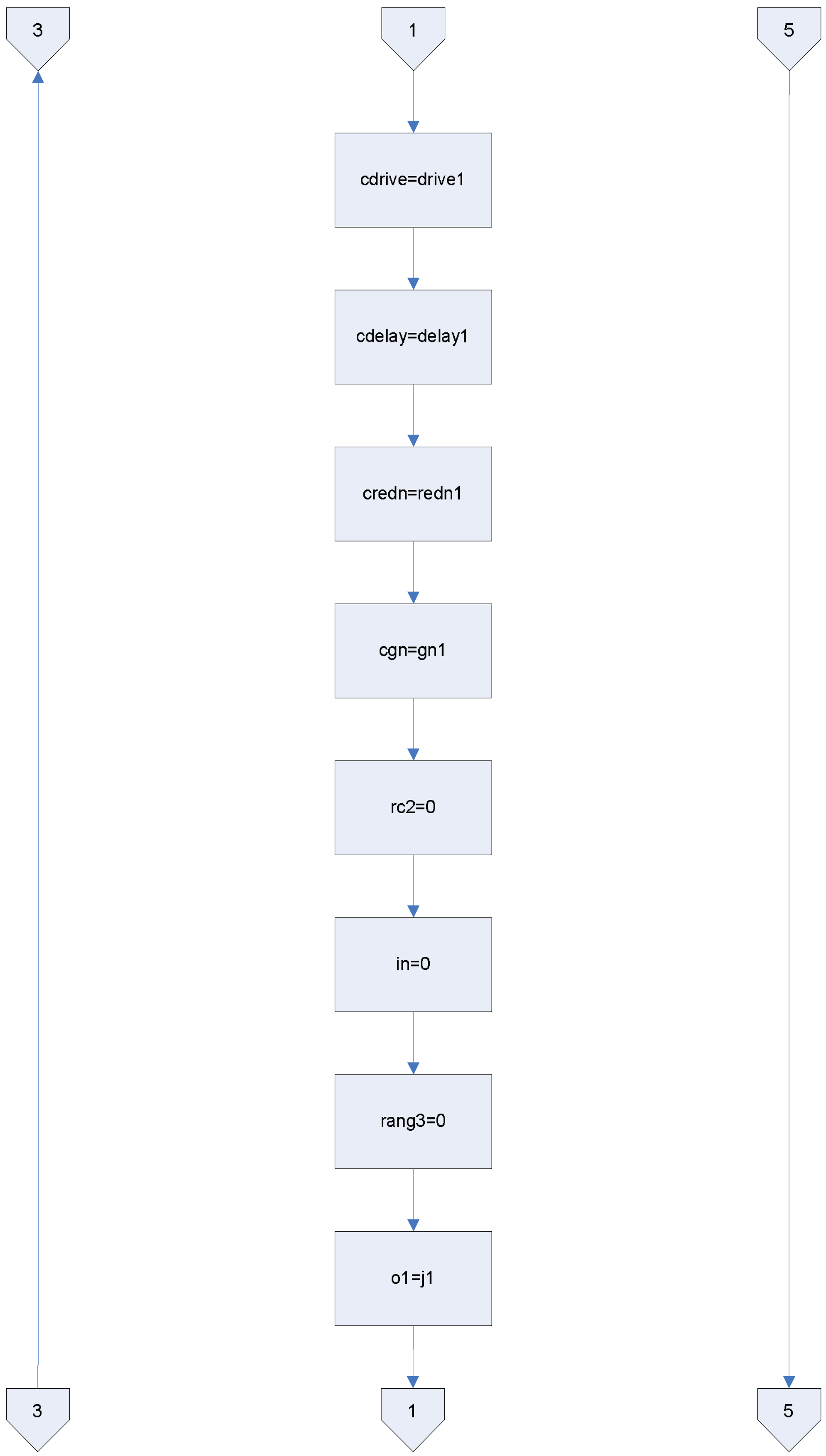

3
4
1
8
6
2
call subroutine expo
a(in)=bx
scut=0
j=0
j=j+1
scut=scut+a(j)
no
j=in
yes
driv(in)=0
3
4
1
8
6
2

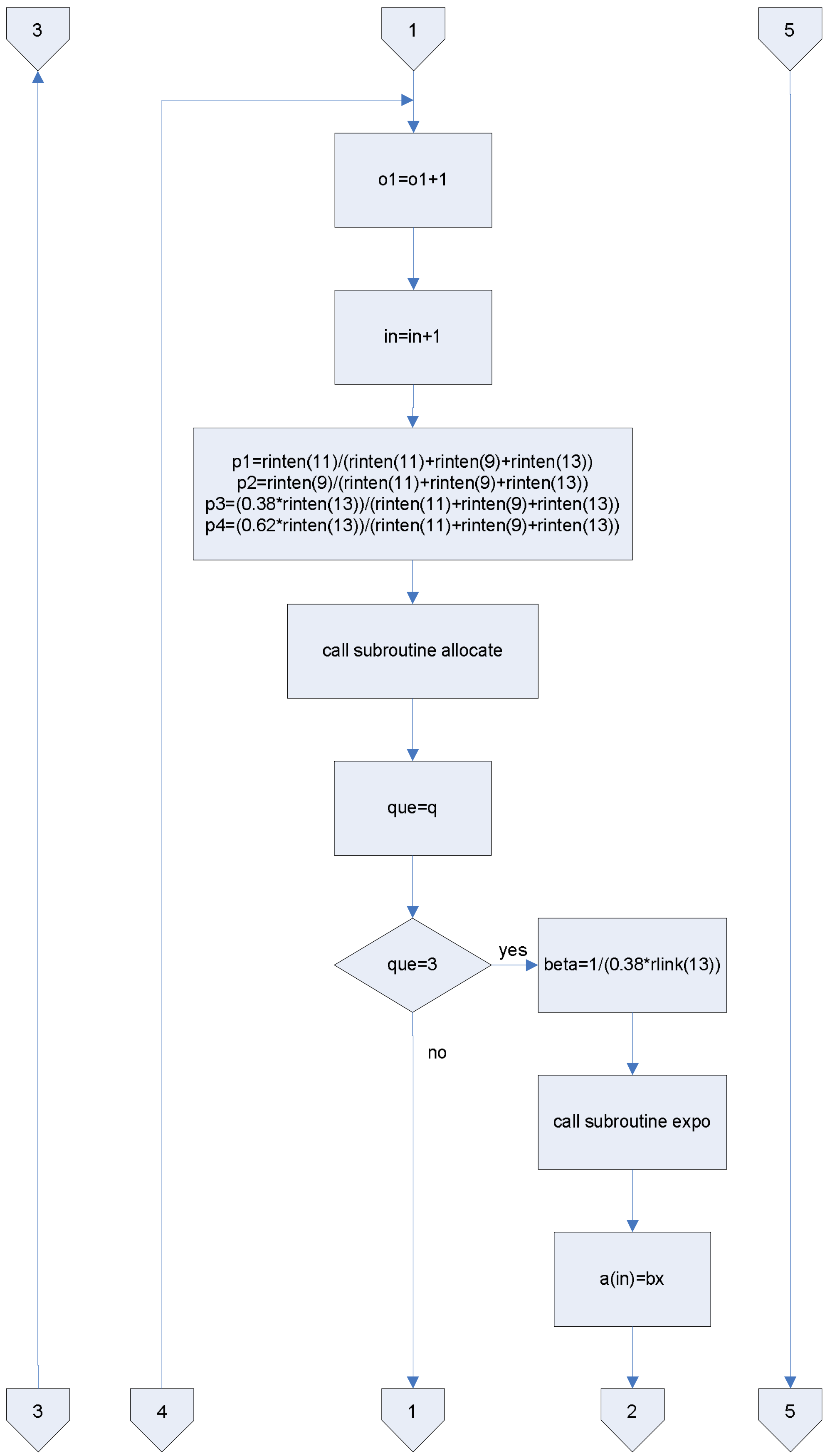

3
4
1
8
6
2
cut(in)=scut
end if
rang=rang+a(in)
rang2=rang2+a(in)
yes
j1-i1=1
no
k1=0
delay=0
drive=0
k1=0
k1=k1+1
3
4
5
1
2

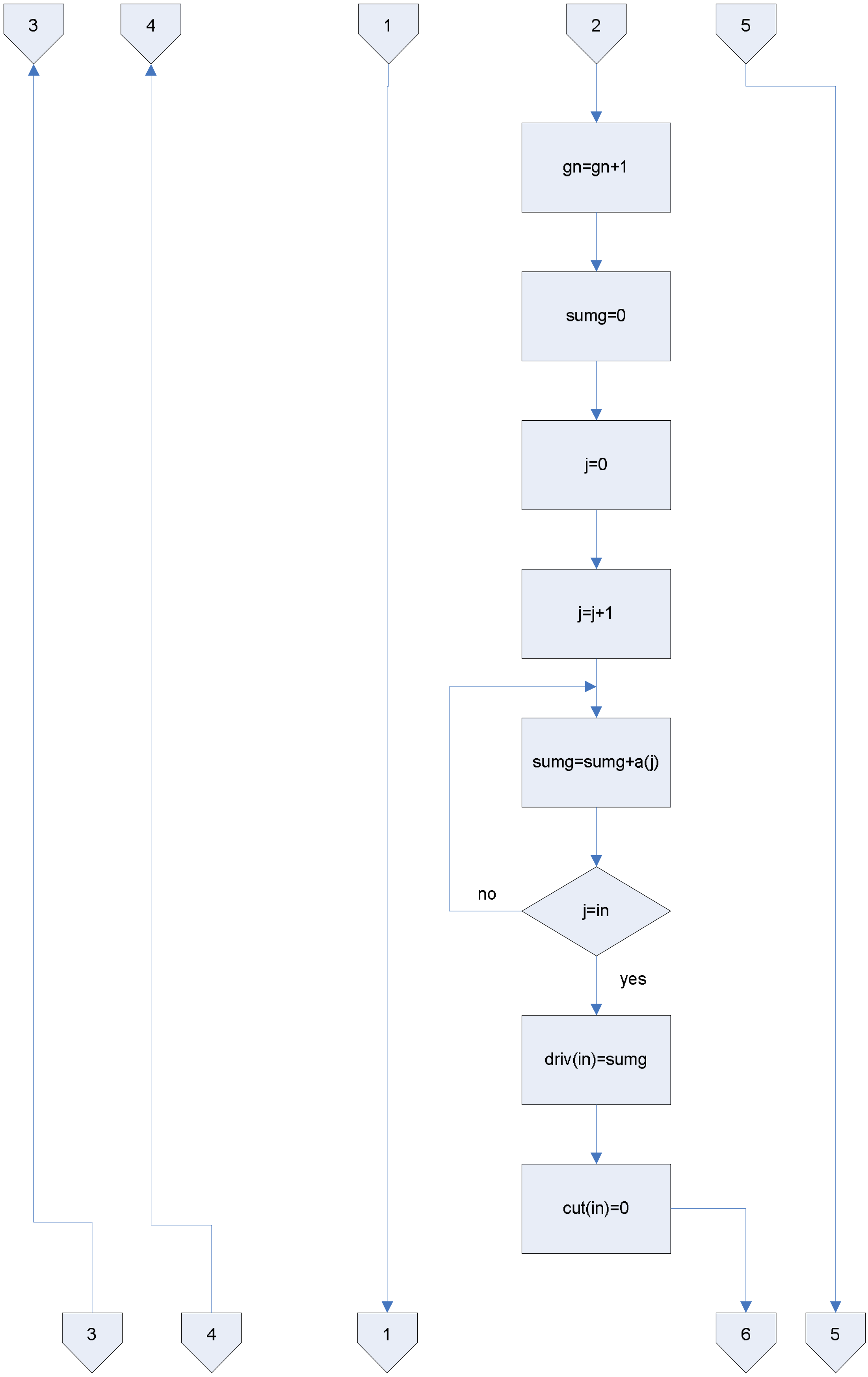

3
4
5
1
2
cut(k1)=0
yes
sumwa(k1)=0
no
cut(k1)>0
yes
sumwa(k1)=rang2-cut(k1)
no
end if
delay=delay+sumwa(k1)
drive=drive+driv(k1)
no
k1=in
drive=drive+add*a(i1+1)
redn=rc1+rc3
3
4
1
2

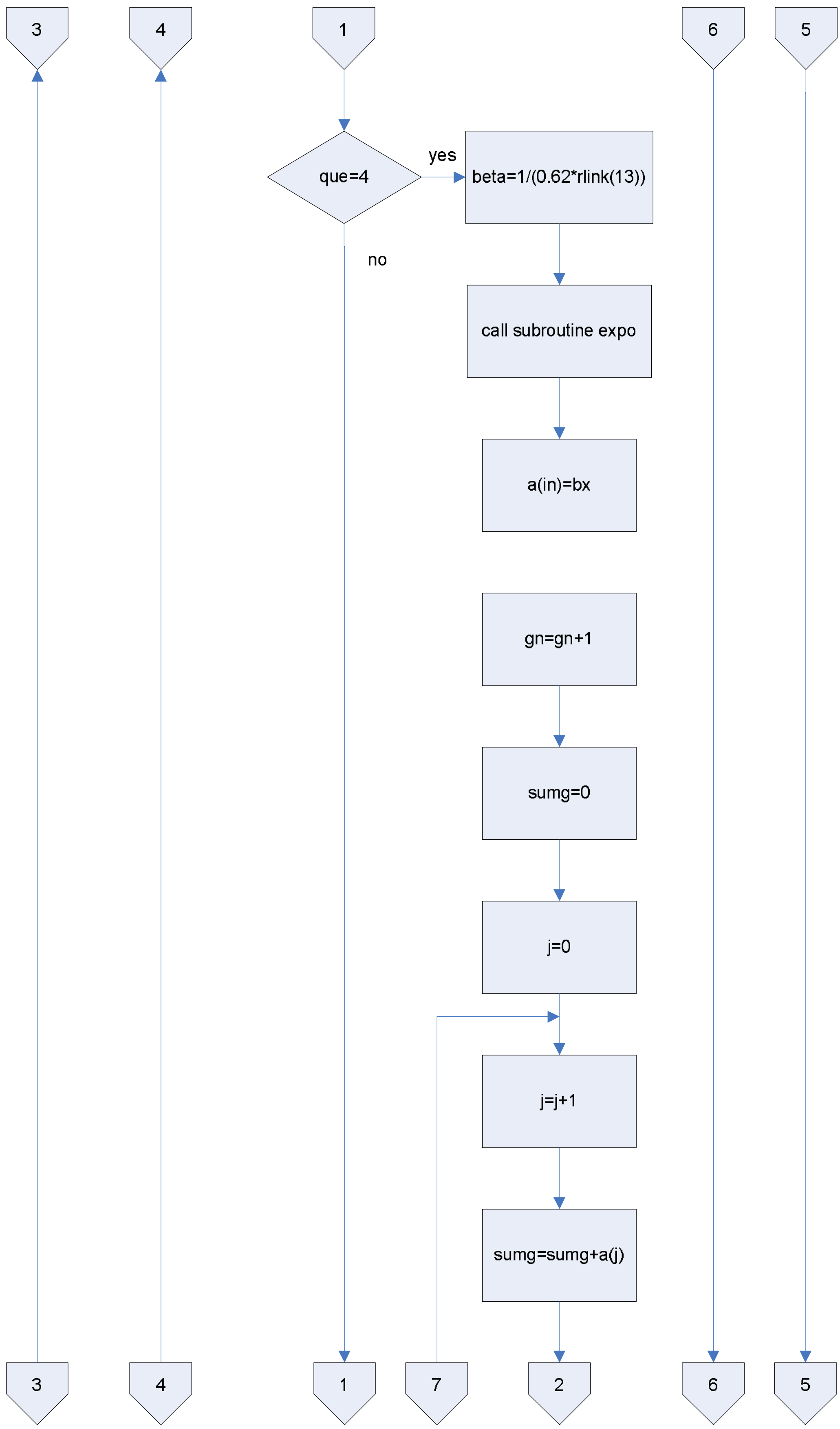

3
4
1
2
g=3*gn/rang2
red=6*redn/rang2
wait=rang2
drive1=cdrive+drive
delay1=cdelay+delay
redn1=credn+redn
gn1=cgn+gn
cost=(delay1*gn1*(redn+gn))/(100*redn1*gn*drive1)
3
4
1
2

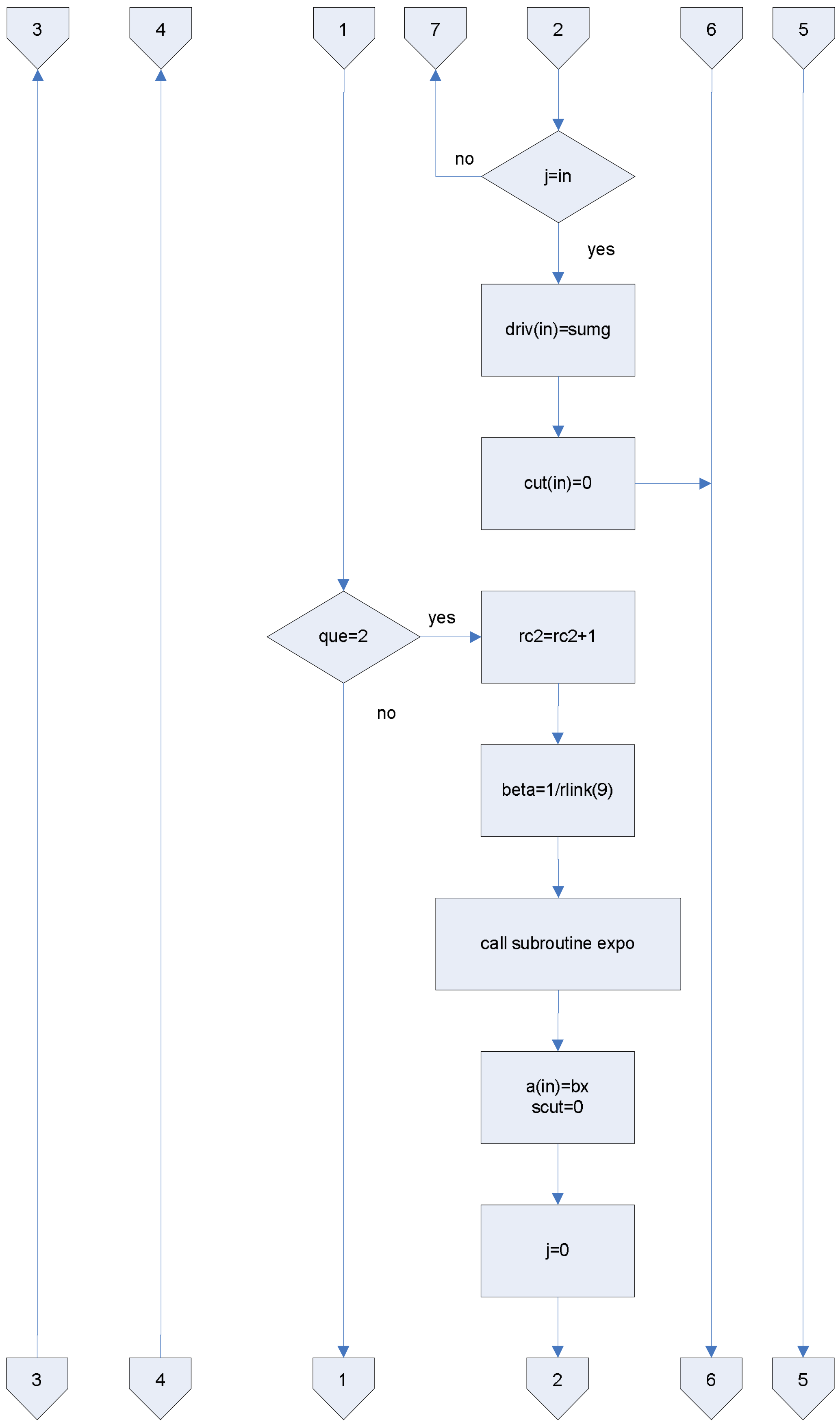

3
4
1
2
call subroutine fuzzy
degree=mu
no
degree=1
yes
ub2(c1)=wait
c1=c1+1
yes
rang>1800
no
gn=rc3
add=rc3
3
1
5

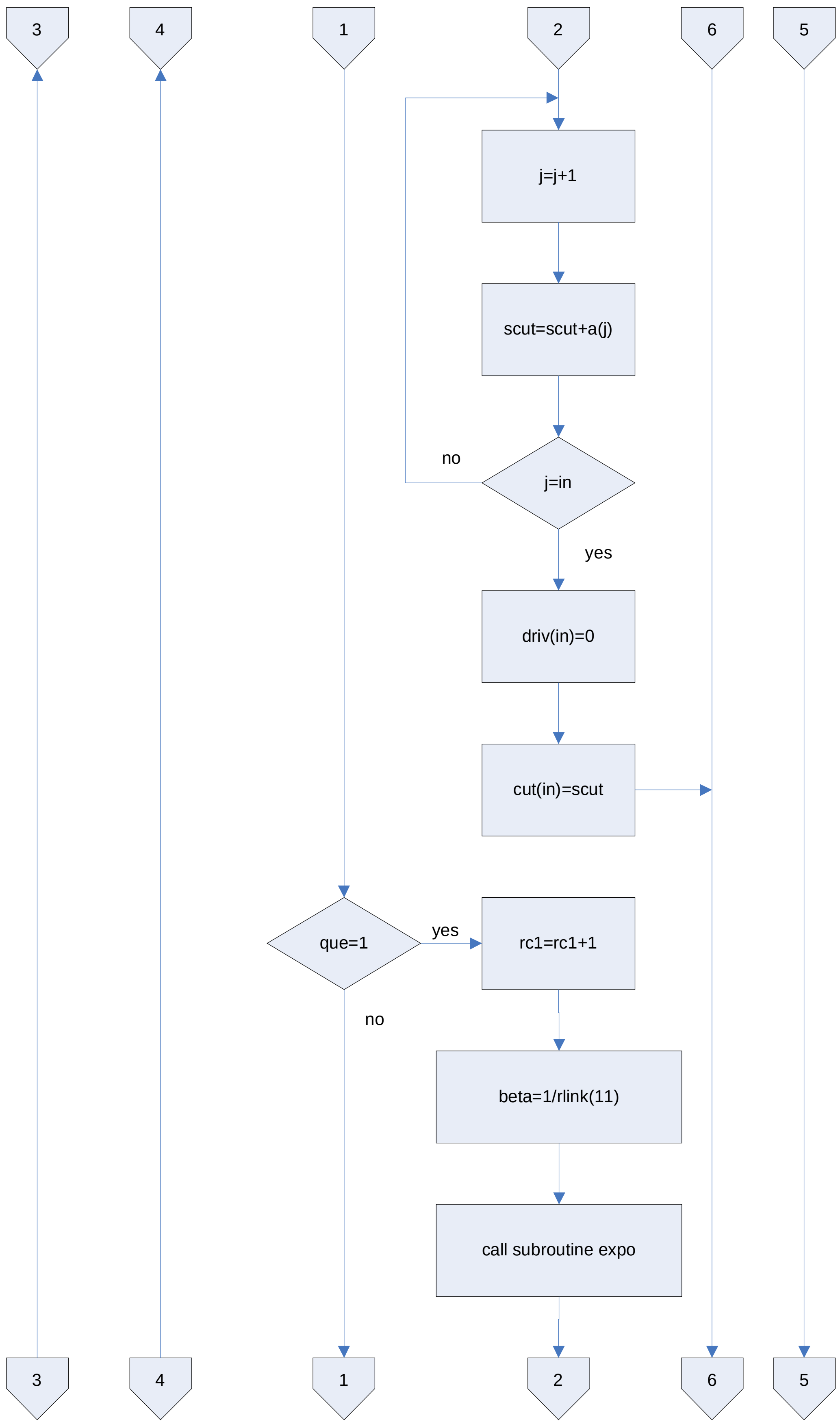

3
1
5
cdrive=drive1
cdelay=delay1
credn=redn1
cgn=gn1
rc2=0
in=0
rang3=0
o1=j1
3
1
5

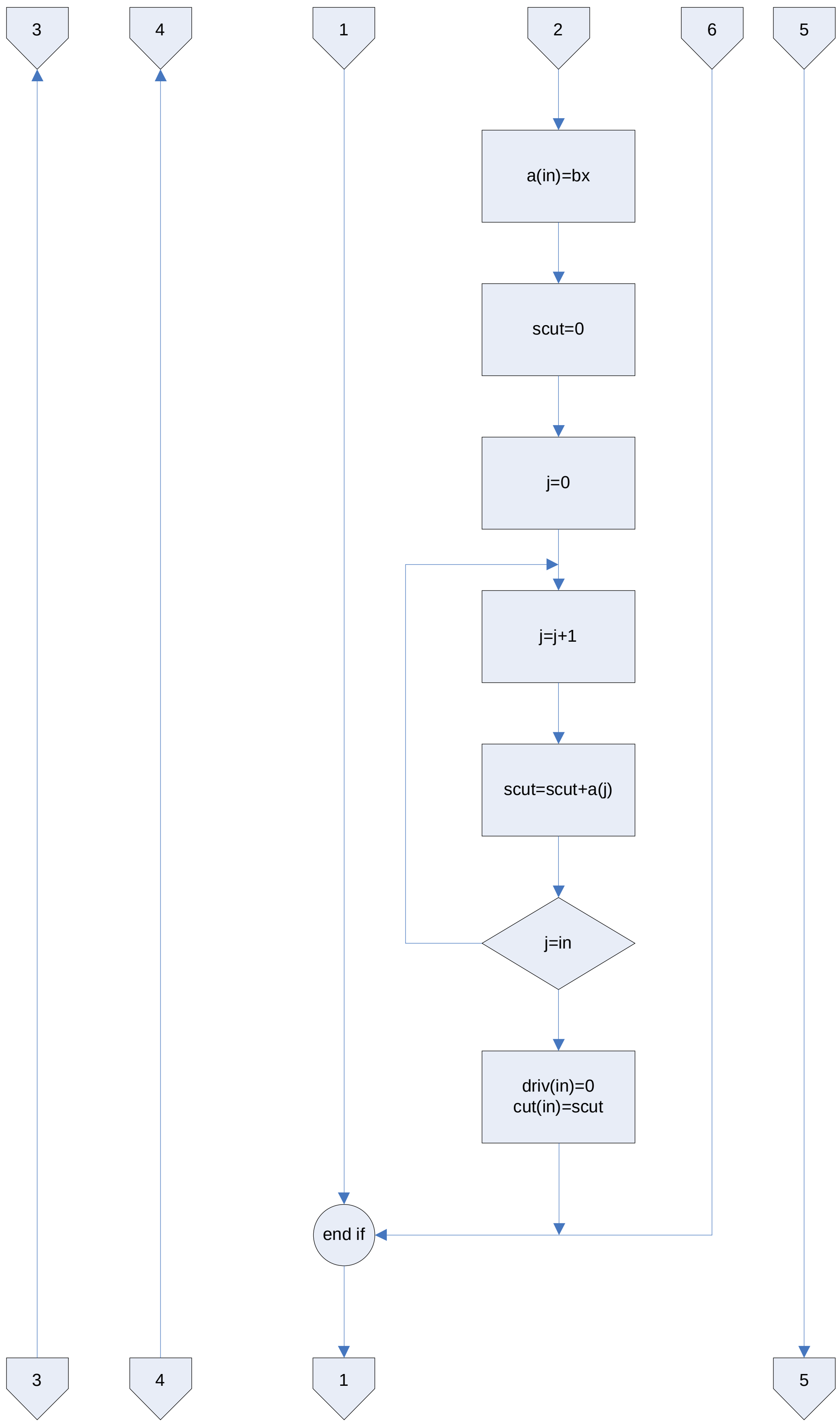

3
1
5
o1=o1+1
in=in+1
p1=rinten(11)/(rinten(11)+rinten(9)+rinten(13))
p2=rinten(9)/(rinten(11)+rinten(9)+rinten(13))
p3=(0.38*rinten(13))/(rinten(11)+rinten(9)+rinten(13))
p4=(0.62*rinten(13))/(rinten(11)+rinten(9)+rinten(13))
call subroutine allocate
que=q
que=3
yes
beta=1/(0.38*rlink(13))
no
call subroutine expo
a(in)=bx
3
4
1
2
5

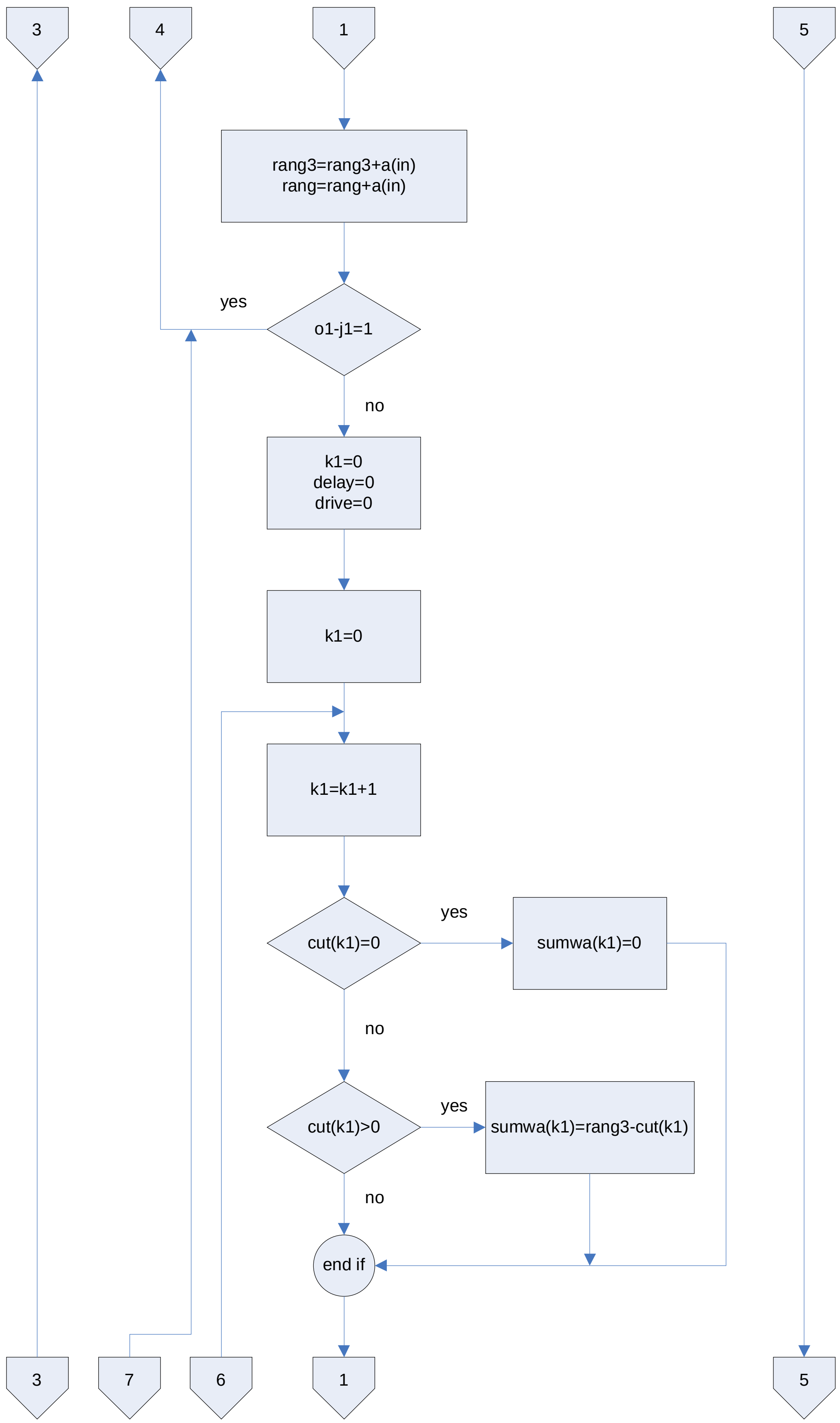

3
4
1
2
5
gn=gn+1
sumg=0
j=0
j=j+1
sumg=sumg+a(j)
no
j=in
yes
driv(in)=sumg
cut(in)=0
3
4
1
6
5

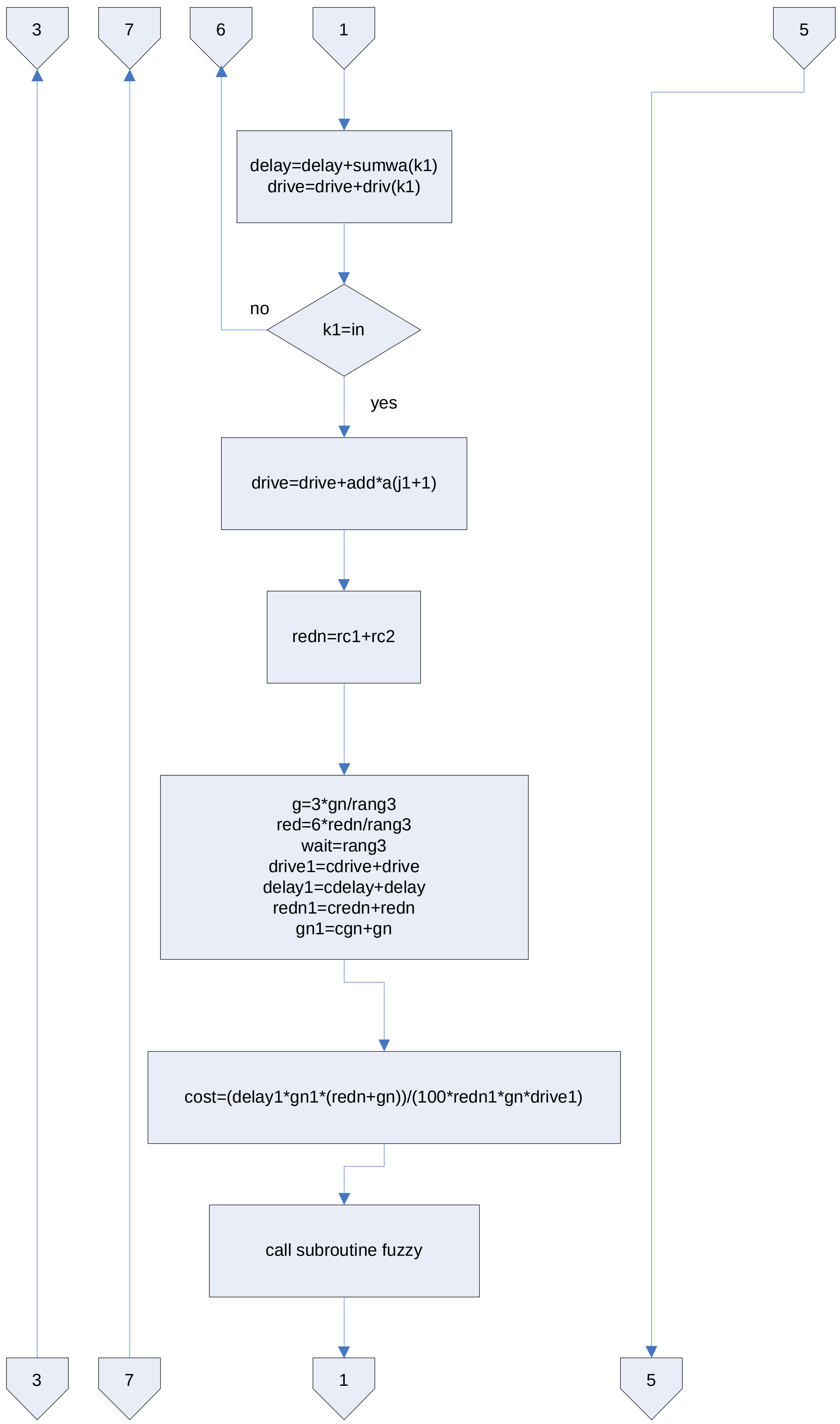

3
4
1
6
5
que=4
yes
beta=1/(0.62*rlink(13))
no
call subroutine expo
a(in)=bx
gn=gn+1
sumg=0
j=0
j=j+1
sumg=sumg+a(j)
3
4
1
7
2
6
5

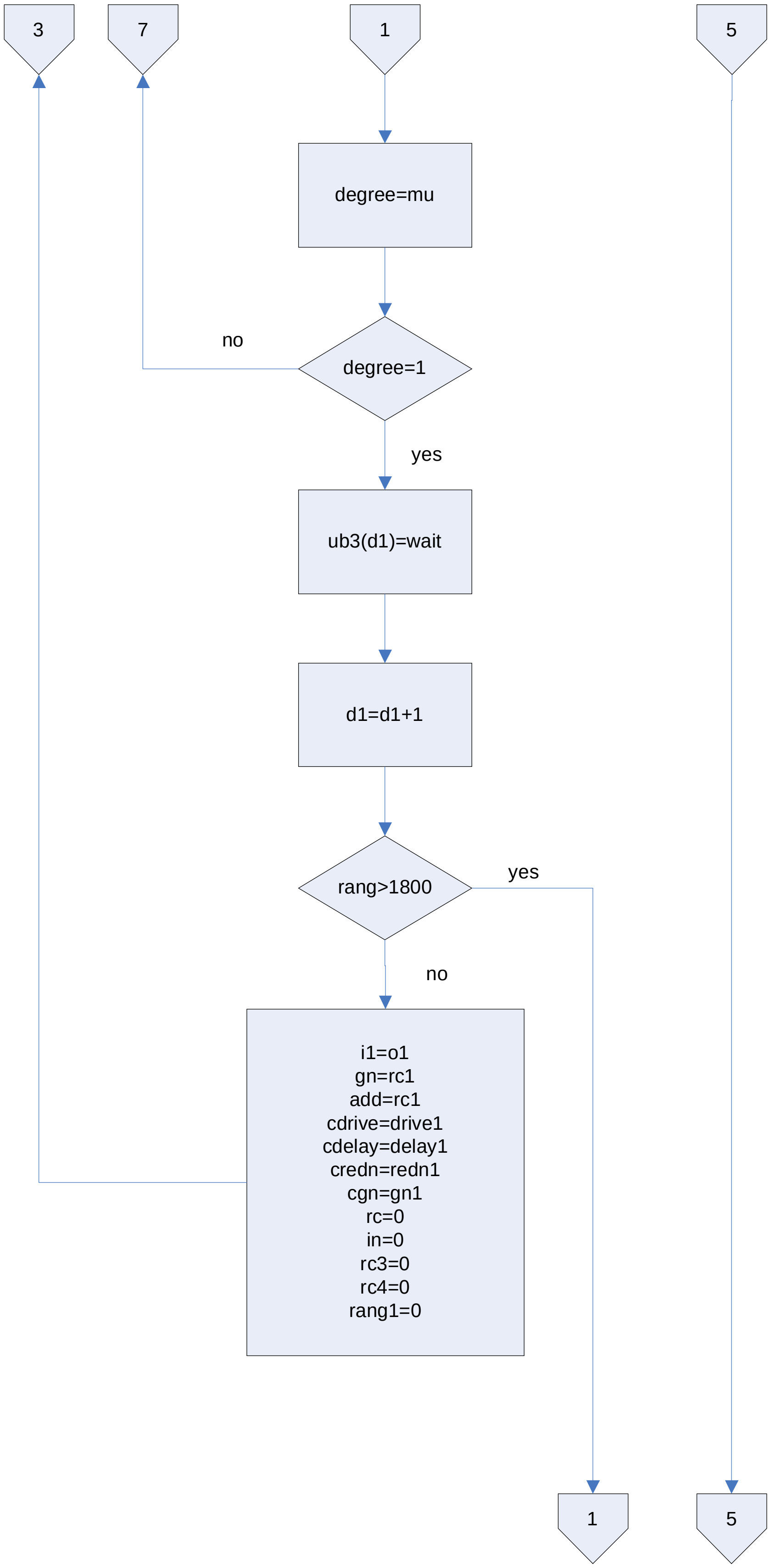

3
4
1
7
2
6
5
j=in
no
yes
driv(in)=sumg
cut(in)=0
que=2
yes
no
rc2=rc2+1
beta=1/rlink(9)
call subroutine expo
a(in)=bx
scut=0
j=0
3
4
1
2
6
5

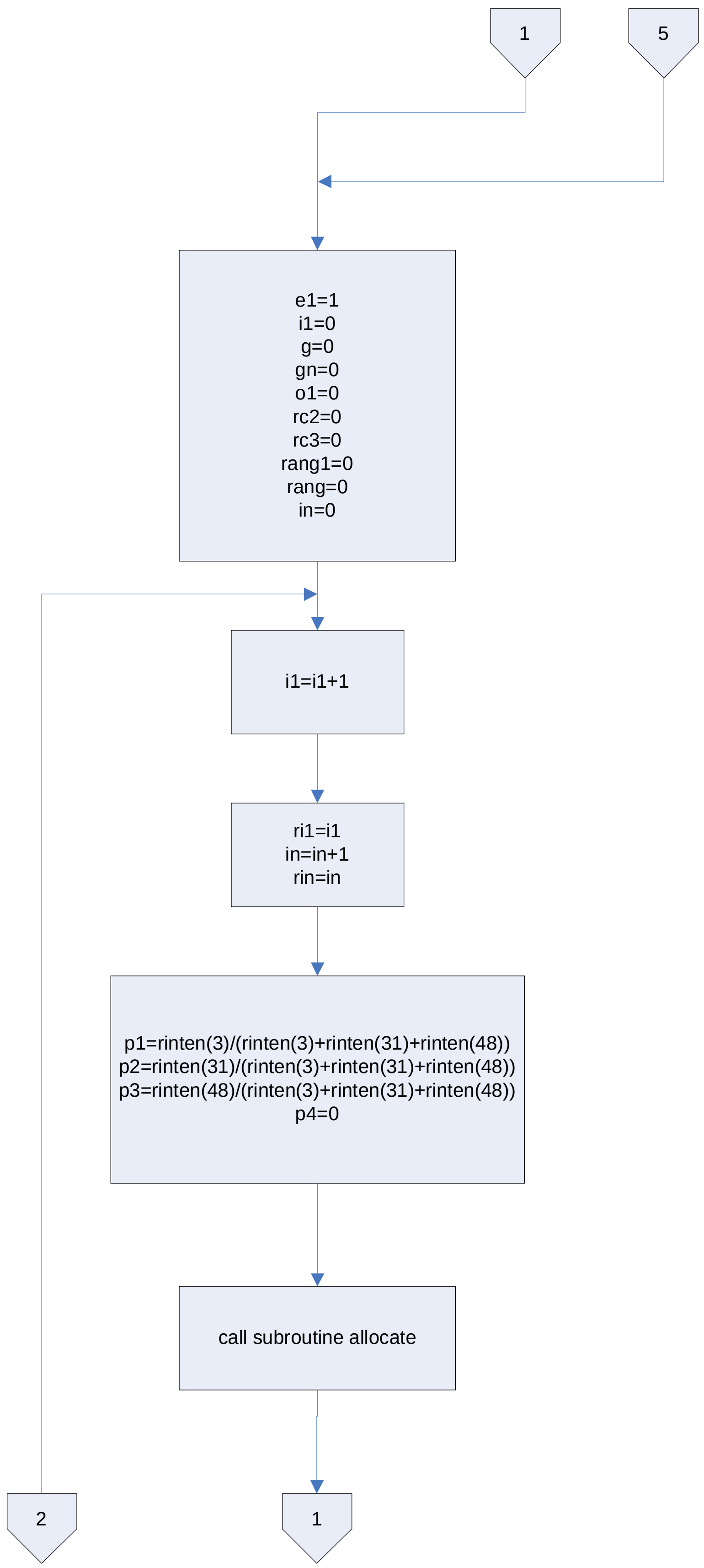

3
4
1
2
6
5
j=j+1
scut=scut+a(j)
no
j=in
yes
driv(in)=0
cut(in)=scut
que=1
yes
rc1=rc1+1
no
beta=1/rlink(11)
call subroutine expo
3
4
1
2
6
5

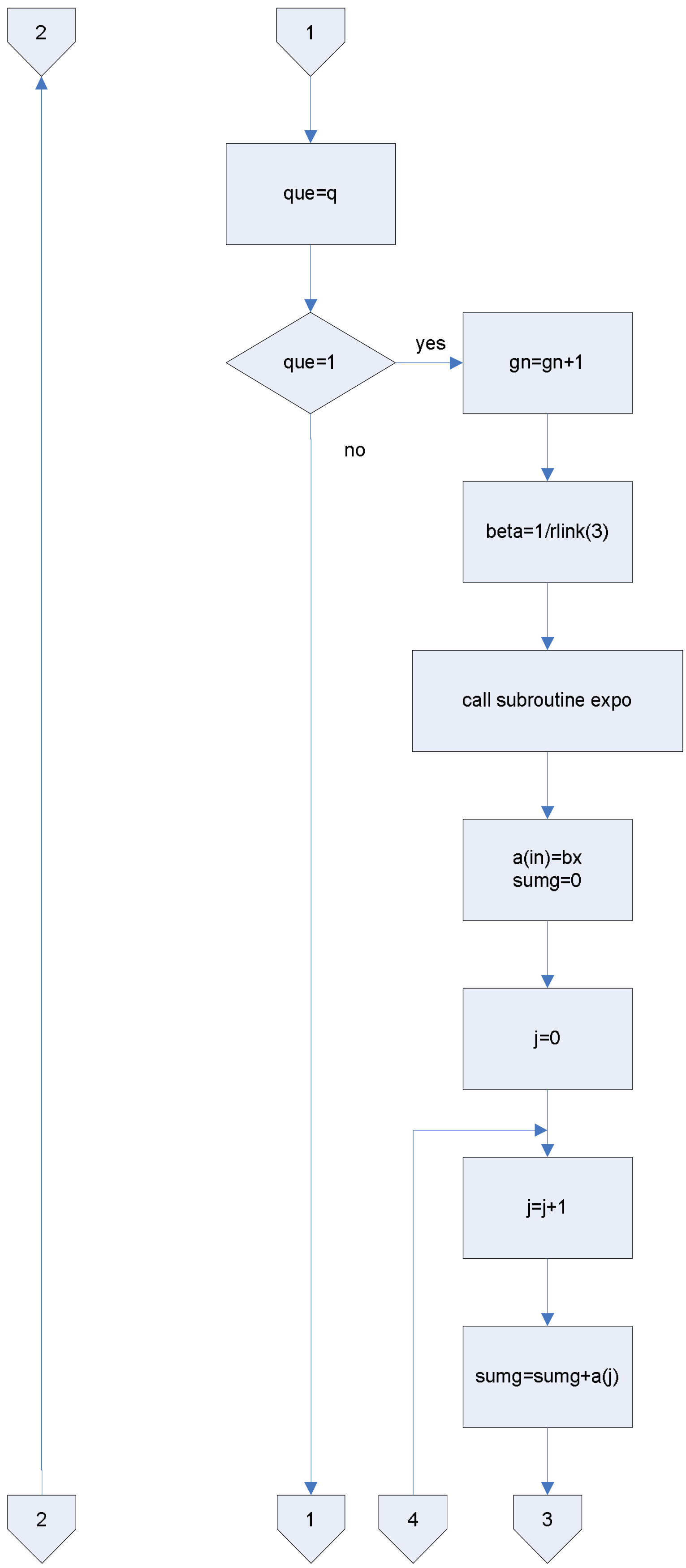

3
4
1
2
6
5
a(in)=bx
scut=0
j=0
j=j+1
scut=scut+a(j)
j=in
driv(in)=0
cut(in)=scut
end if
3
4
1
5

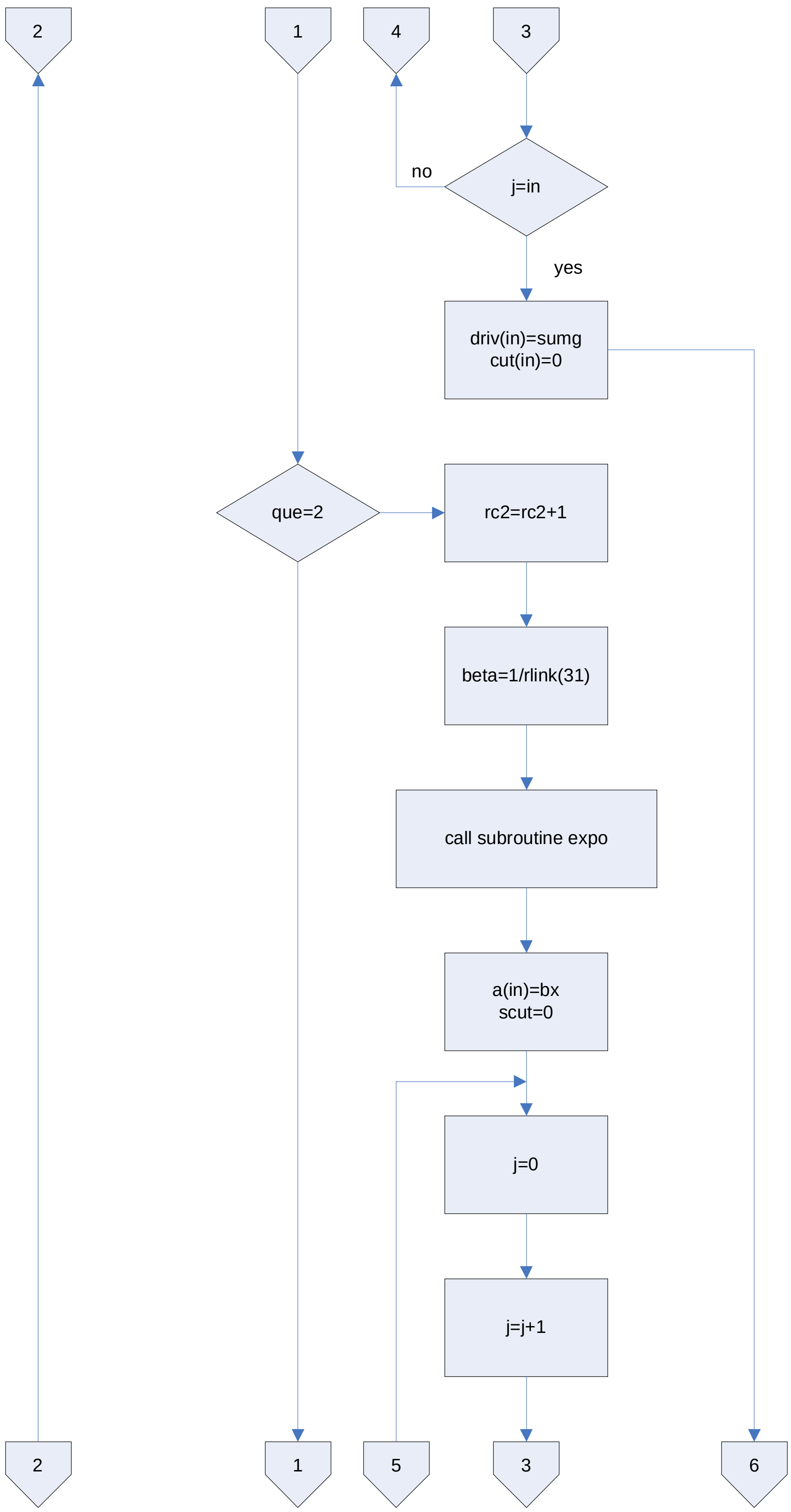

3
4
1
5
rang3=rang3+a(in)
rang=rang+a(in)
yes
o1-j1=1
no
k1=0
delay=0
drive=0
k1=0
k1=k1+1
cut(k1)=0
yes
sumwa(k1)=0
no
cut(k1)>0
yes
sumwa(k1)=rang3-cut(k1)
no
end if
3
7
6
1
5

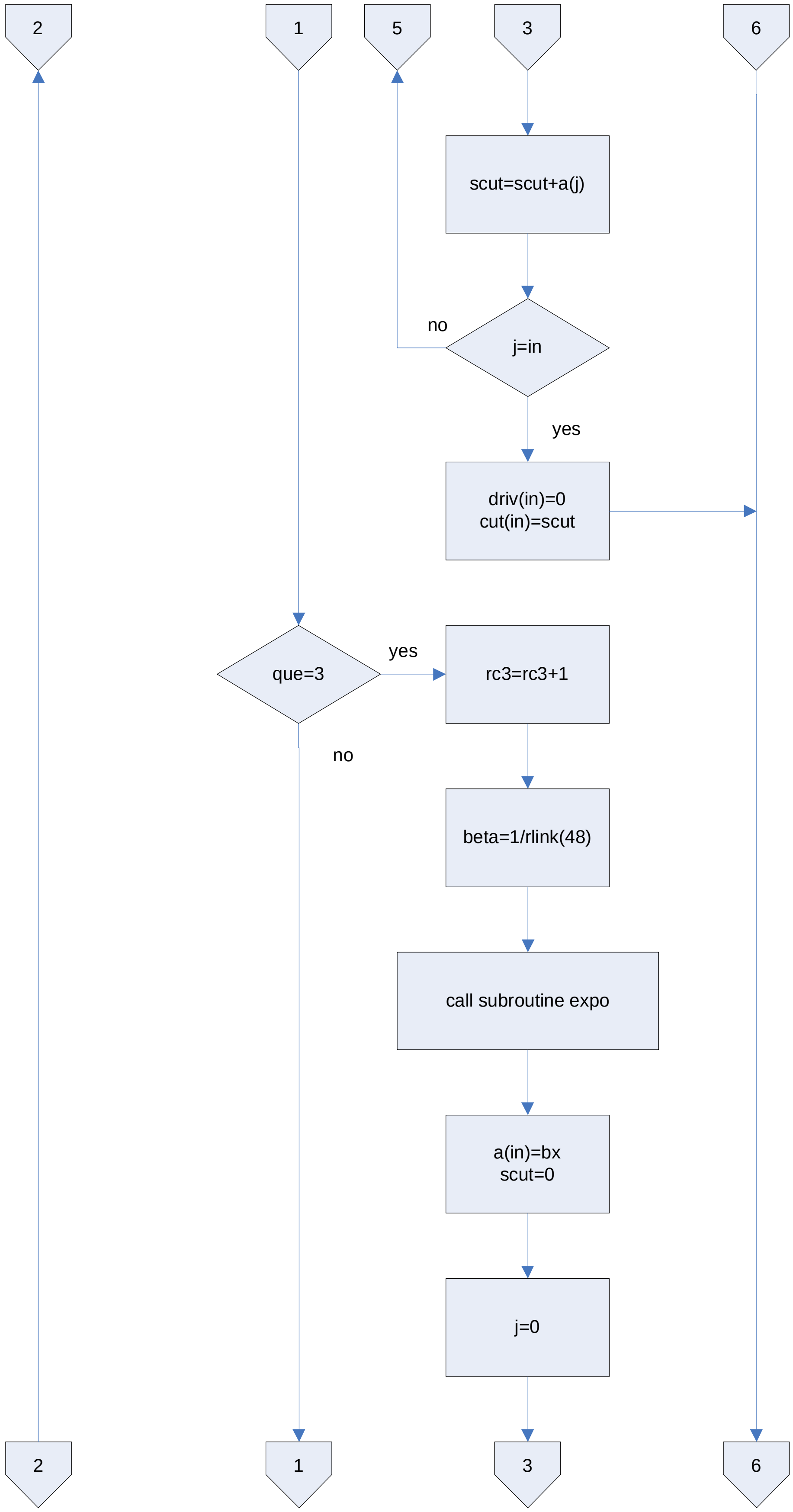

3
7
6
1
5
delay=delay+sumwa(k1)
drive=drive+driv(k1)
k1=in
no
yes
drive=drive+add*a(j1+1)
redn=rc1+rc2
g=3*gn/rang3
red=6*redn/rang3
wait=rang3
drive1=cdrive+drive
delay1=cdelay+delay
redn1=credn+redn
gn1=cgn+gn
cost=(delay1*gn1*(redn+gn))/(100*redn1*gn*drive1)
call subroutine fuzzy
3
7
1
5

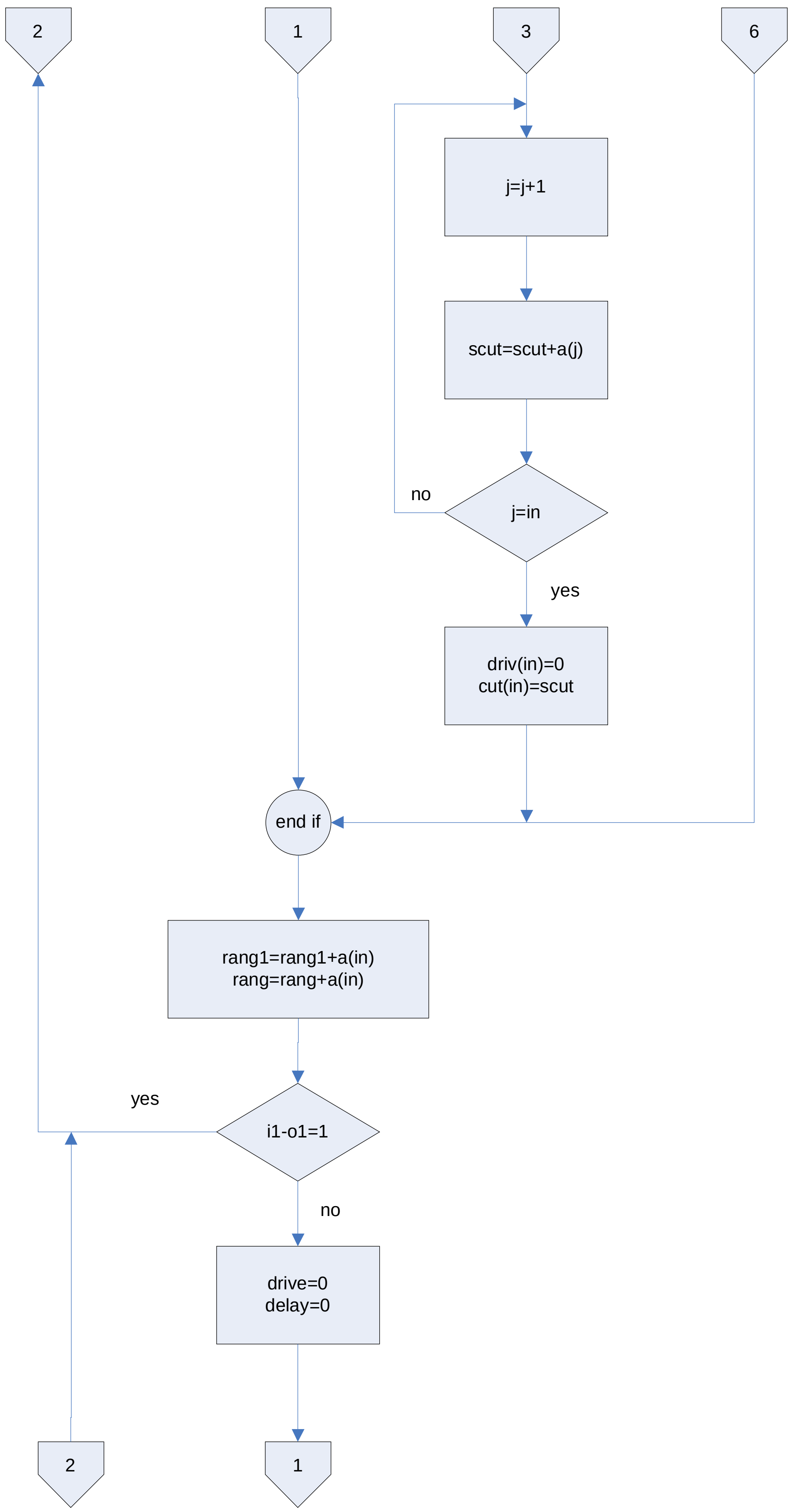

3
7
1
5
degree=mu
no
degree=1
yes
ub3(d1)=wait
d1=d1+1
yes
rang>1800
no
i1=o1
gn=rc1
add=rc1
cdrive=drive1
cdelay=delay1
credn=redn1
cgn=gn1
rc=0
in=0
rc3=0
rc4=0
rang1=0
1
5

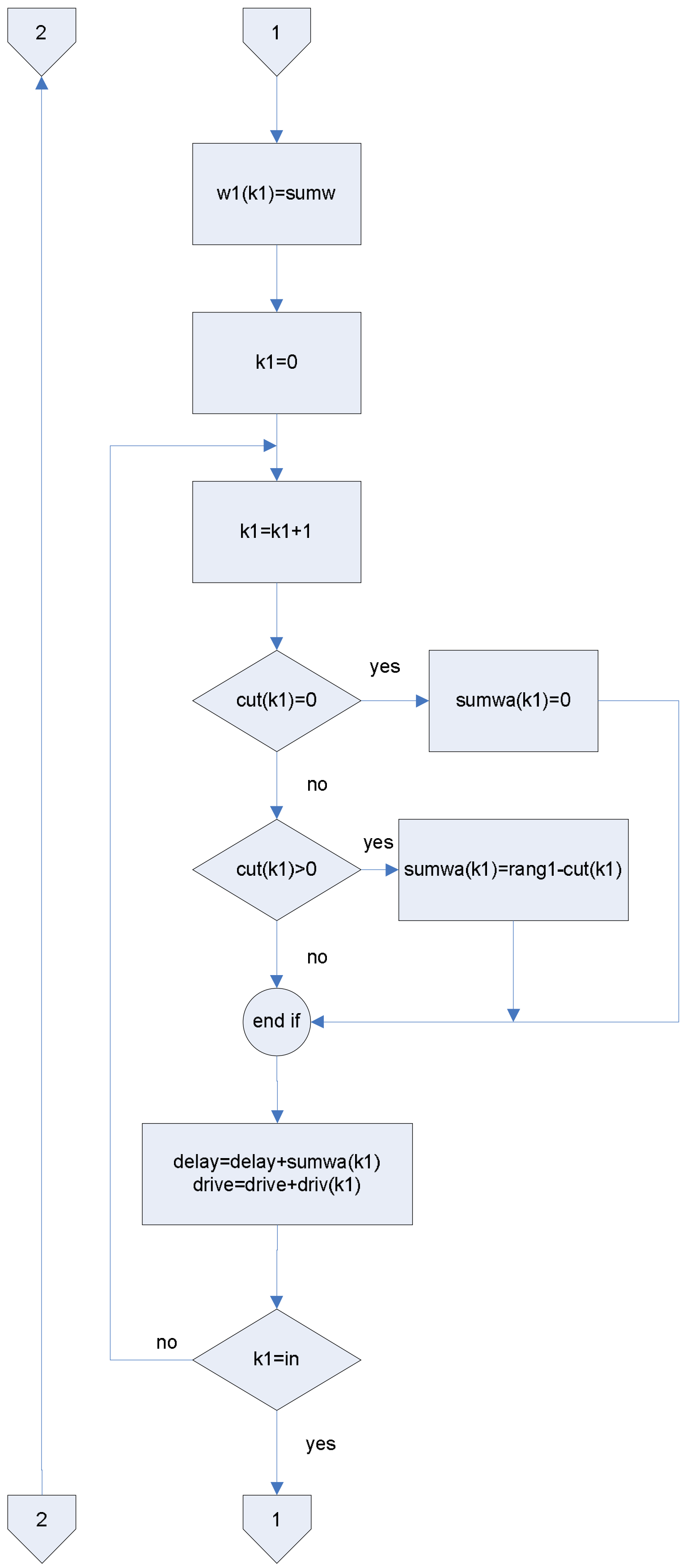

1
5
e1=1
i1=0
g=0
gn=0
o1=0
rc2=0
rc3=0
rang1=0
rang=0
in=0
i1=i1+1
ri1=i1
in=in+1
rin=in
p1=rinten(3)/(rinten(3)+rinten(31)+rinten(48))
p2=rinten(31)/(rinten(3)+rinten(31)+rinten(48))
p3=rinten(48)/(rinten(3)+rinten(31)+rinten(48))
p4=0
call subroutine allocate
2
1

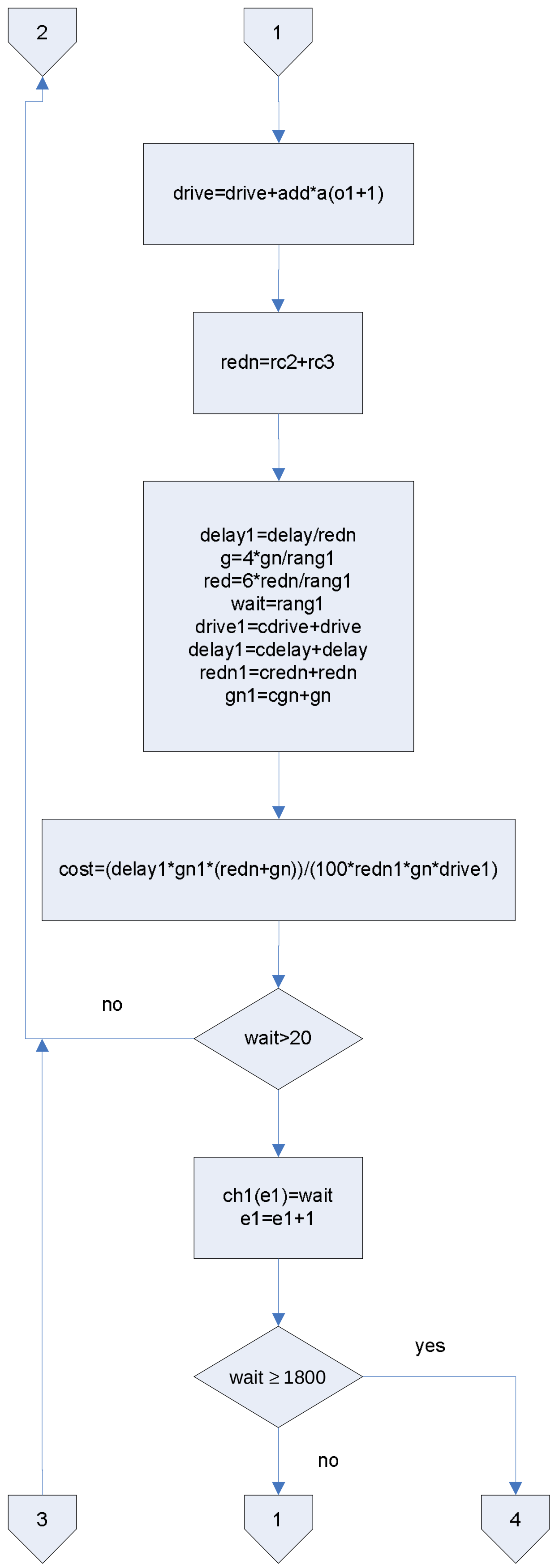

2
1
que=q
que=1
yes
no
gn=gn+1
beta=1/rlink(3)
call subroutine expo
a(in)=bx
sumg=0
j=0
j=j+1
sumg=sumg+a(j)
2
1
4
3

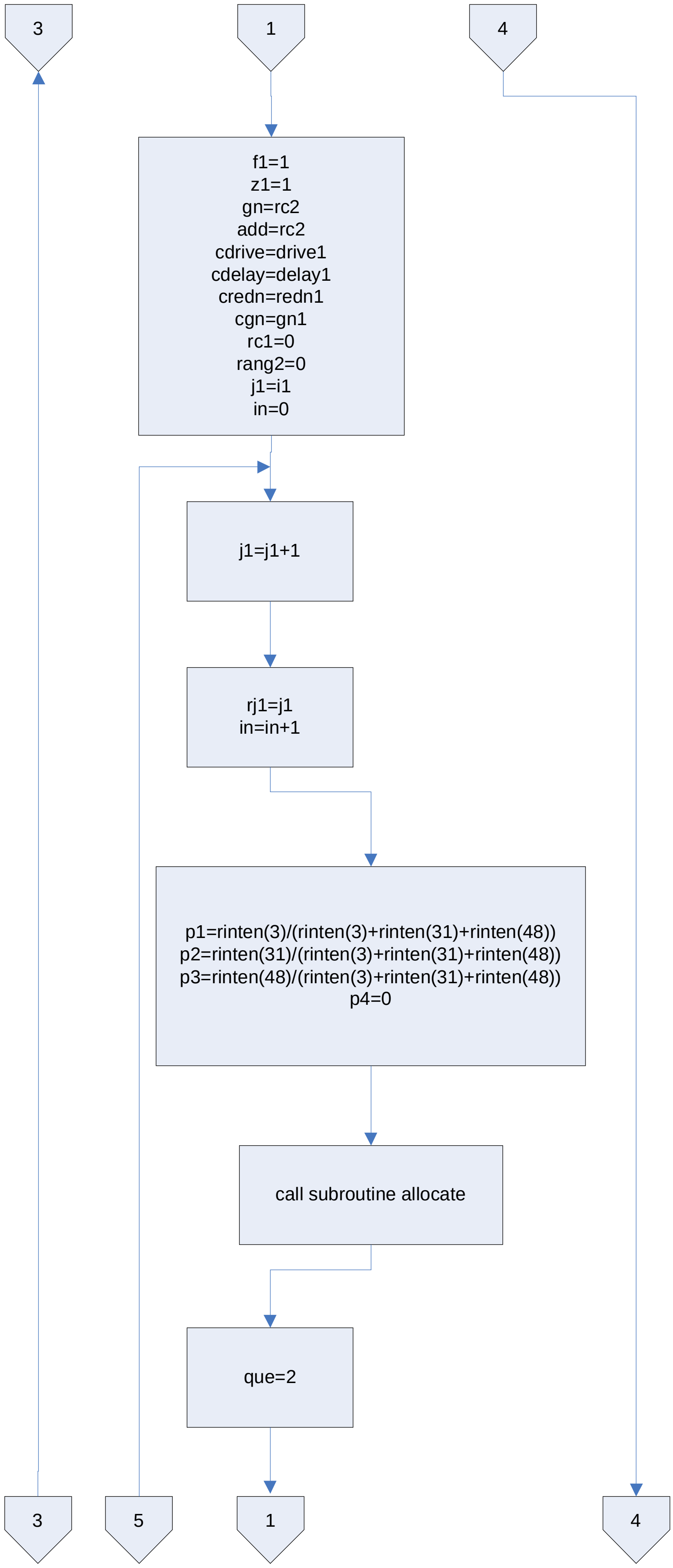

2
1
4
3
no
j=in
yes
driv(in)=sumg
cut(in)=0
que=2
rc2=rc2+1
beta=1/rlink(31)
call subroutine expo
a(in)=bx
scut=0
j=0
j=j+1
2
1
5
3
6

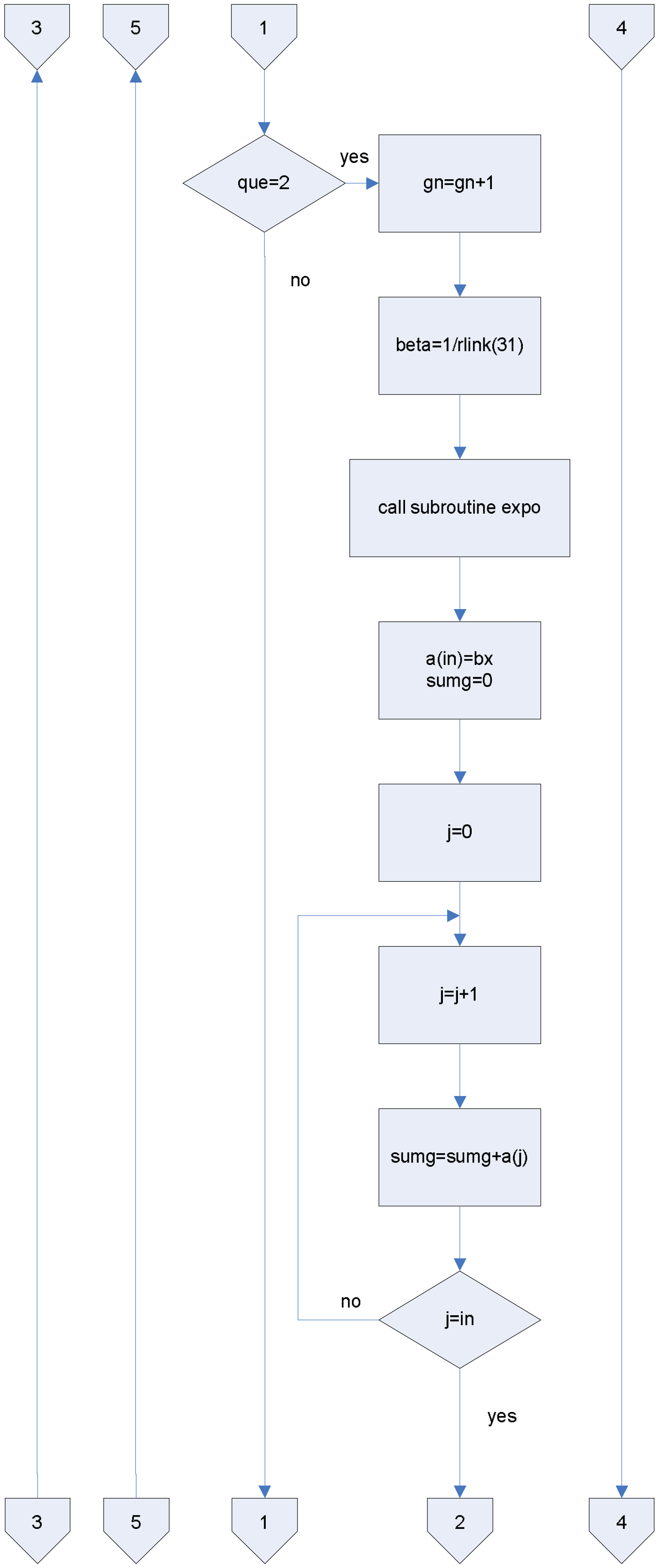

2
1
5
3
6
scut=scut+a(j)
no
j=in
yes
driv(in)=0
cut(in)=scut
que=3
yes
rc3=rc3+1
no
beta=1/rlink(48)
call subroutine expo
a(in)=bx
scut=0
j=0
2
1
3
6

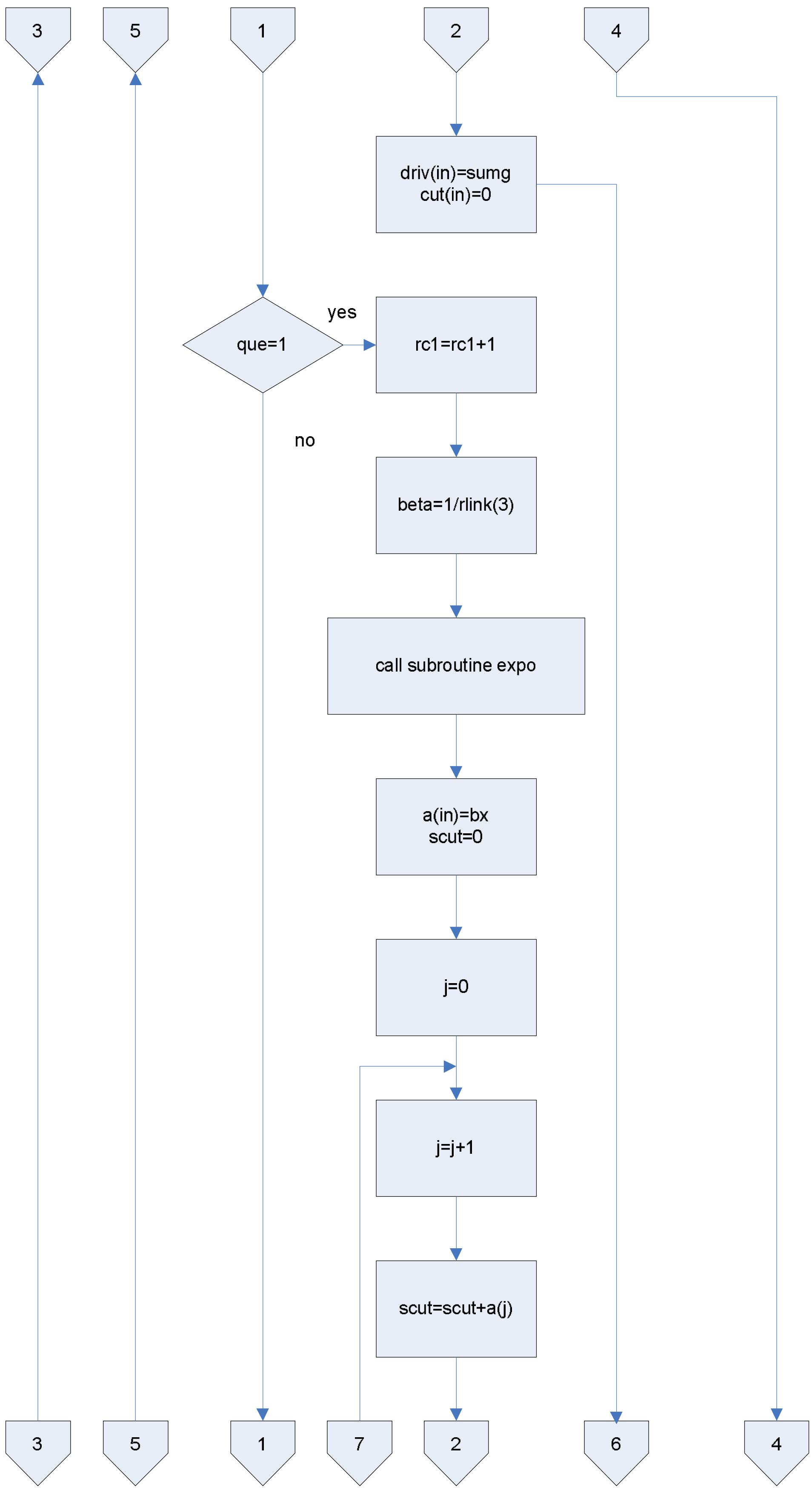

2
1
3
6
j=j+1
scut=scut+a(j)
j=in
no
yes
driv(in)=0
cut(in)=scut
end if
rang1=rang1+a(in)
rang=rang+a(in)
yes
i1-o1=1
no
drive=0
delay=0
2
1

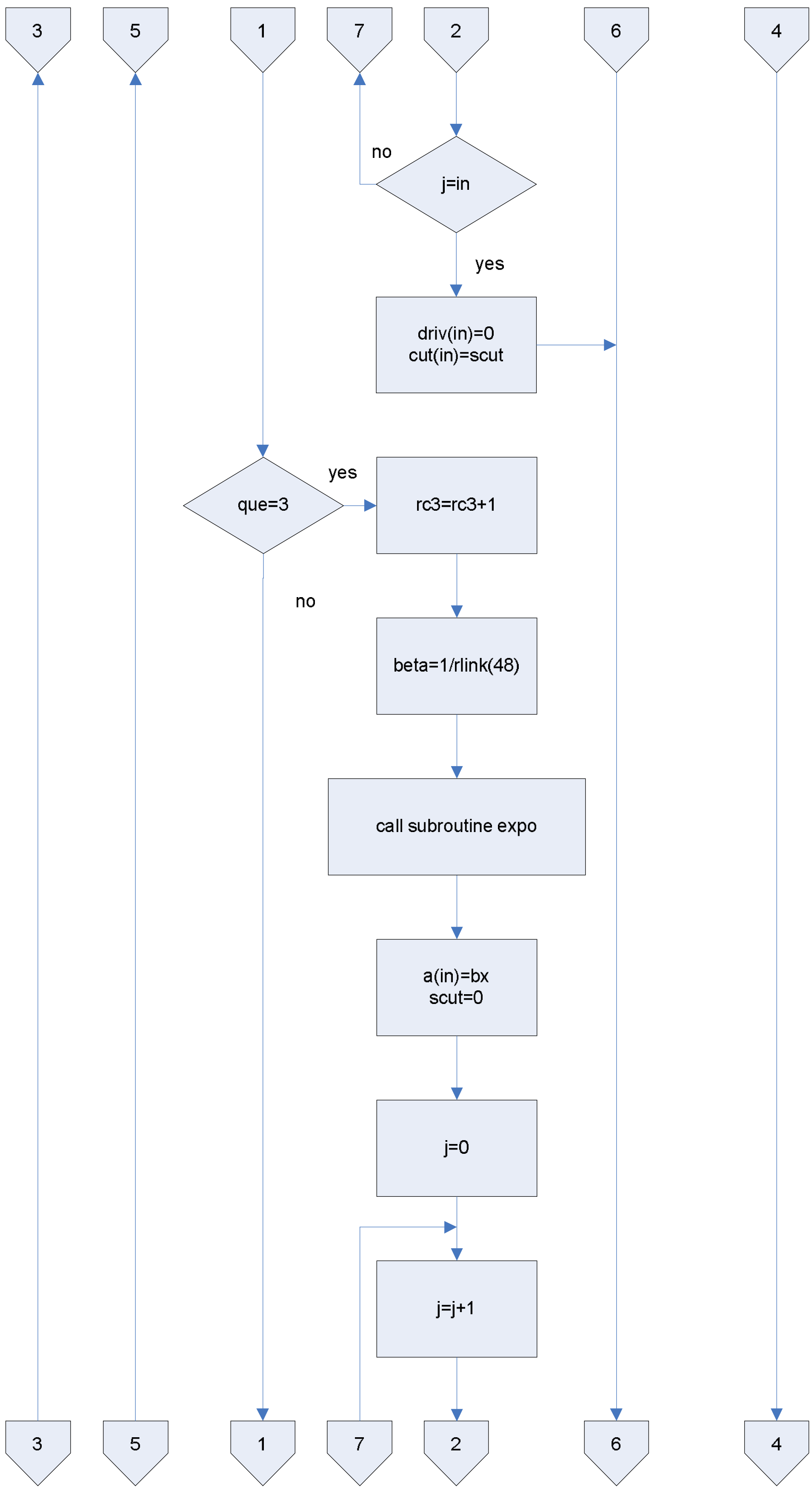

2
1
w1(k1)=sumw
k1=0
k1=k1+1
cut(k1)=0
yes
sumwa(k1)=0
no
cut(k1)>0
yes
sumwa(k1)=rang1-cut(k1)
no
end if
delay=delay+sumwa(k1)
drive=drive+driv(k1)
no
k1=in
yes
2
1

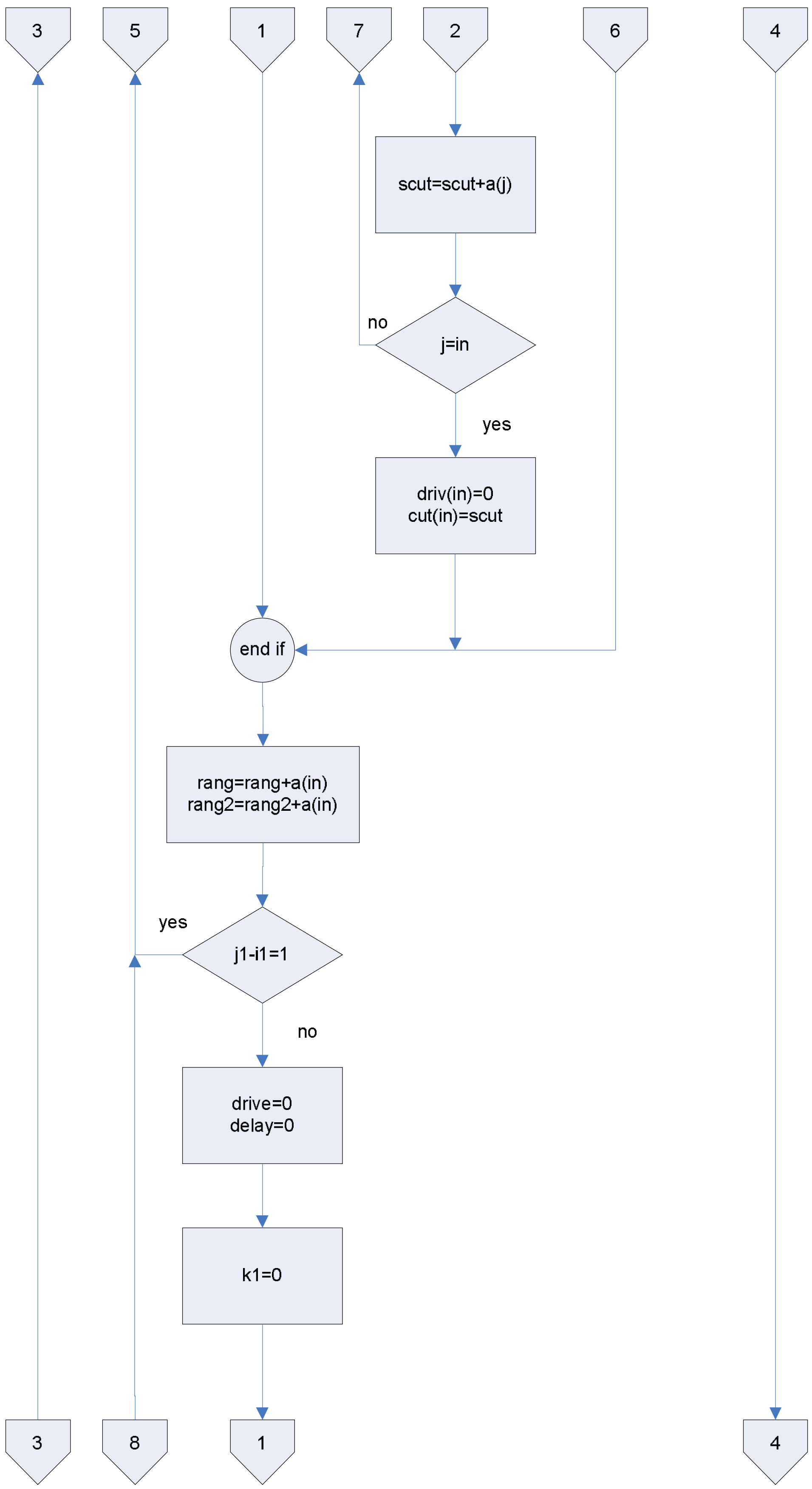

2
1
drive=drive+add*a(o1+1)
redn=rc2+rc3
delay1=delay/redn
g=4*gn/rang1
red=6*redn/rang1
wait=rang1
drive1=cdrive+drive
delay1=cdelay+delay
redn1=credn+redn
gn1=cgn+gn
cost=(delay1*gn1*(redn+gn))/(100*redn1*gn*drive1)
no
wait>20
ch1(e1)=wait
e1=e1+1
yes
wait ≥ 1800
no
3
1
4

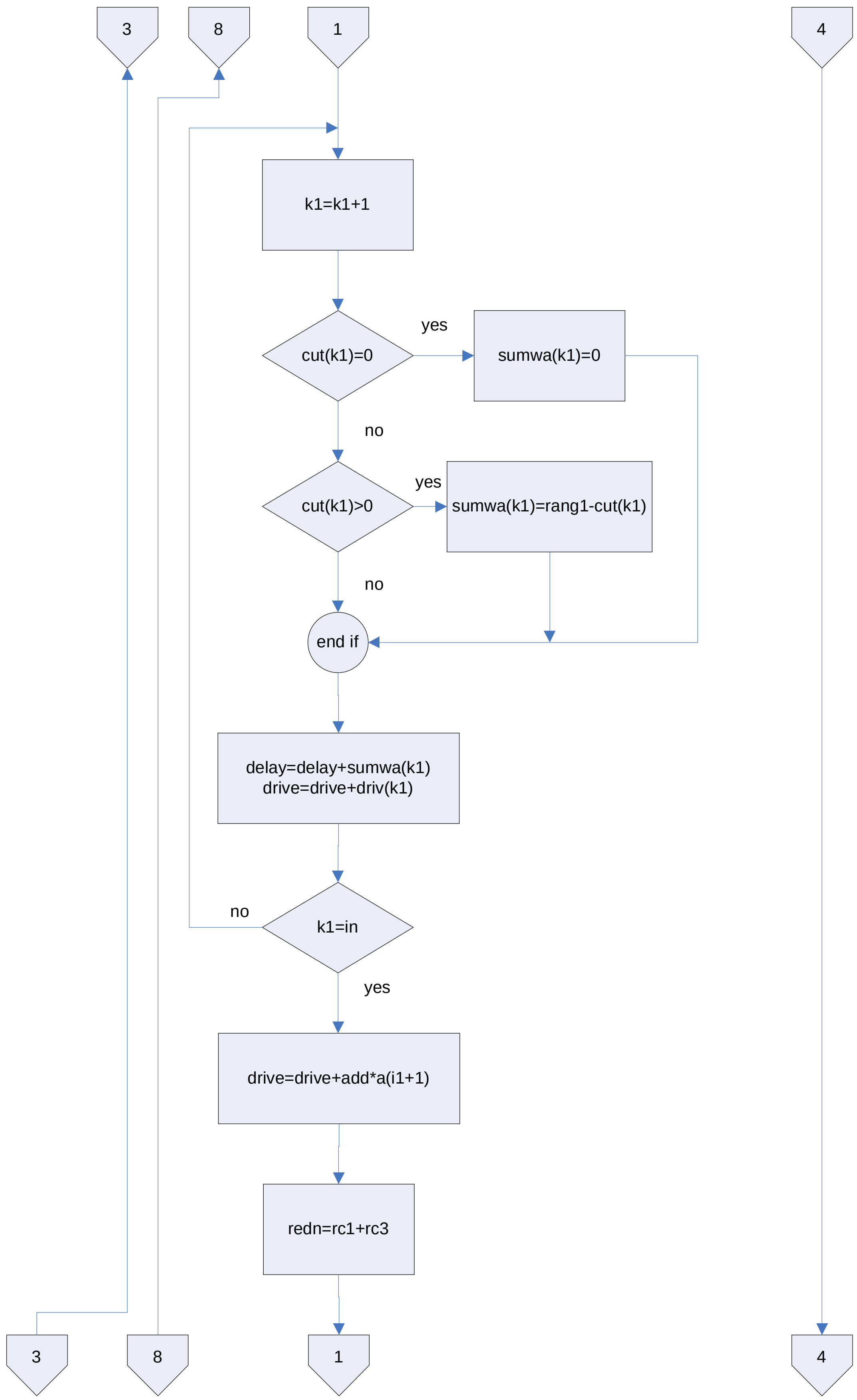

3
1
4
f1=1
z1=1
gn=rc2
add=rc2
cdrive=drive1
cdelay=delay1
credn=redn1
cgn=gn1
rc1=0
rang2=0
j1=i1
in=0
j1=j1+1
rj1=j1
in=in+1
p1=rinten(3)/(rinten(3)+rinten(31)+rinten(48))
p2=rinten(31)/(rinten(3)+rinten(31)+rinten(48))
p3=rinten(48)/(rinten(3)+rinten(31)+rinten(48))
p4=0
call subroutine allocate
que=2
3
5
1
4

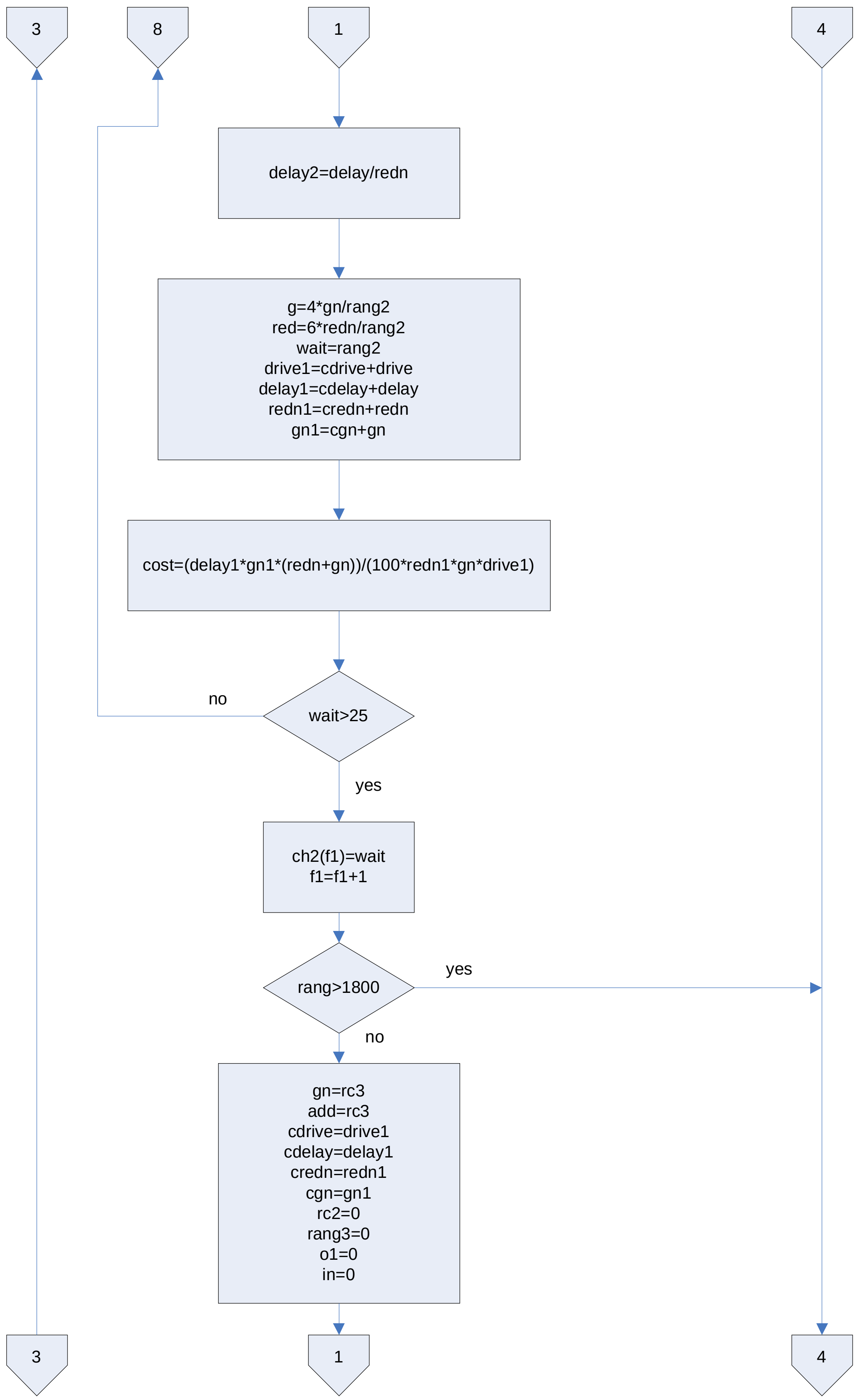

3
5
1
4
que=2
yes
gn=gn+1
no
beta=1/rlink(31)
call subroutine expo
a(in)=bx
sumg=0
j=0
j=j+1
sumg=sumg+a(j)
no
j=in
yes
3
5
1
2
4

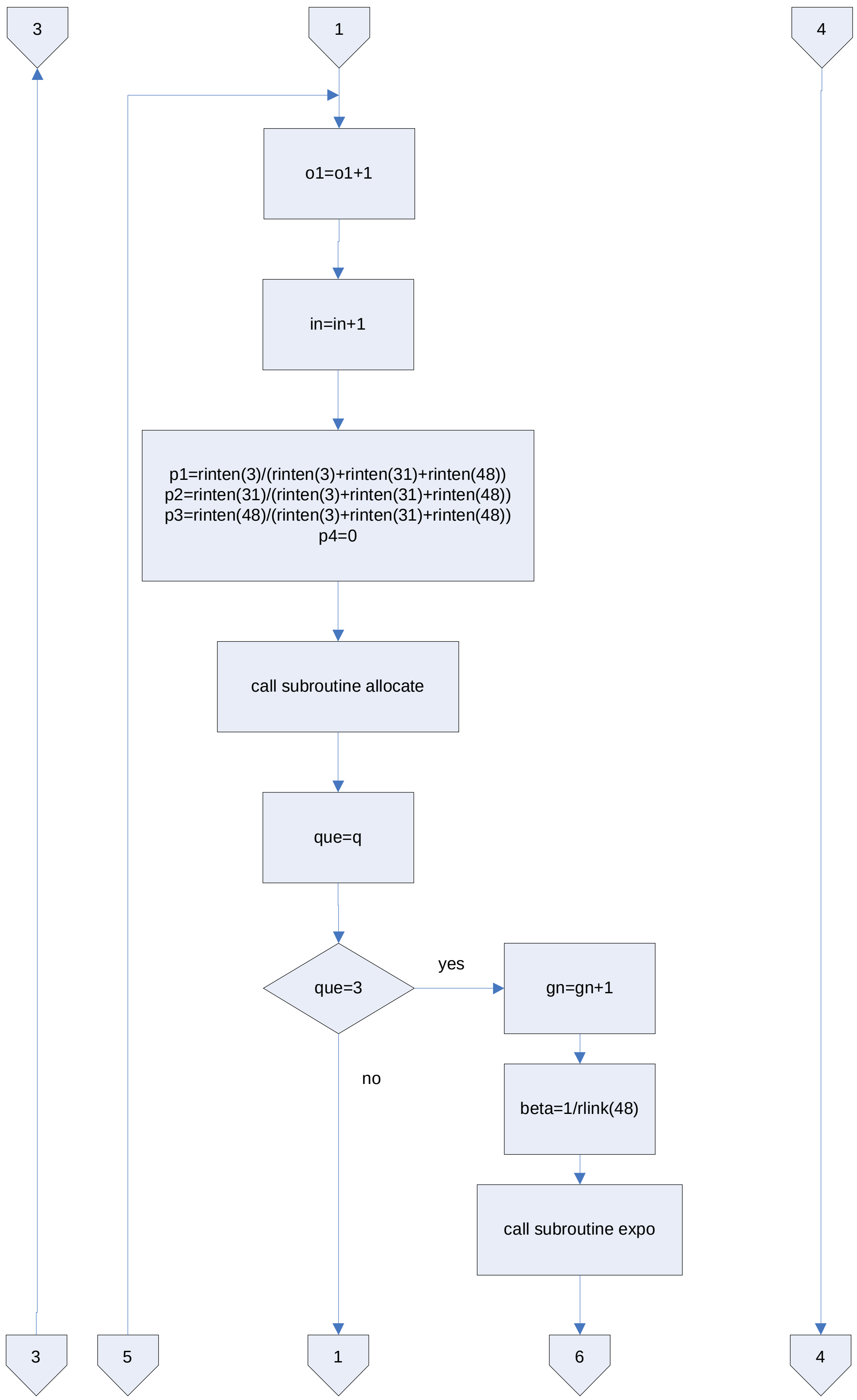

3
5
1
2
4
driv(in)=sumg
cut(in)=0
que=1
yes
rc1=rc1+1
no
beta=1/rlink(3)
call subroutine expo
a(in)=bx
scut=0
j=0
j=j+1
scut=scut+a(j)
3
5
1
7
2
6
4

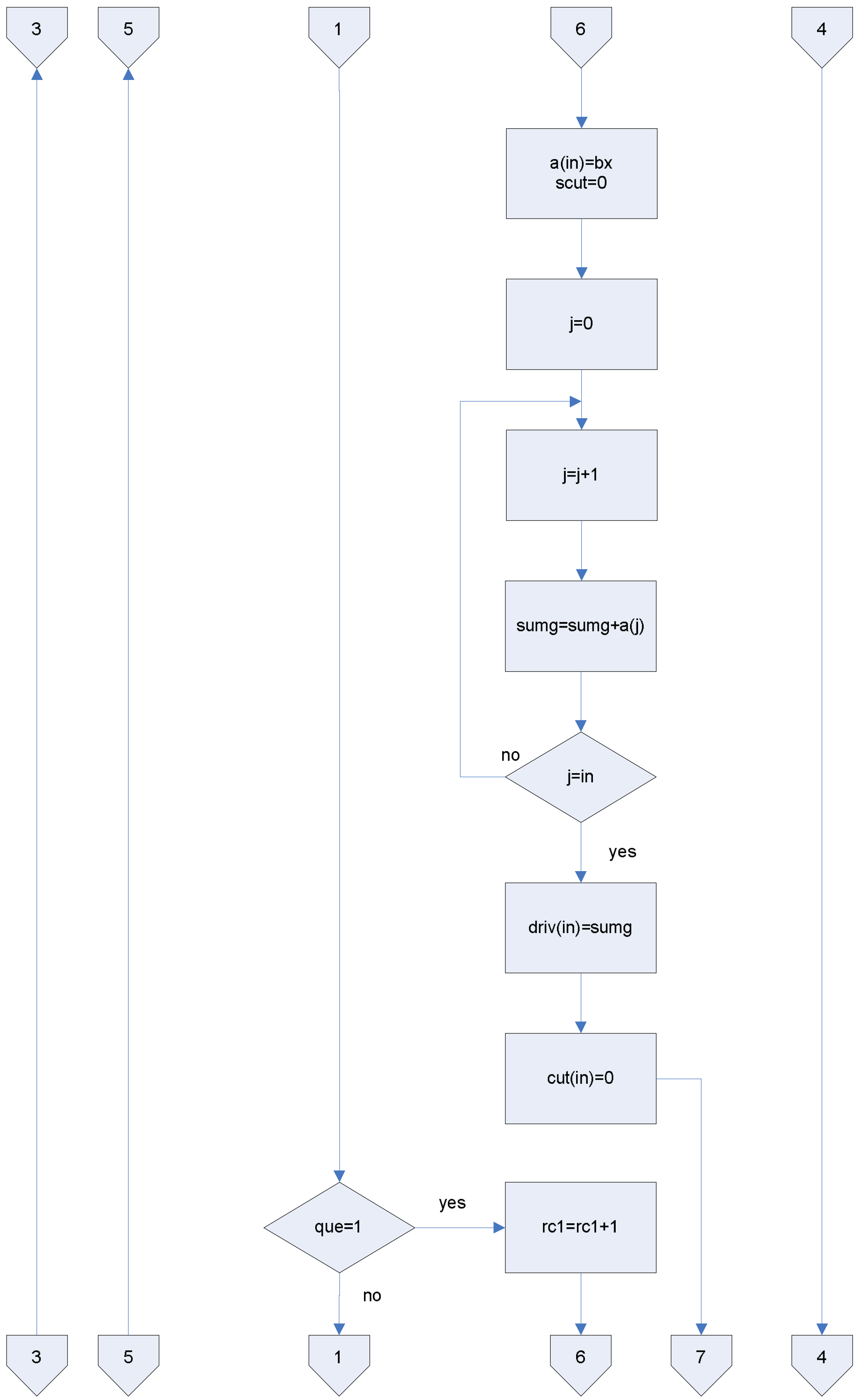

3
5
1
7
2
6
4
no
j=in
yes
driv(in)=0
cut(in)=scut
que=3
yes
rc3=rc3+1
no
beta=1/rlink(48)
call subroutine expo
a(in)=bx
scut=0
j=0
j=j+1
3
5
1
7
2
6
4

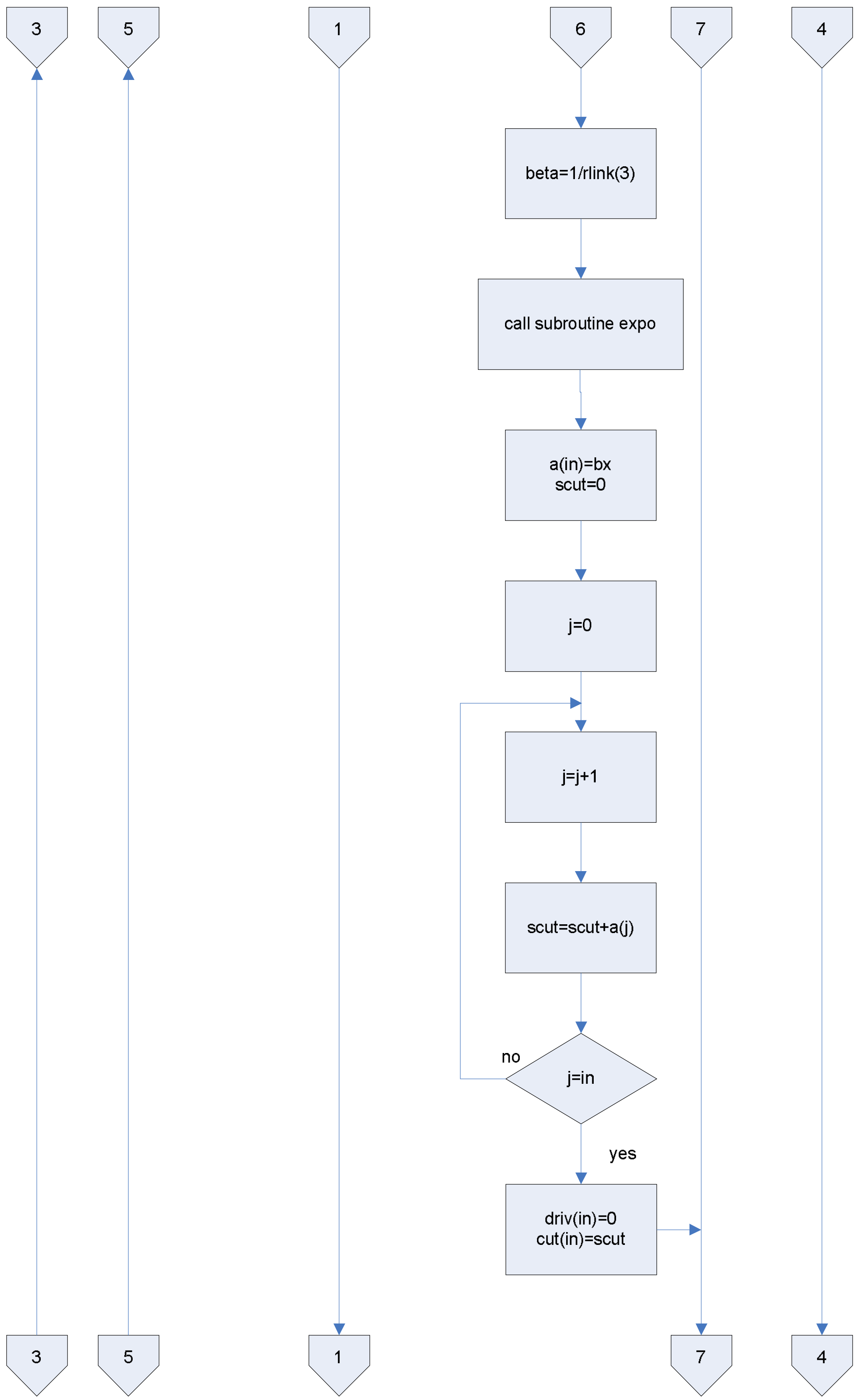

3
5
1
7
2
6
4
scut=scut+a(j)
no
j=in
yes
driv(in)=0
cut(in)=scut
end if
rang=rang+a(in)
rang2=rang2+a(in)
yes
j1-i1=1
no
drive=0
delay=0
k1=0
3
8
1
4

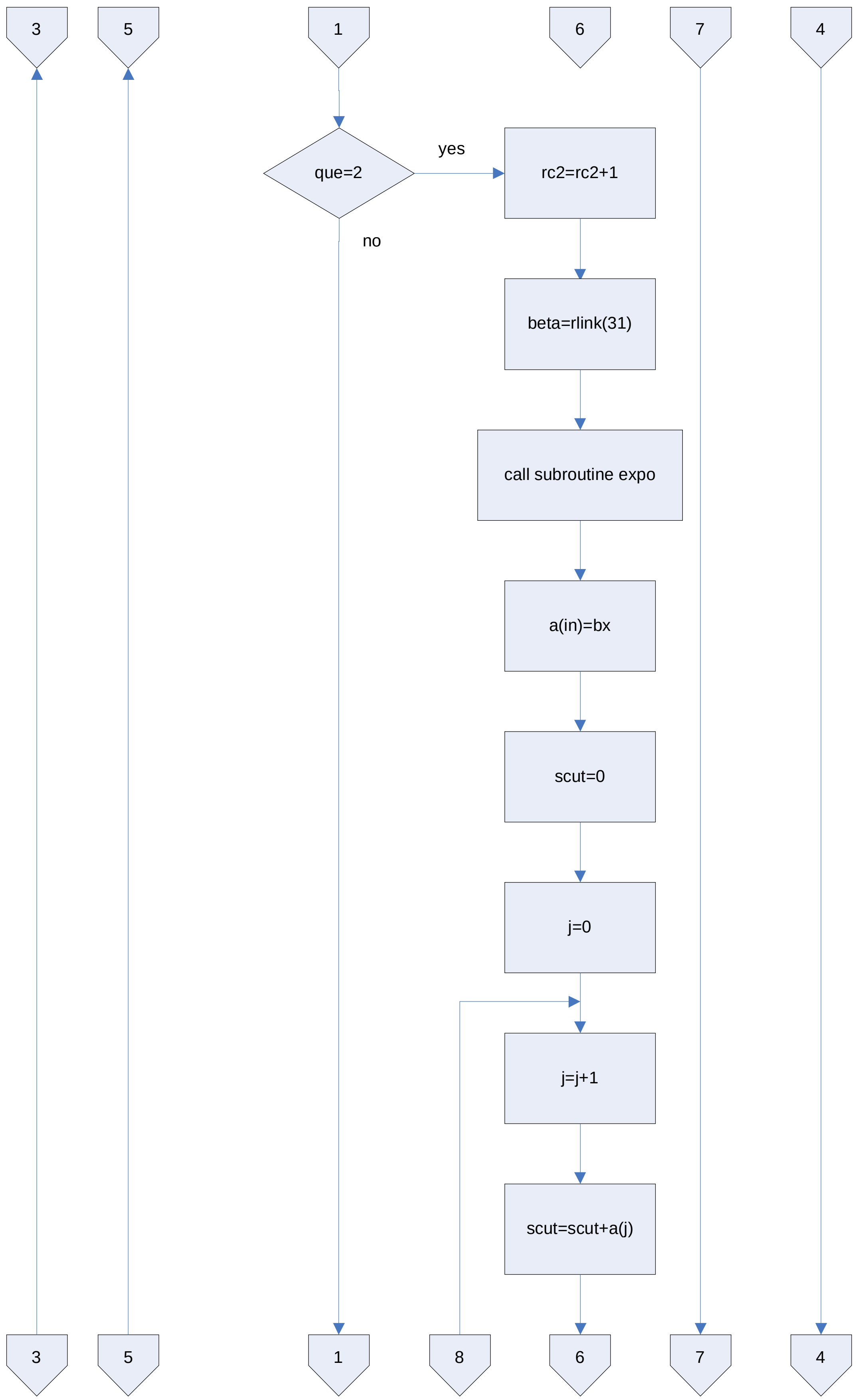

3
8
1
4
k1=k1+1
cut(k1)=0
yes
sumwa(k1)=0
no
cut(k1)>0
yes
sumwa(k1)=rang1-cut(k1)
no
end if
delay=delay+sumwa(k1)
drive=drive+driv(k1)
no
k1=in
yes
drive=drive+add*a(i1+1)
redn=rc1+rc3
3
8
1
4

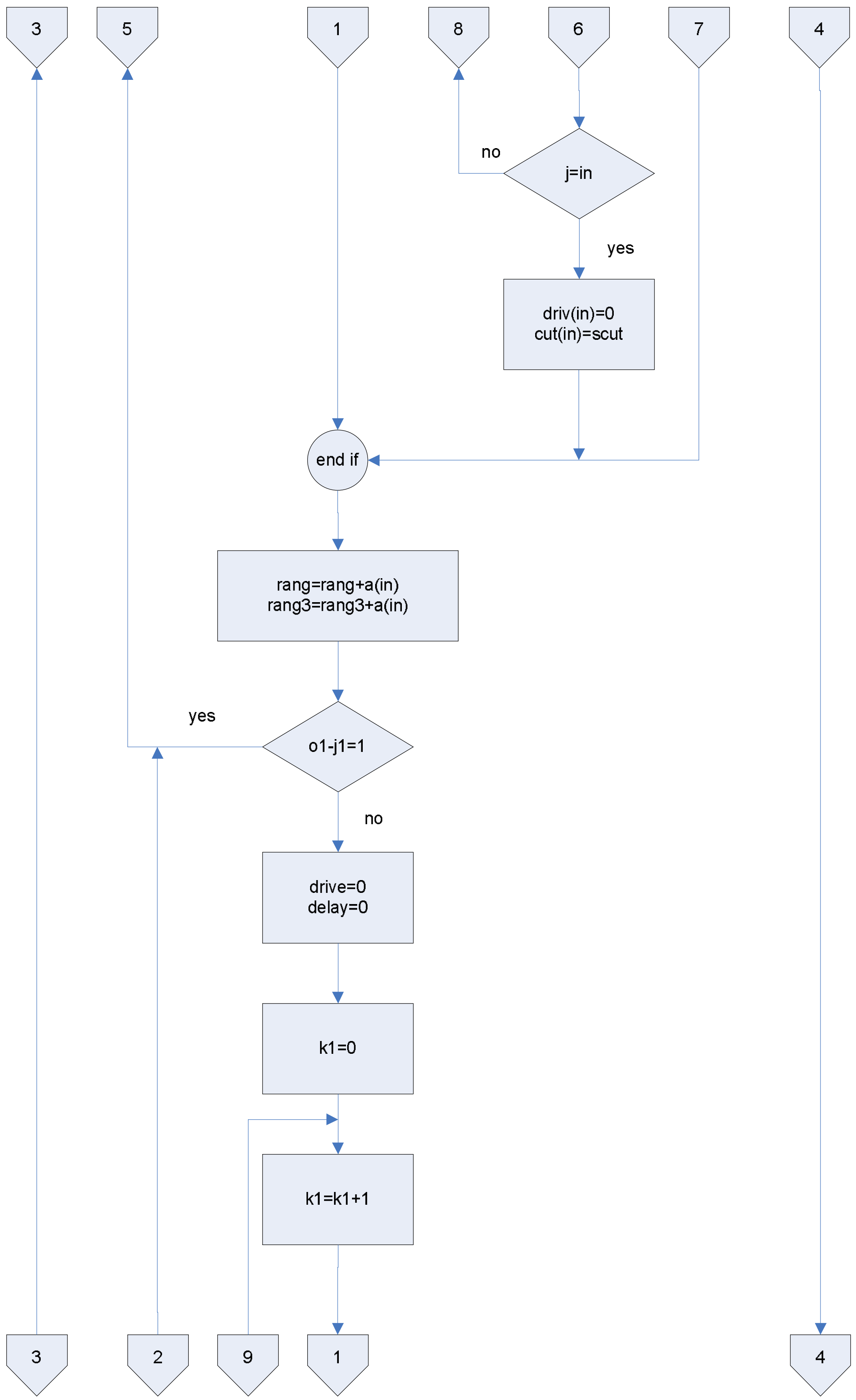

3
8
1
4
delay2=delay/redn
g=4*gn/rang2
red=6*redn/rang2
wait=rang2
drive1=cdrive+drive
delay1=cdelay+delay
redn1=credn+redn
gn1=cgn+gn
cost=(delay1*gn1*(redn+gn))/(100*redn1*gn*drive1)
no
wait>25
yes
ch2(f1)=wait
f1=f1+1
yes
rang>1800
no
gn=rc3
add=rc3
cdrive=drive1
cdelay=delay1
credn=redn1
cgn=gn1
rc2=0
rang3=0
o1=0
in=0
3
1
4

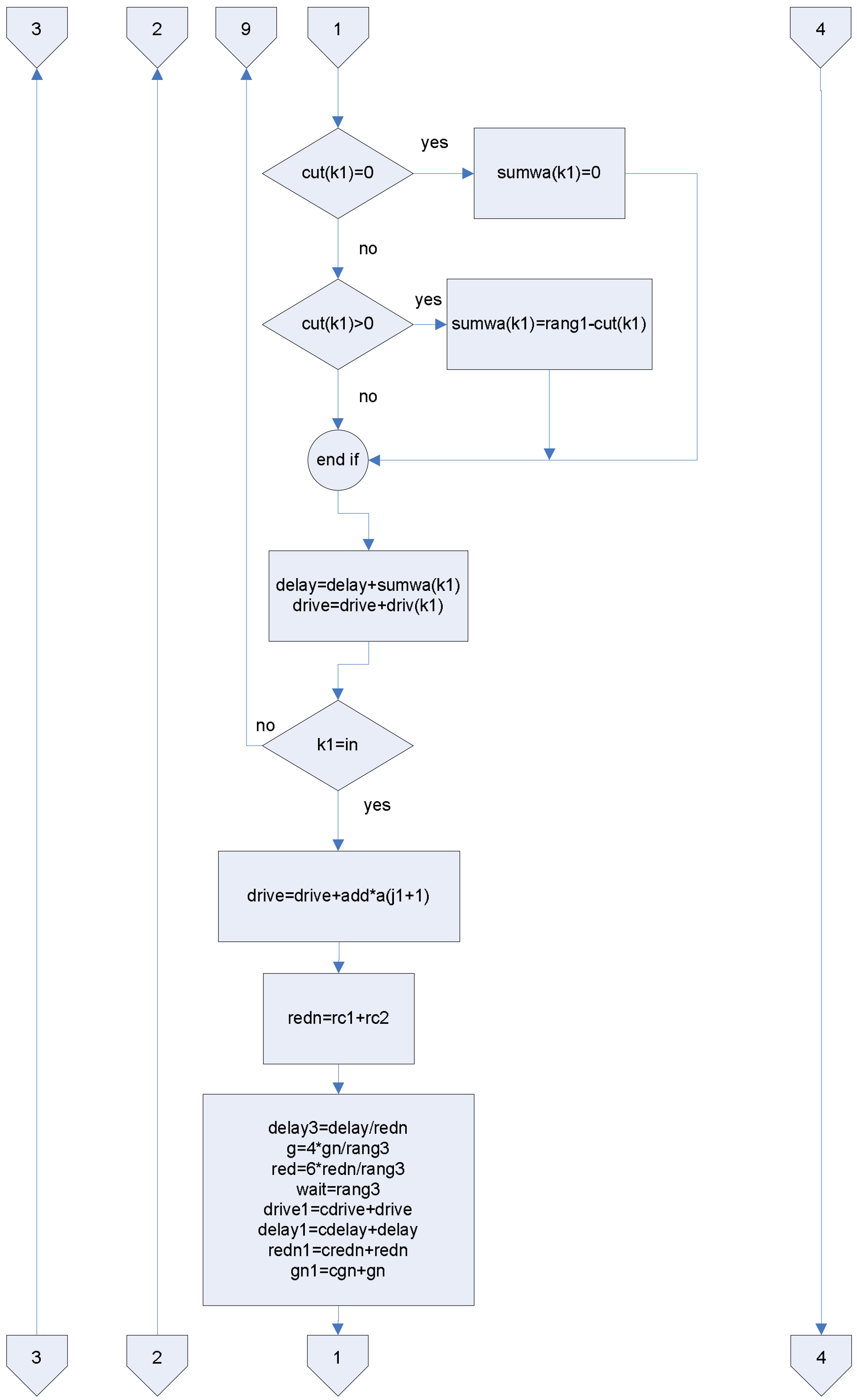

3
1
4
o1=o1+1
in=in+1
p1=rinten(3)/(rinten(3)+rinten(31)+rinten(48))
p2=rinten(31)/(rinten(3)+rinten(31)+rinten(48))
p3=rinten(48)/(rinten(3)+rinten(31)+rinten(48))
p4=0
call subroutine allocate
que=q
que=3
yes
gn=gn+1
no
beta=1/rlink(48)
call subroutine expo
3
5
1
6
4

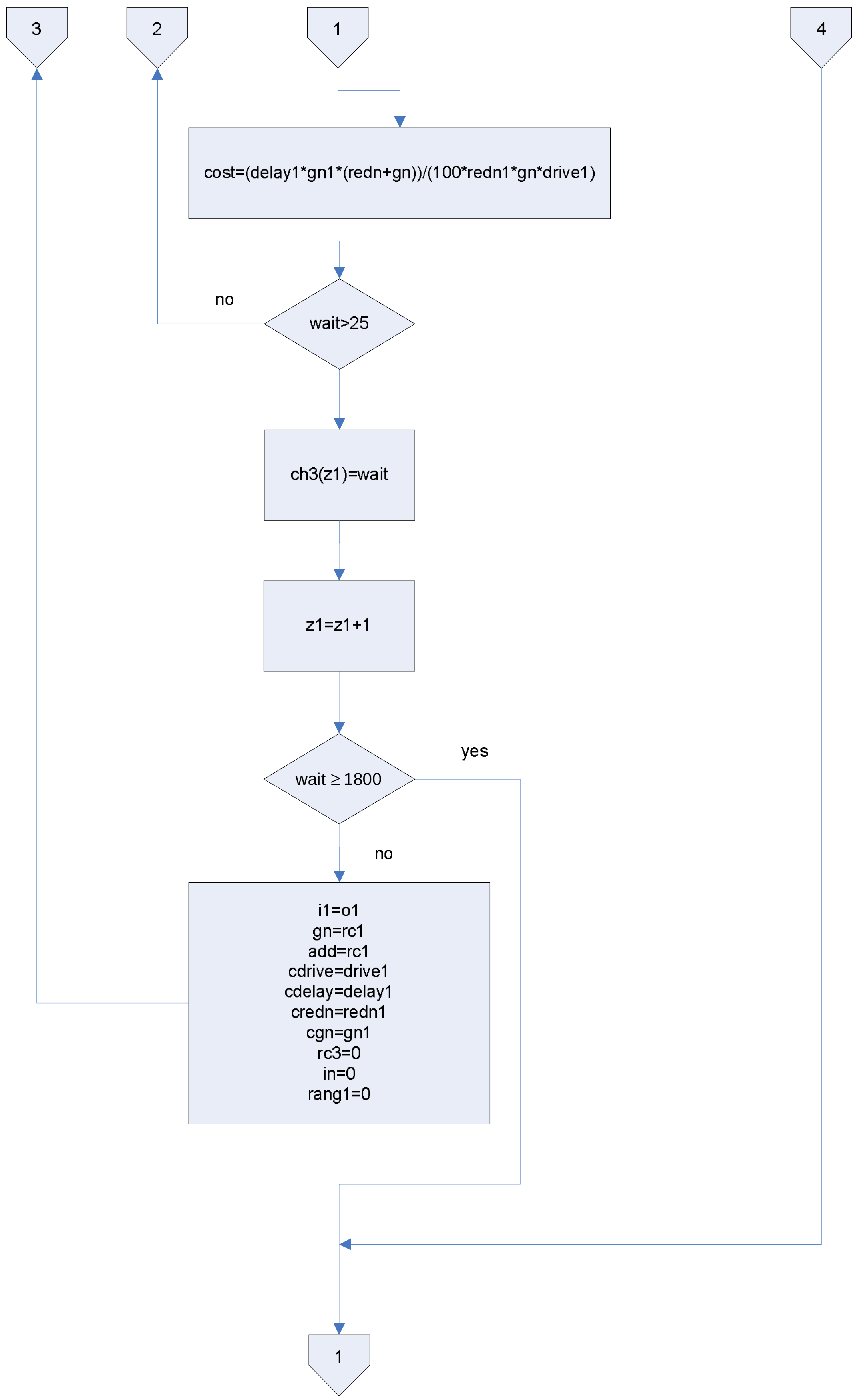

3
5
1
6
4
a(in)=bx
scut=0
j=0
j=j+1
sumg=sumg+a(j)
no
j=in
yes
driv(in)=sumg
cut(in)=0
que=1
yes
rc1=rc1+1
no
3
5
1
6
7
4

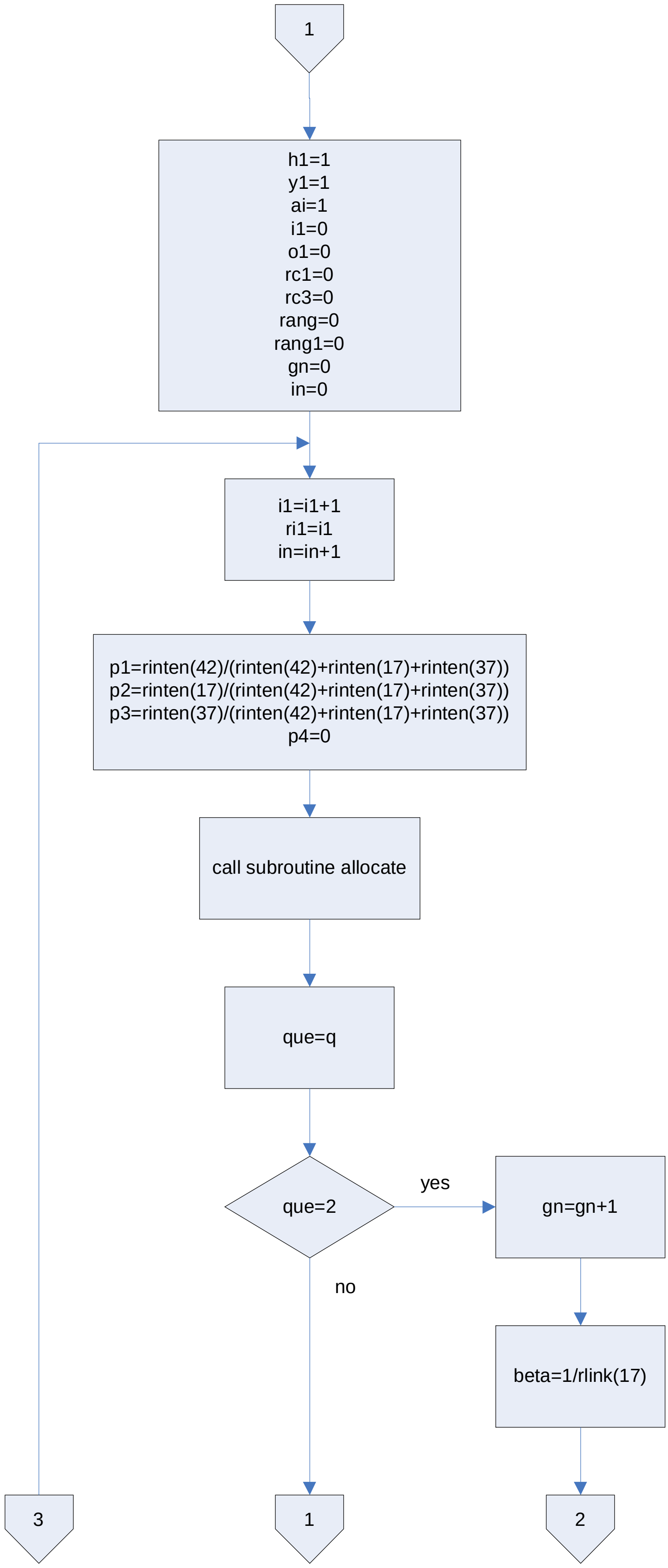

3
5
1
6
7
4
beta=1/rlink(3)
call subroutine expo
a(in)=bx
scut=0
j=0
j=j+1
scut=scut+a(j)
no
j=in
yes
driv(in)=0
cut(in)=scut
3
5
1
7
4

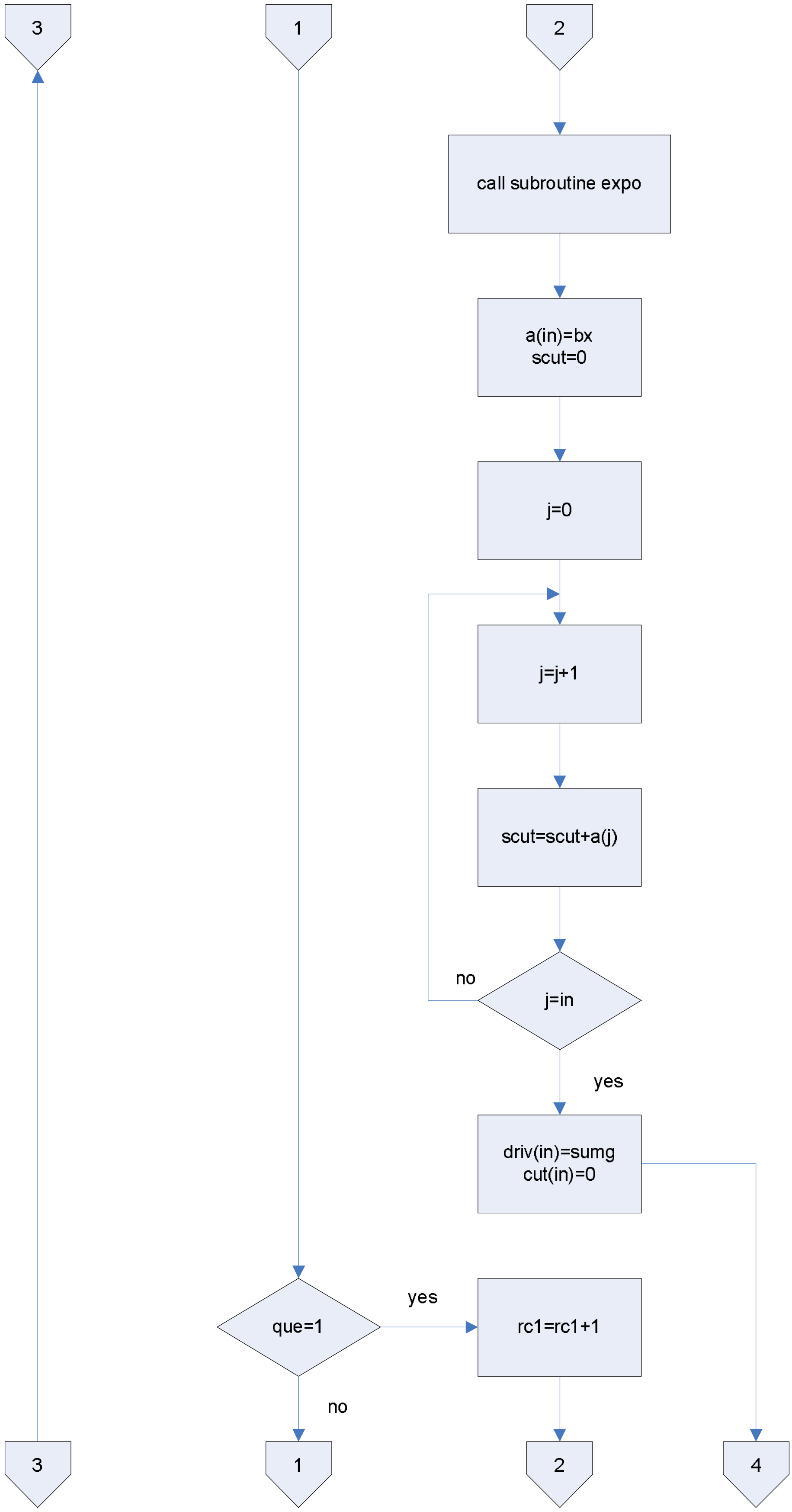

3
5
1
6
7
4
que=2
yes
no
rc2=rc2+1
beta=rlink(31)
call subroutine expo
a(in)=bx
scut=0
j=0
j=j+1
scut=scut+a(j)
3
5
1
8
6
7
4

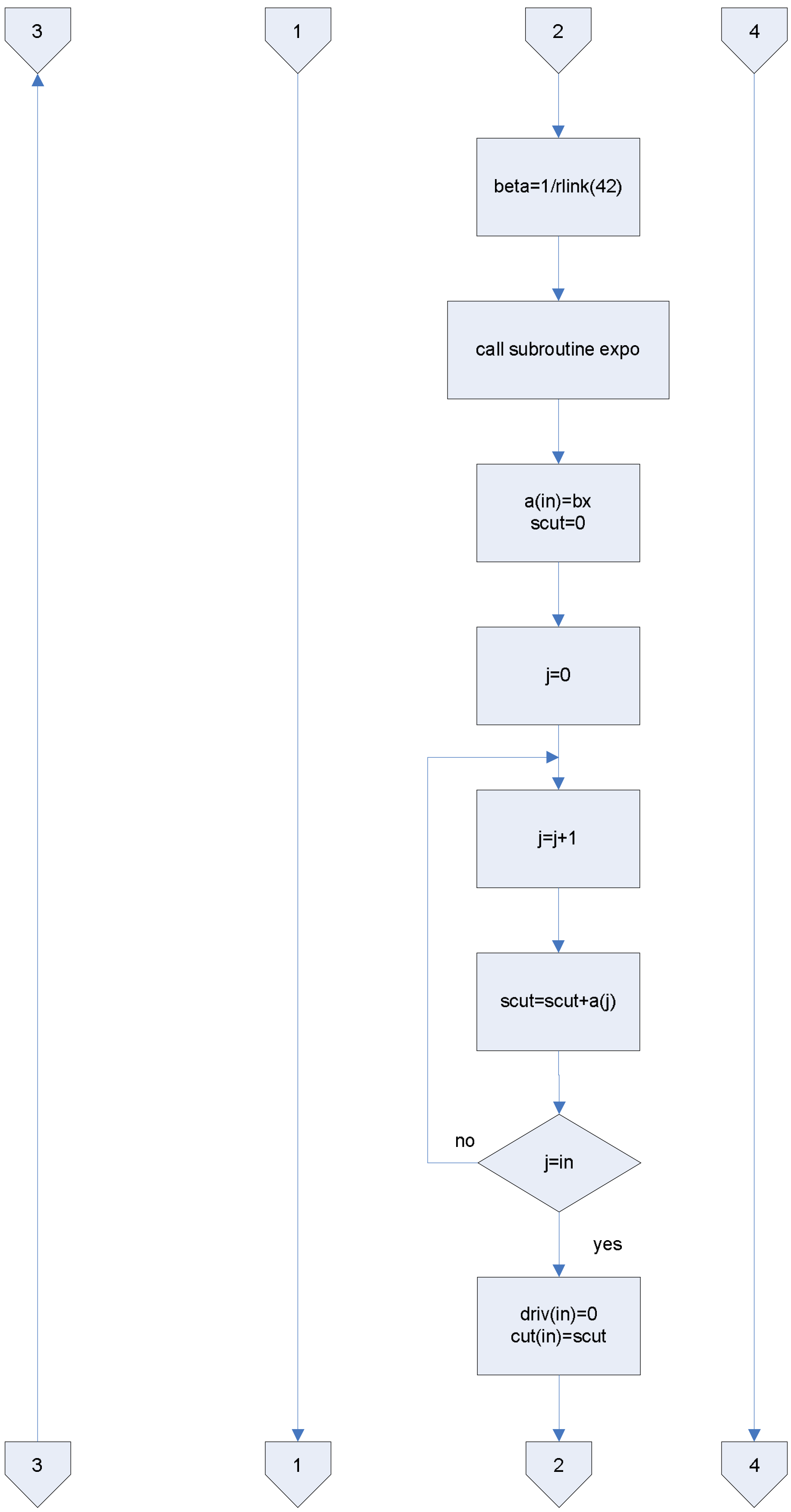

3
5
1
8
6
7
4
no
j=in
yes
driv(in)=0
cut(in)=scut
end if
rang=rang+a(in)
rang3=rang3+a(in)
yes
o1-j1=1
no
drive=0
delay=0
k1=0
k1=k1+1
3
2
9
1
4

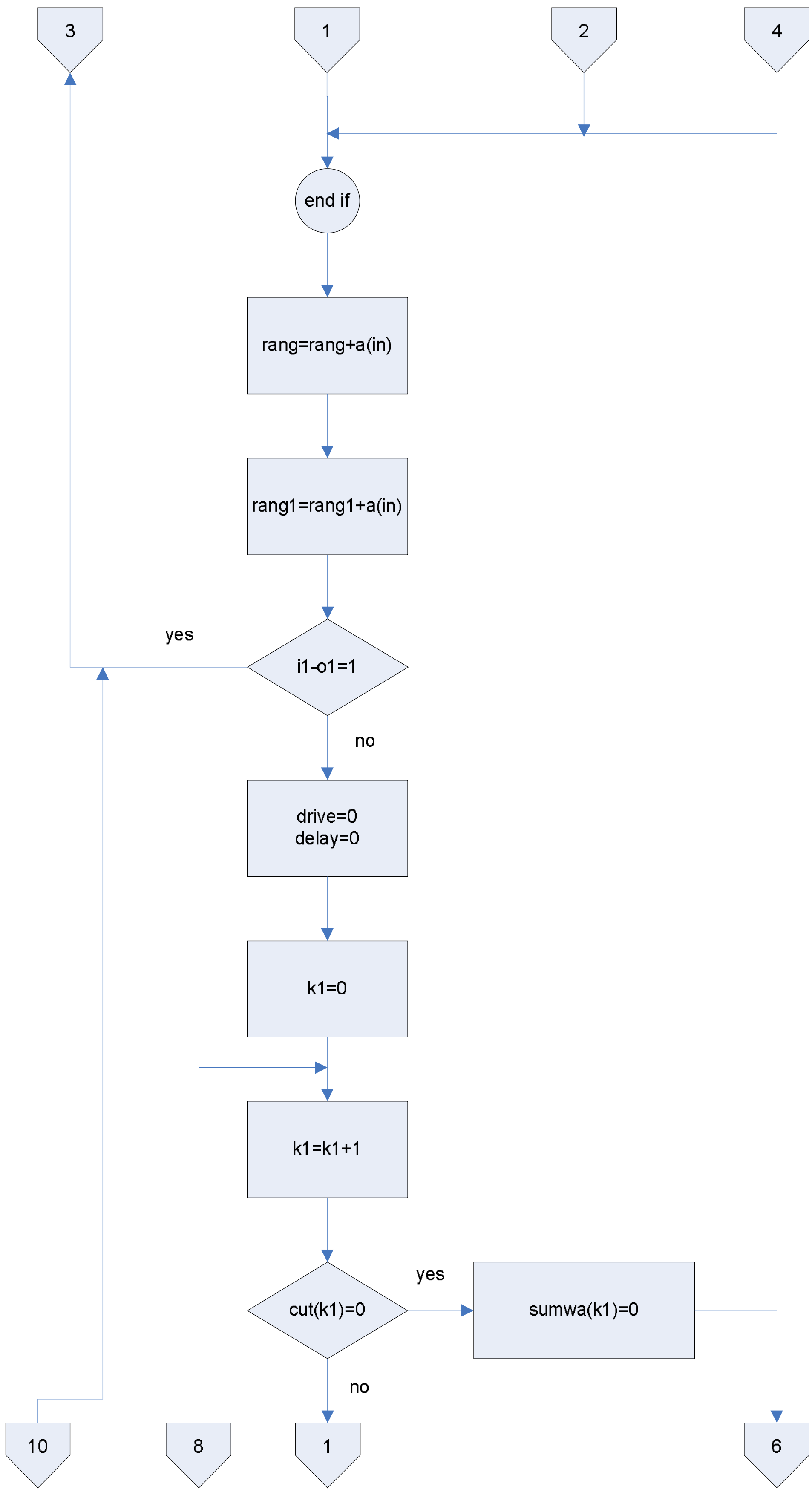

3
2
9
1
4
cut(k1)=0
yes
sumwa(k1)=0
no
cut(k1)>0
yes
sumwa(k1)=rang1-cut(k1)
no
end if
delay=delay+sumwa(k1)
drive=drive+driv(k1)
no
k1=in
yes
drive=drive+add*a(j1+1)
redn=rc1+rc2
delay3=delay/redn
g=4*gn/rang3
red=6*redn/rang3
wait=rang3
drive1=cdrive+drive
delay1=cdelay+delay
redn1=credn+redn
gn1=cgn+gn
3
2
1
4

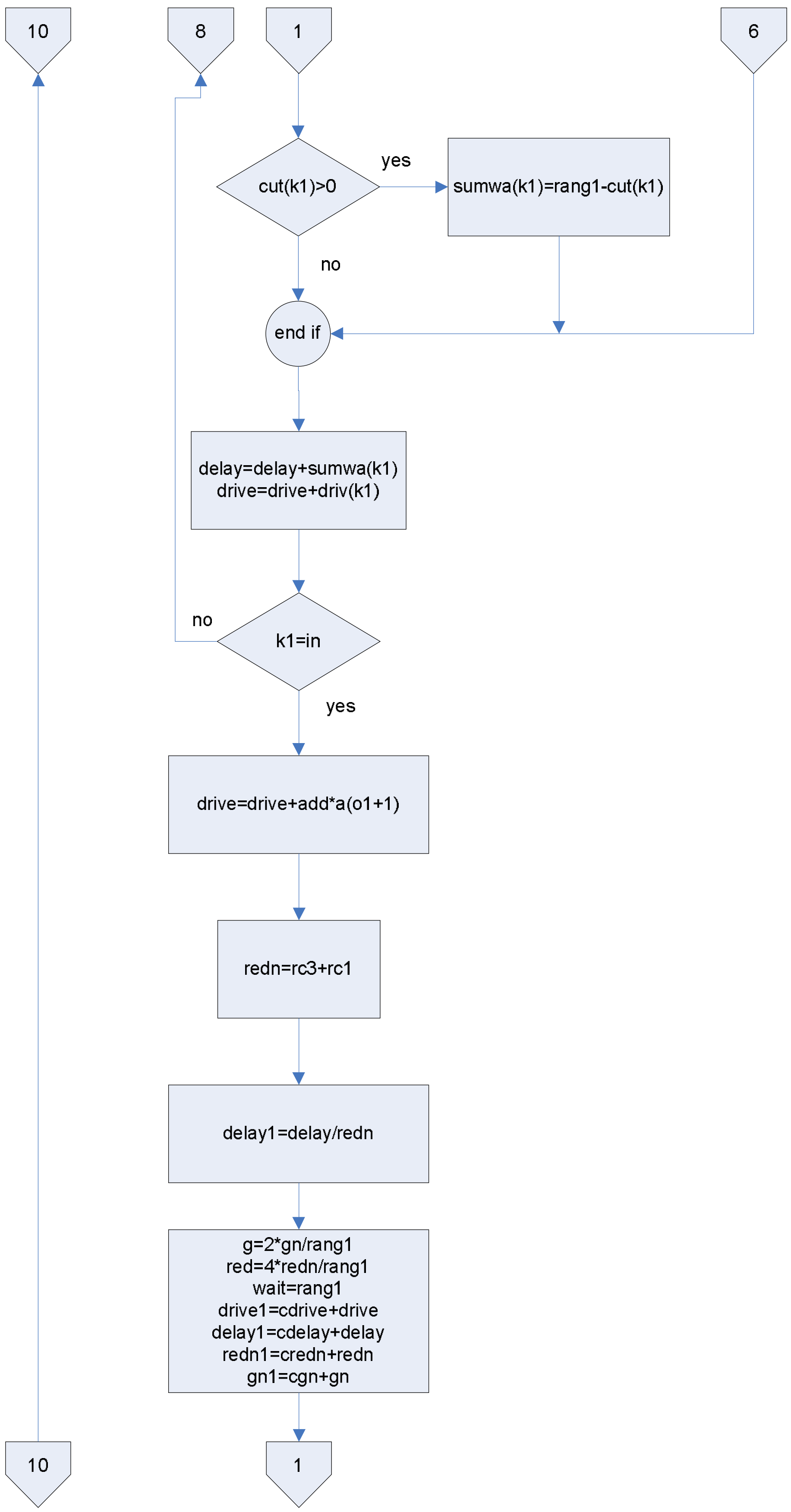

3
2
1
4
cost=(delay1*gn1*(redn+gn))/(100*redn1*gn*drive1)
no
wait>25
ch3(z1)=wait
z1=z1+1
yes
wait ≥ 1800
no
i1=o1
gn=rc1
add=rc1
cdrive=drive1
cdelay=delay1
credn=redn1
cgn=gn1
rc3=0
in=0
rang1=0
1

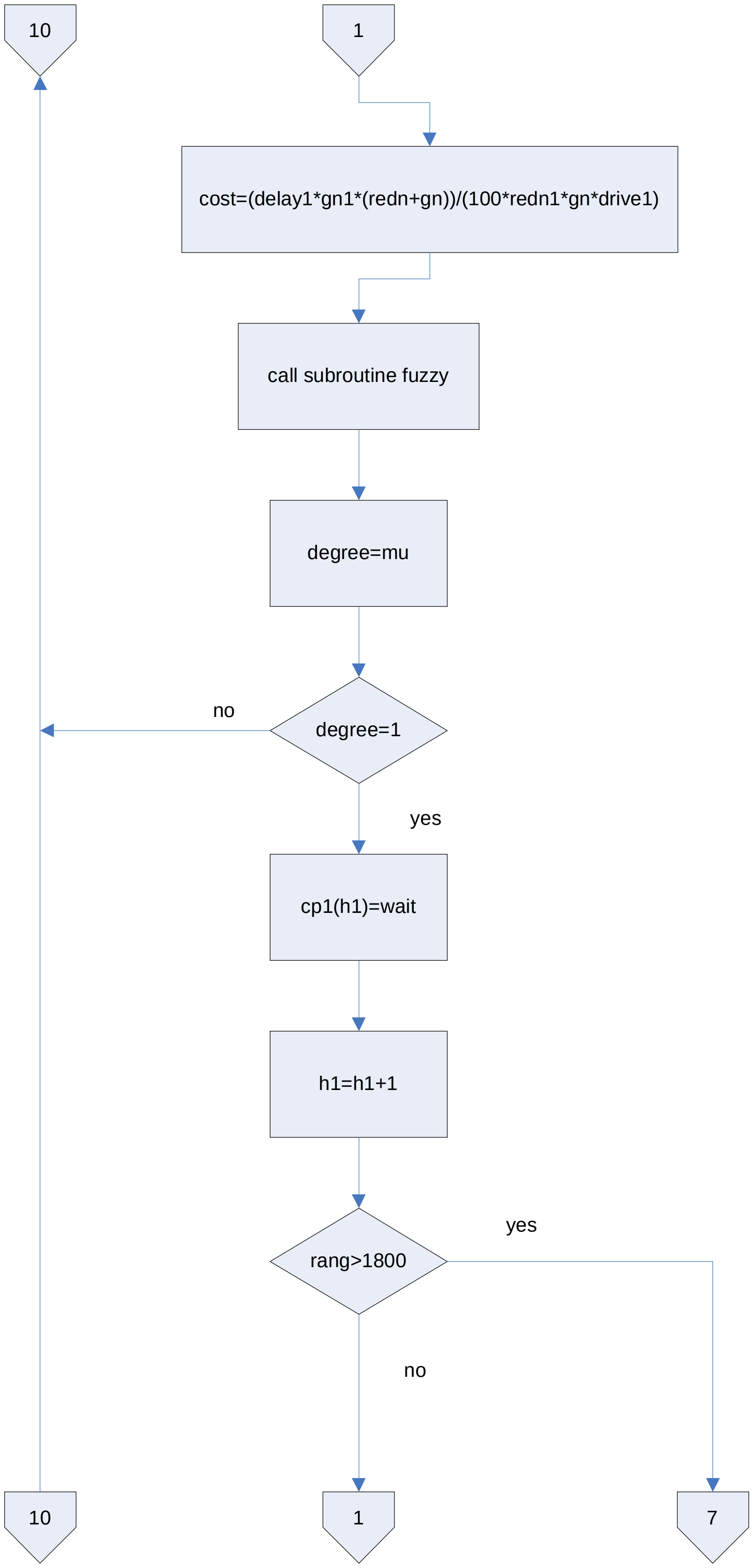

1
h1=1
y1=1
ai=1
i1=0
o1=0
rc1=0
rc3=0
rang=0
rang1=0
gn=0
in=0
i1=i1+1
ri1=i1
in=in+1
p1=rinten(42)/(rinten(42)+rinten(17)+rinten(37))
p2=rinten(17)/(rinten(42)+rinten(17)+rinten(37))
p3=rinten(37)/(rinten(42)+rinten(17)+rinten(37))
p4=0
call subroutine allocate
que=q
que=2
yes
gn=gn+1
no
beta=1/rlink(17)
3
1
2

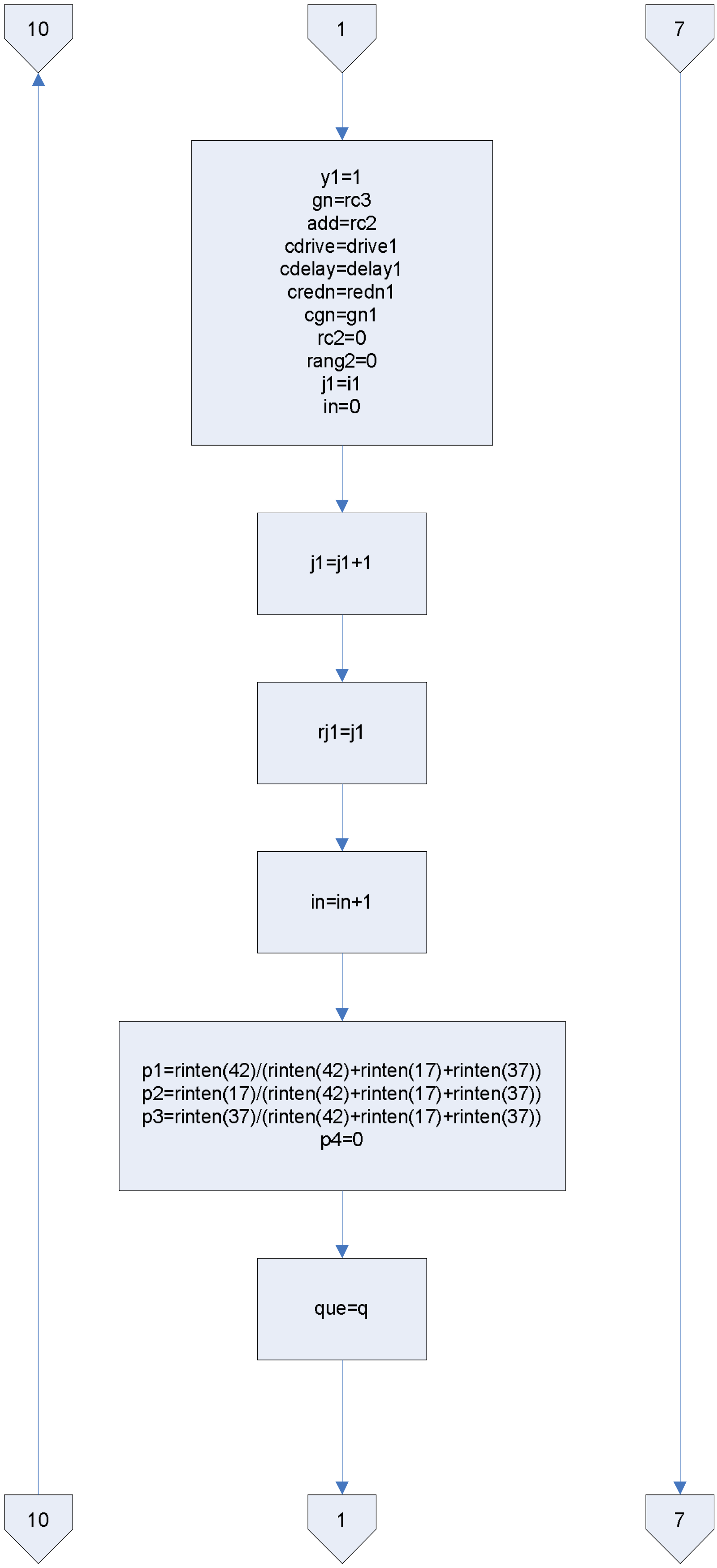

3
1
2
call subroutine expo
a(in)=bx
scut=0
j=0
j=j+1
scut=scut+a(j)
no
j=in
yes
driv(in)=sumg
cut(in)=0
que=1
yes
rc1=rc1+1
no
3
1
2
4

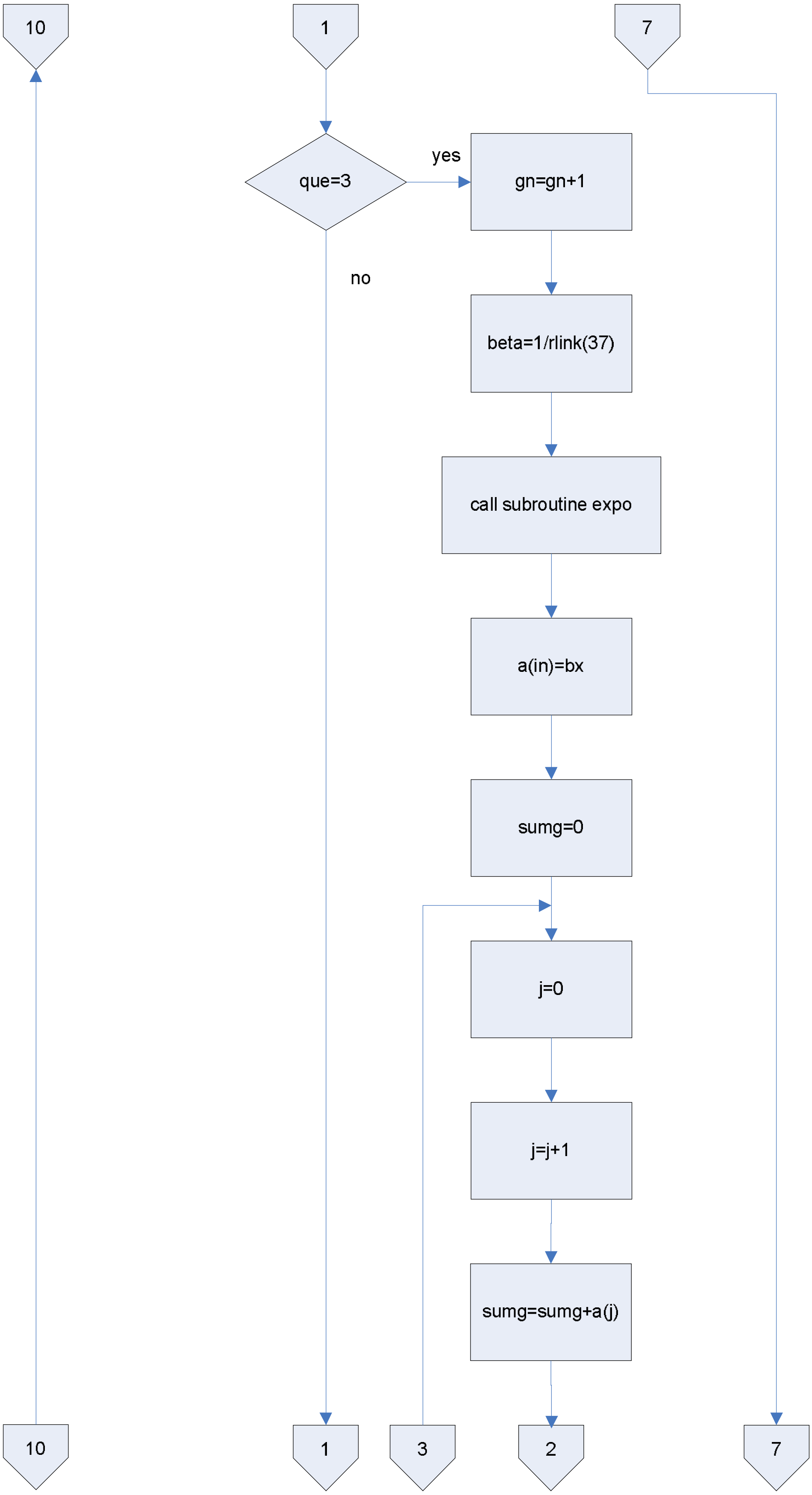

3
1
2
4
beta=1/rlink(42)
call subroutine expo
a(in)=bx
scut=0
j=0
j=j+1
scut=scut+a(j)
no
j=in
yes
driv(in)=0
cut(in)=scut
3
1
2
4


3
1
2
4
end if
rang=rang+a(in)
rang1=rang1+a(in)
yes
i1-o1=1
no
drive=0
delay=0
k1=0
k1=k1+1
yes
cut(k1)=0
sumwa(k1)=0
no
10
8
1
6

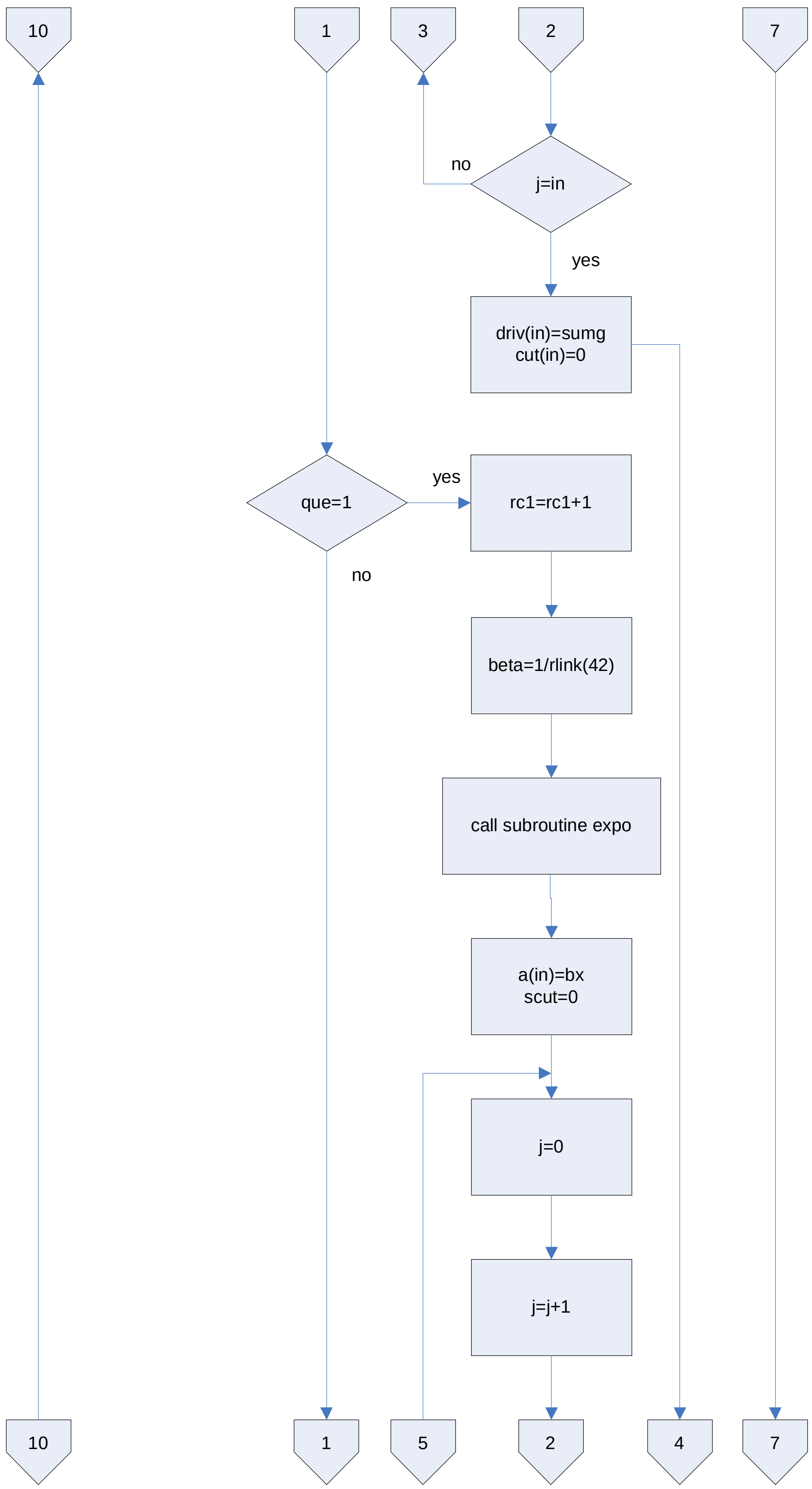

10
8
1
6
cut(k1)>0
yes
sumwa(k1)=rang1-cut(k1)
no
end if
delay=delay+sumwa(k1)
drive=drive+driv(k1)
no
k1=in
yes
drive=drive+add*a(o1+1)
redn=rc3+rc1
delay1=delay/redn
g=2*gn/rang1
red=4*redn/rang1
wait=rang1
drive1=cdrive+drive
delay1=cdelay+delay
redn1=credn+redn
gn1=cgn+gn
10
1

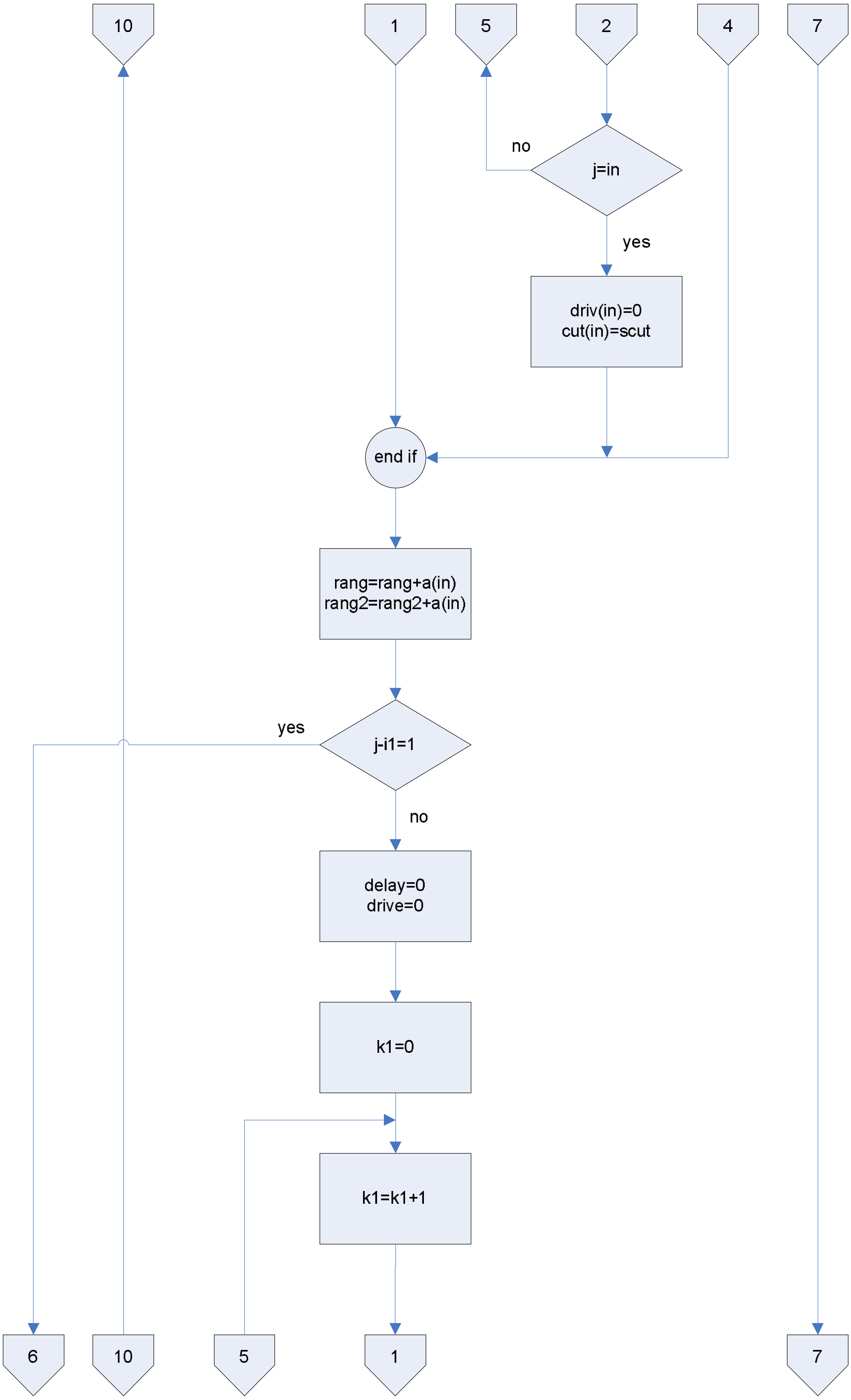

10
1
cost=(delay1*gn1*(redn+gn))/(100*redn1*gn*drive1)
call subroutine fuzzy
degree=mu
no
degree=1
yes
cp1(h1)=wait
h1=h1+1
yes
rang>1800
no
10
1
7

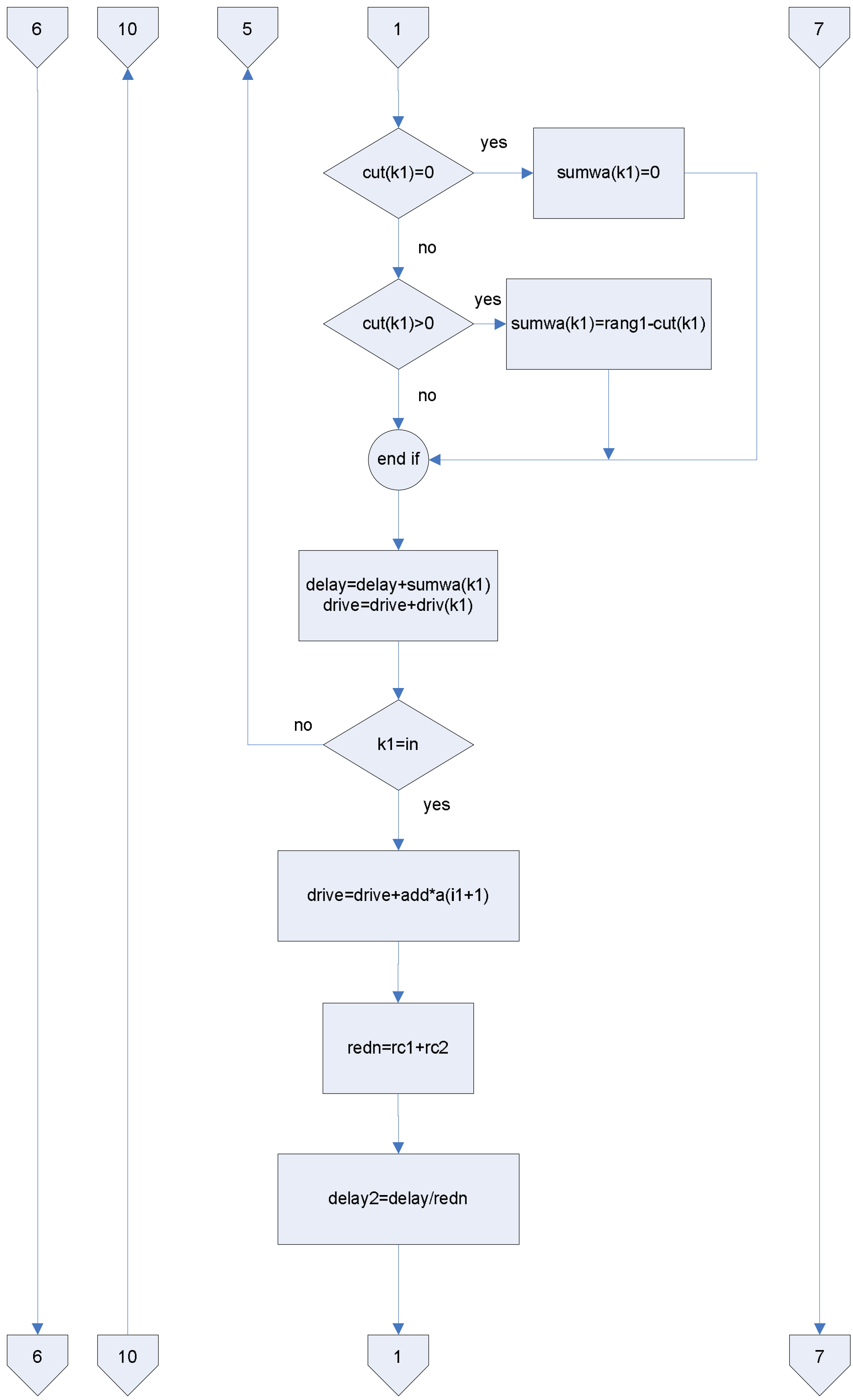

10
1
7
y1=1
gn=rc3
add=rc2
cdrive=drive1
cdelay=delay1
credn=redn1
cgn=gn1
rc2=0
rang2=0
j1=i1
in=0
j1=j1+1
rj1=j1
in=in+1
p1=rinten(42)/(rinten(42)+rinten(17)+rinten(37))
p2=rinten(17)/(rinten(42)+rinten(17)+rinten(37))
p3=rinten(37)/(rinten(42)+rinten(17)+rinten(37))
p4=0
que=q
10
1
7

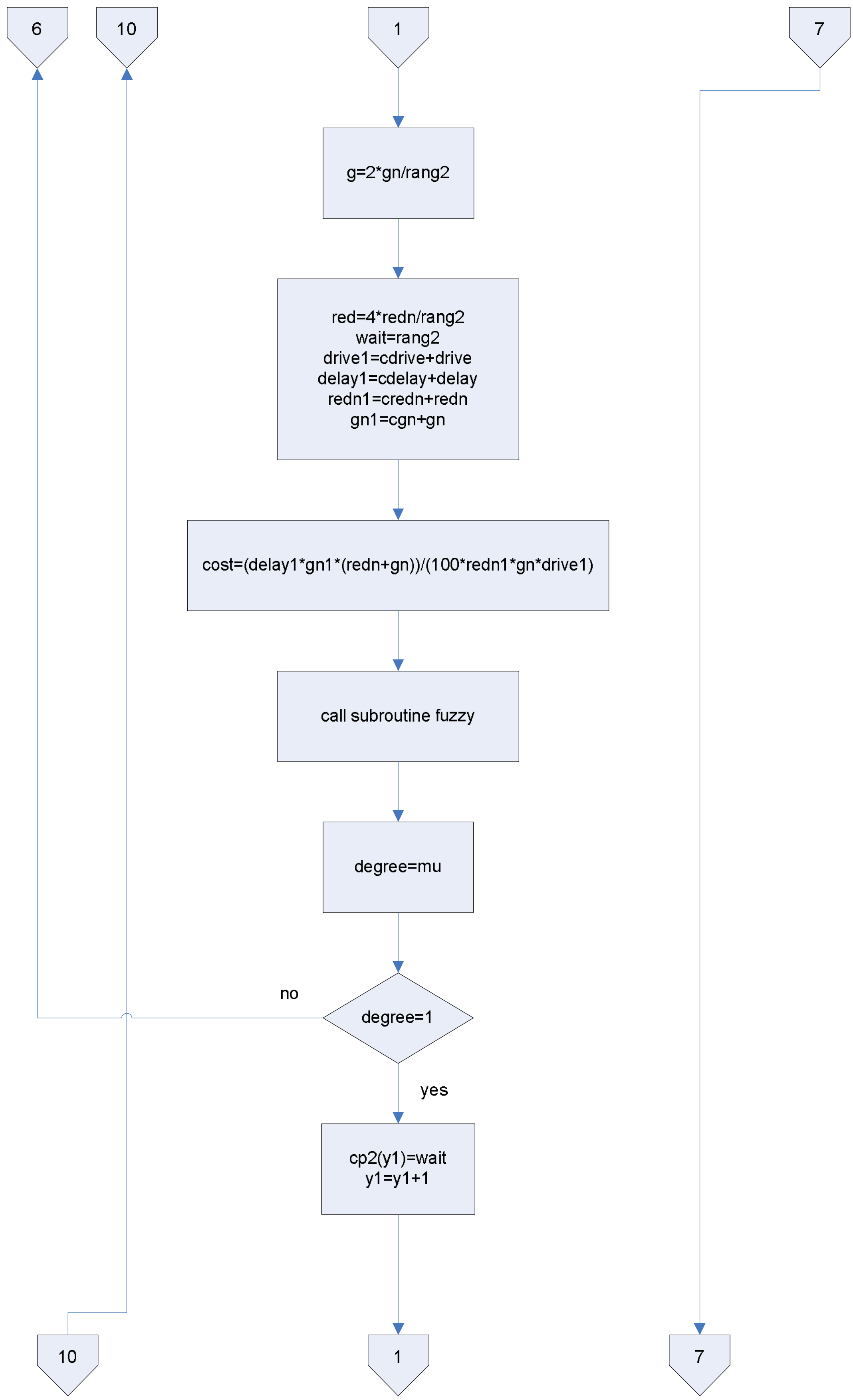


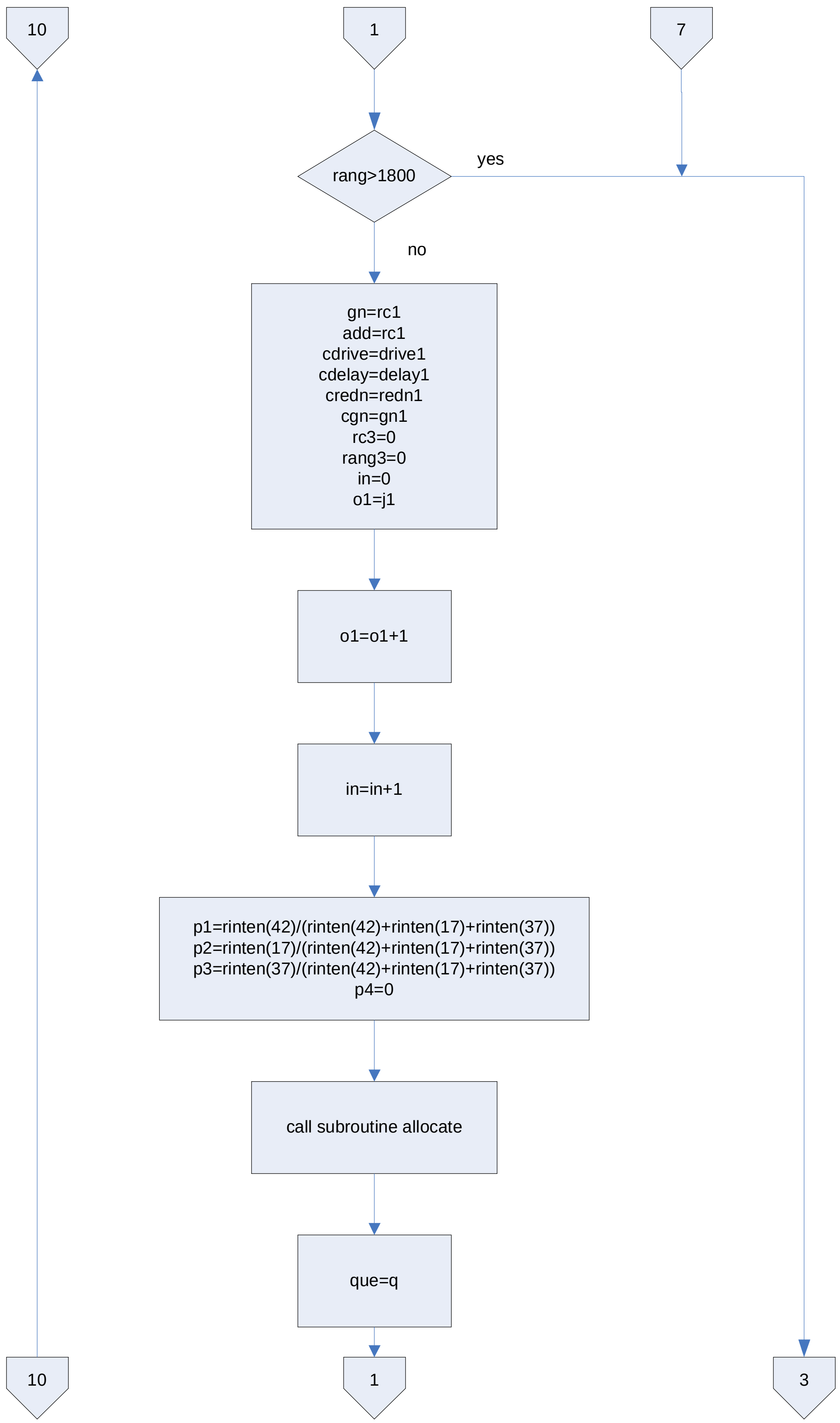
10
1
7
que=3
yes
gn=gn+1
no
beta=1/rlink(37)
call subroutine expo
a(in)=bx
sumg=0
j=0
j=j+1
sumg=sumg+a(j)
10
1
3
2
7

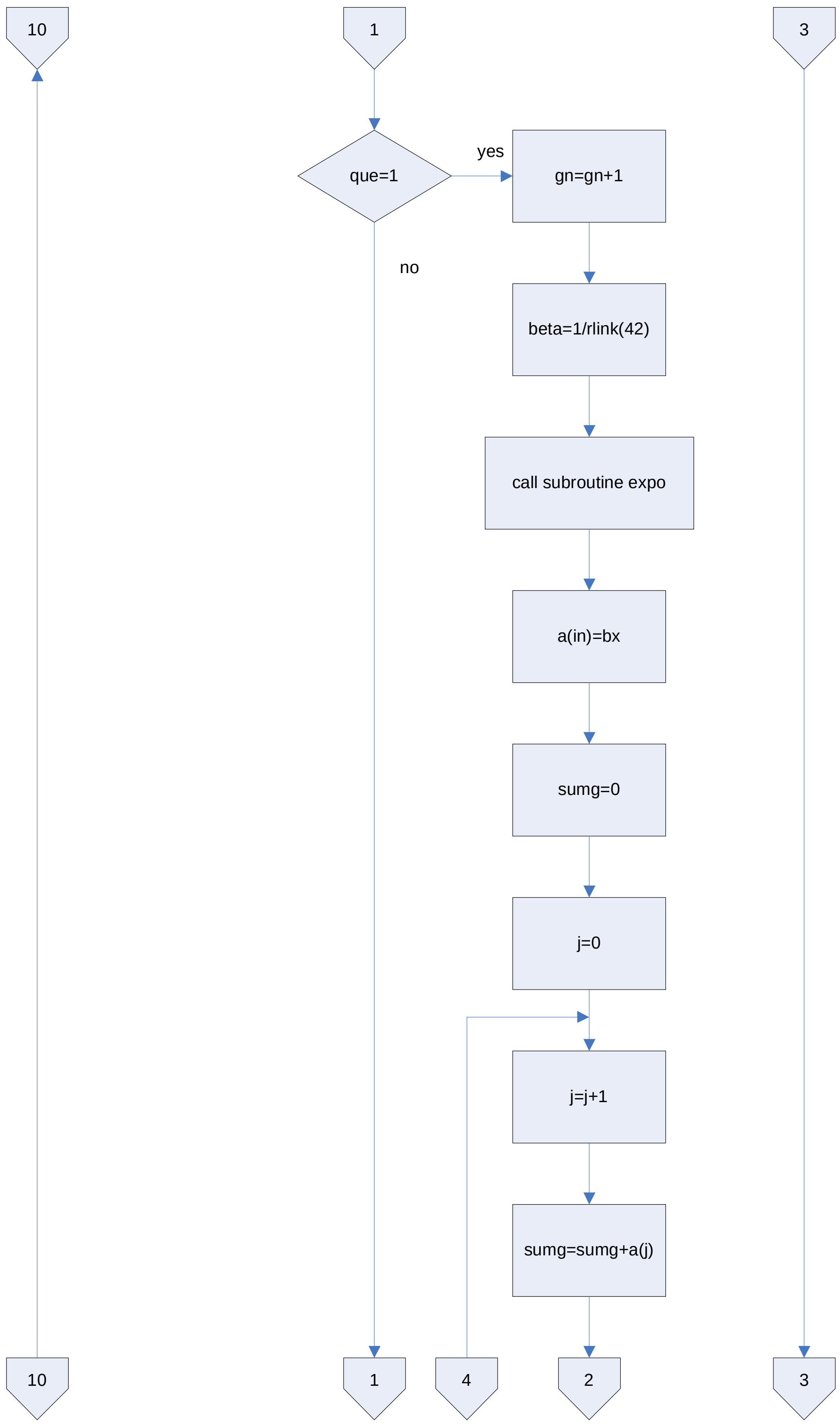

10
1
3
2
7
no
j=in
yes
driv(in)=sumg
cut(in)=0
que=1
yes
rc1=rc1+1
no
beta=1/rlink(42)
call subroutine expo
a(in)=bx
scut=0
j=0
j=j+1
10
1
5
2
4
7

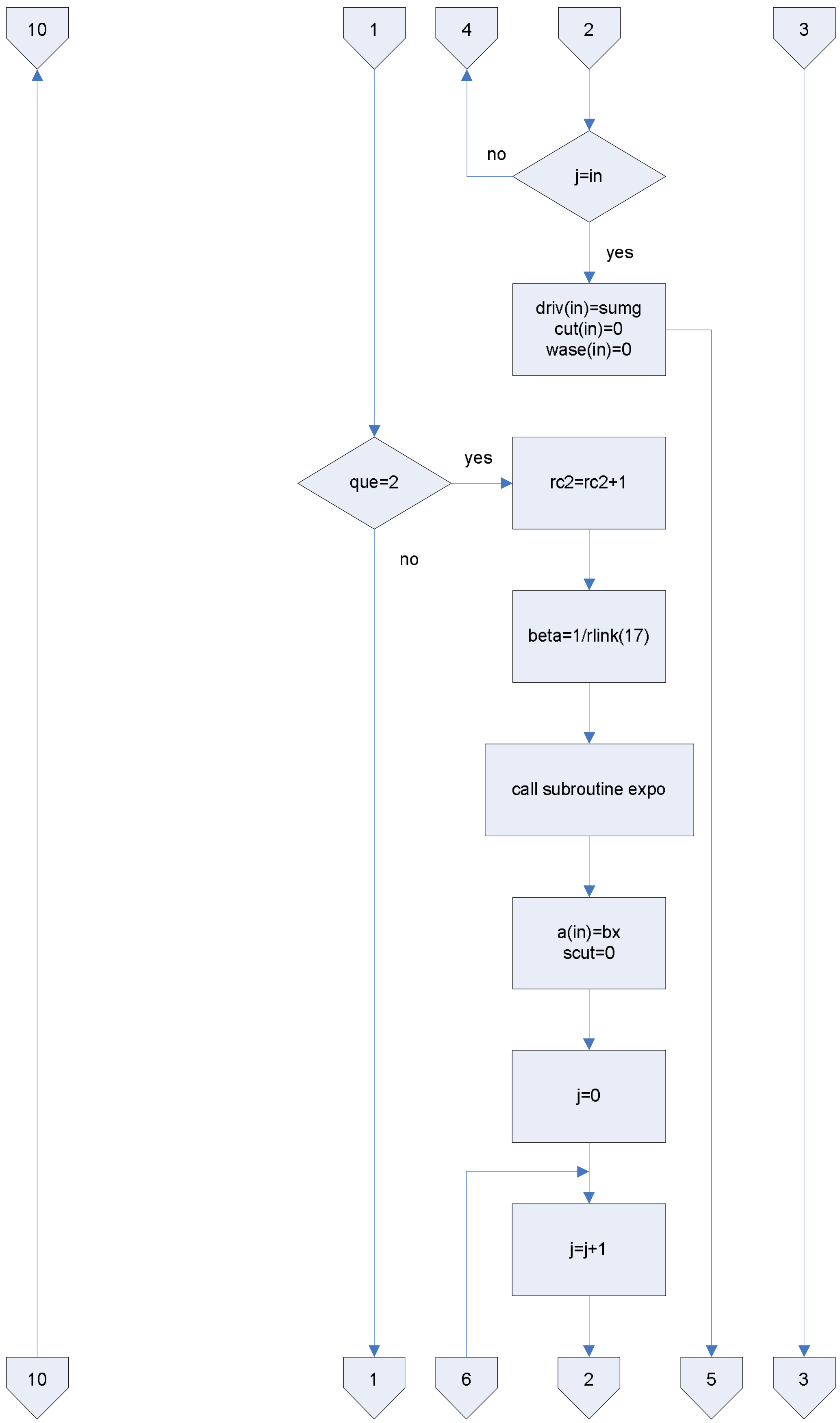

10
1
5
2
4
7
no
j=in
yes
driv(in)=0
cut(in)=scut
end if
rang=rang+a(in)
rang2=rang2+a(in)
yes
j-i1=1
no
delay=0
drive=0
k1=0
k1=k1+1
6
10
5
1
7


6
10
5
1
7
cut(k1)=0
yes
sumwa(k1)=0
no
cut(k1)>0
yes
sumwa(k1)=rang1-cut(k1)
no
end if
delay=delay+sumwa(k1)
drive=drive+driv(k1)
k1=in
no
yes
drive=drive+add*a(i1+1)
redn=rc1+rc2
delay2=delay/redn
6
10
1
7

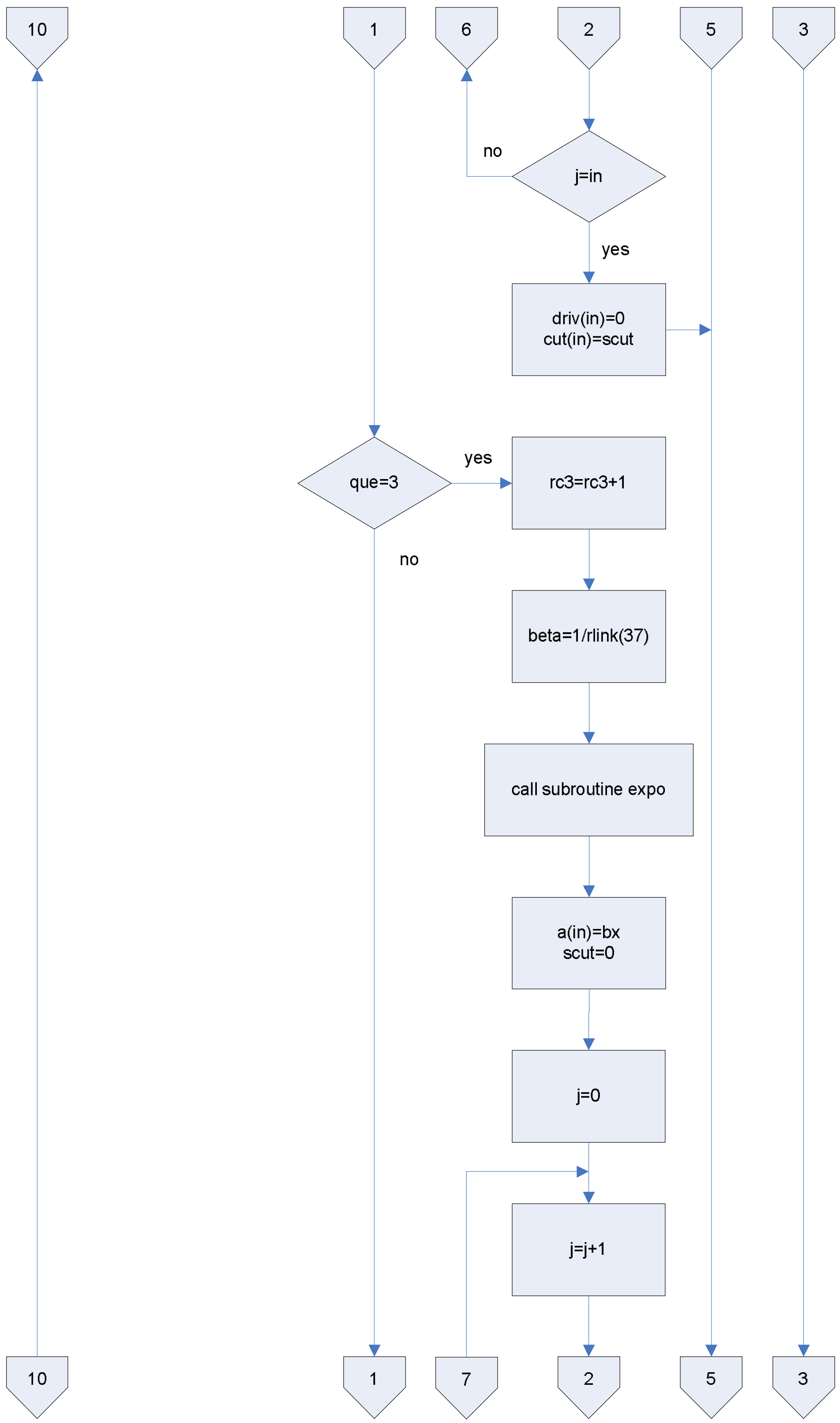

6
10
1
7
g=2*gn/rang2
red=4*redn/rang2
wait=rang2
drive1=cdrive+drive
delay1=cdelay+delay
redn1=credn+redn
gn1=cgn+gn
cost=(delay1*gn1*(redn+gn))/(100*redn1*gn*drive1)
call subroutine fuzzy
degree=mu
no
degree=1
yes
cp2(y1)=wait
y1=y1+1
10
1
7

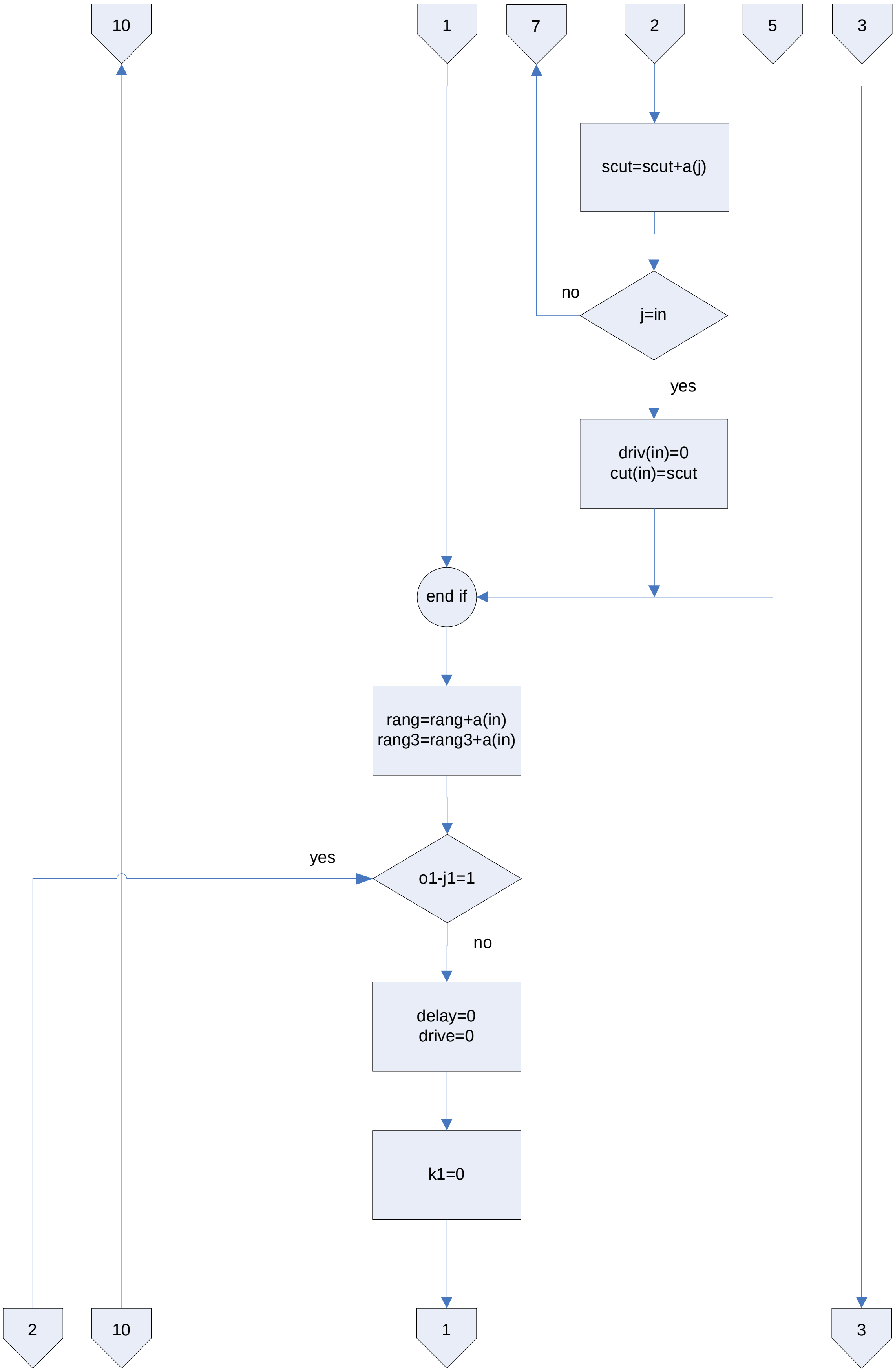

10
1
7
rang>1800
yes
no
gn=rc1
add=rc1
cdrive=drive1
cdelay=delay1
credn=redn1
cgn=gn1
rc3=0
rang3=0
in=0
o1=j1
o1=o1+1
in=in+1
p1=rinten(42)/(rinten(42)+rinten(17)+rinten(37))
p2=rinten(17)/(rinten(42)+rinten(17)+rinten(37))
p3=rinten(37)/(rinten(42)+rinten(17)+rinten(37))
p4=0
call subroutine allocate
que=q
10
1
3

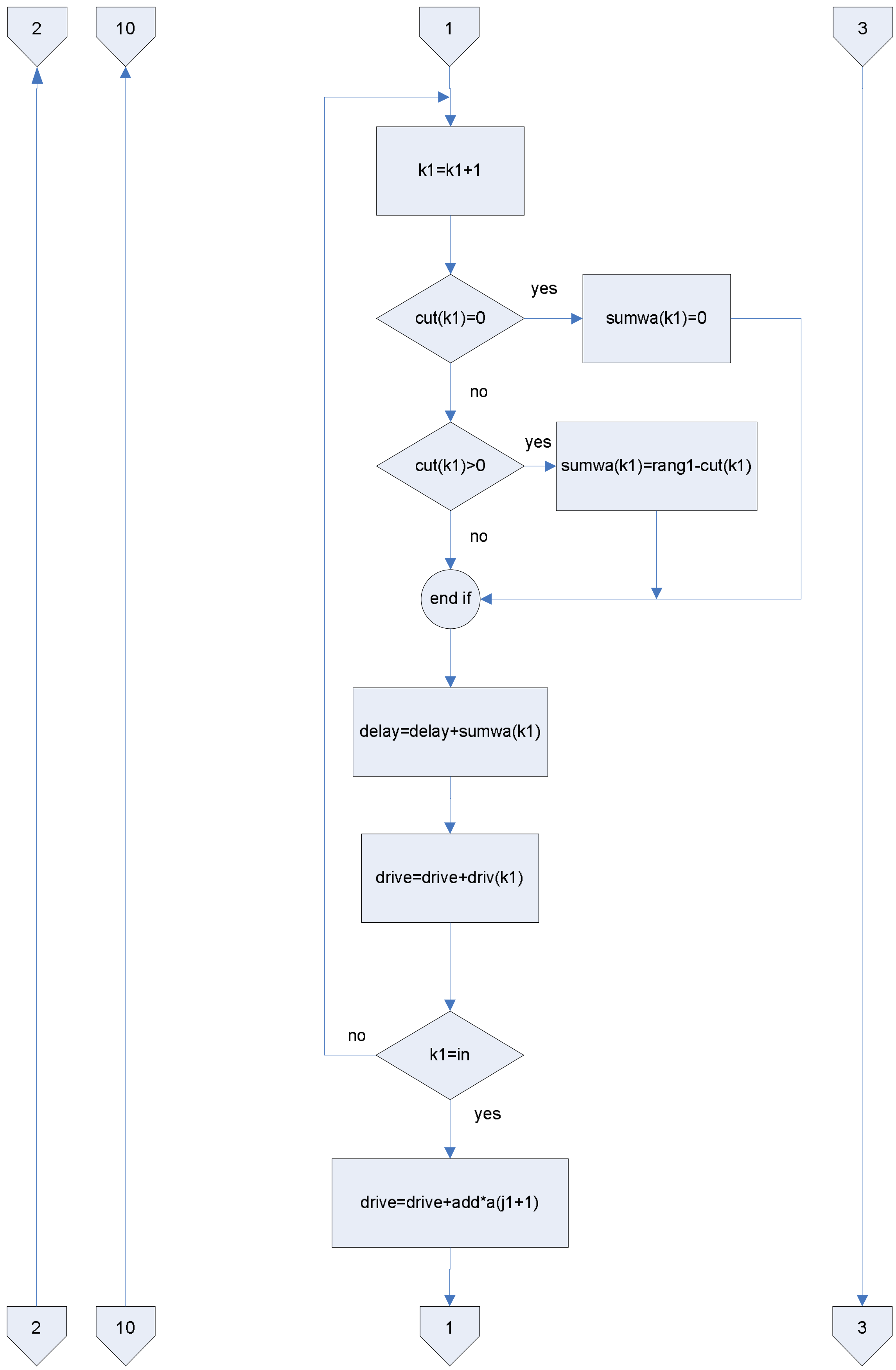

10
1
3
que=1
yes
gn=gn+1
no
beta=1/rlink(42)
call subroutine expo
a(in)=bx
sumg=0
j=0
j=j+1
sumg=sumg+a(j)
10
1
4
2
3

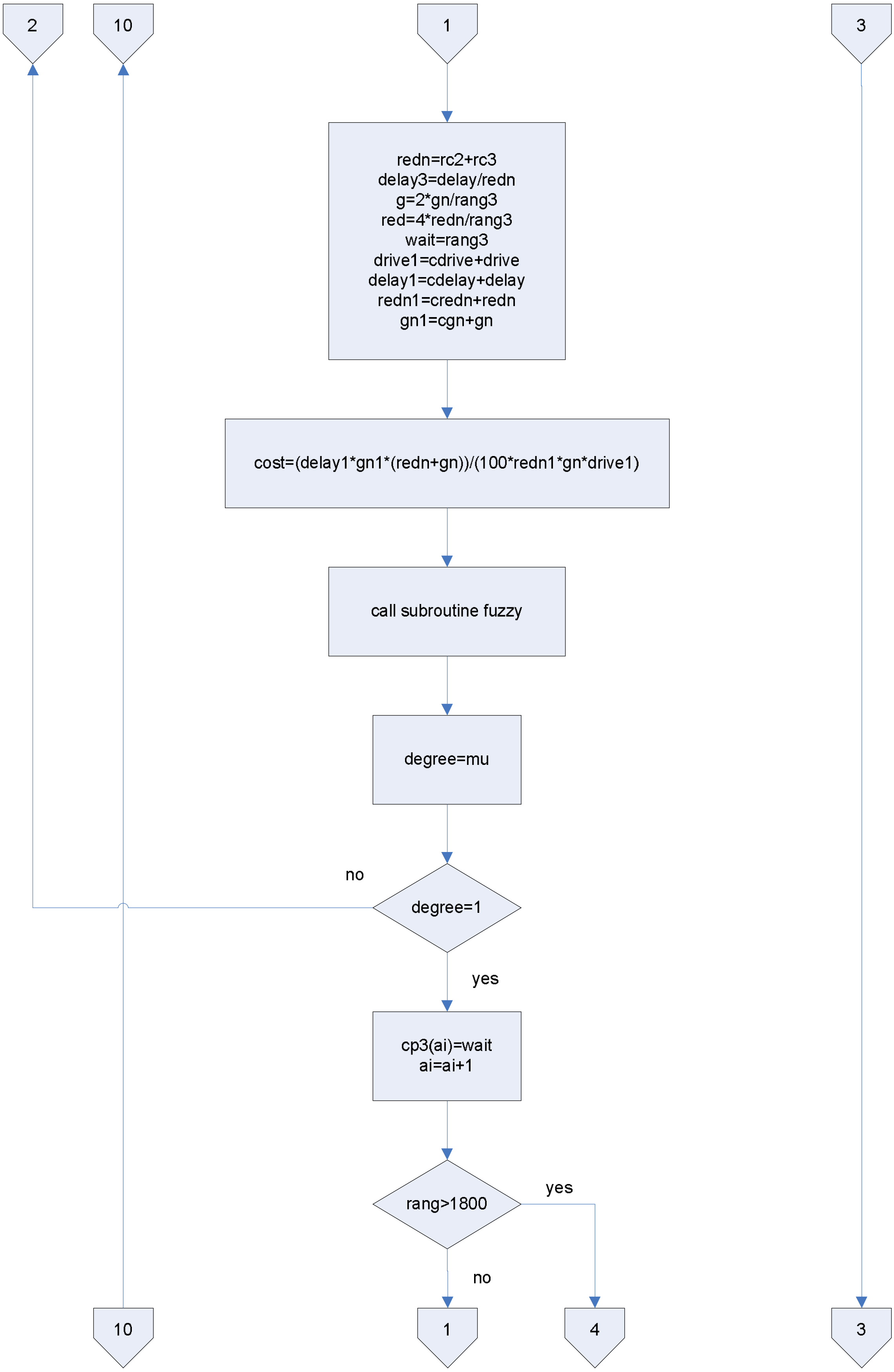


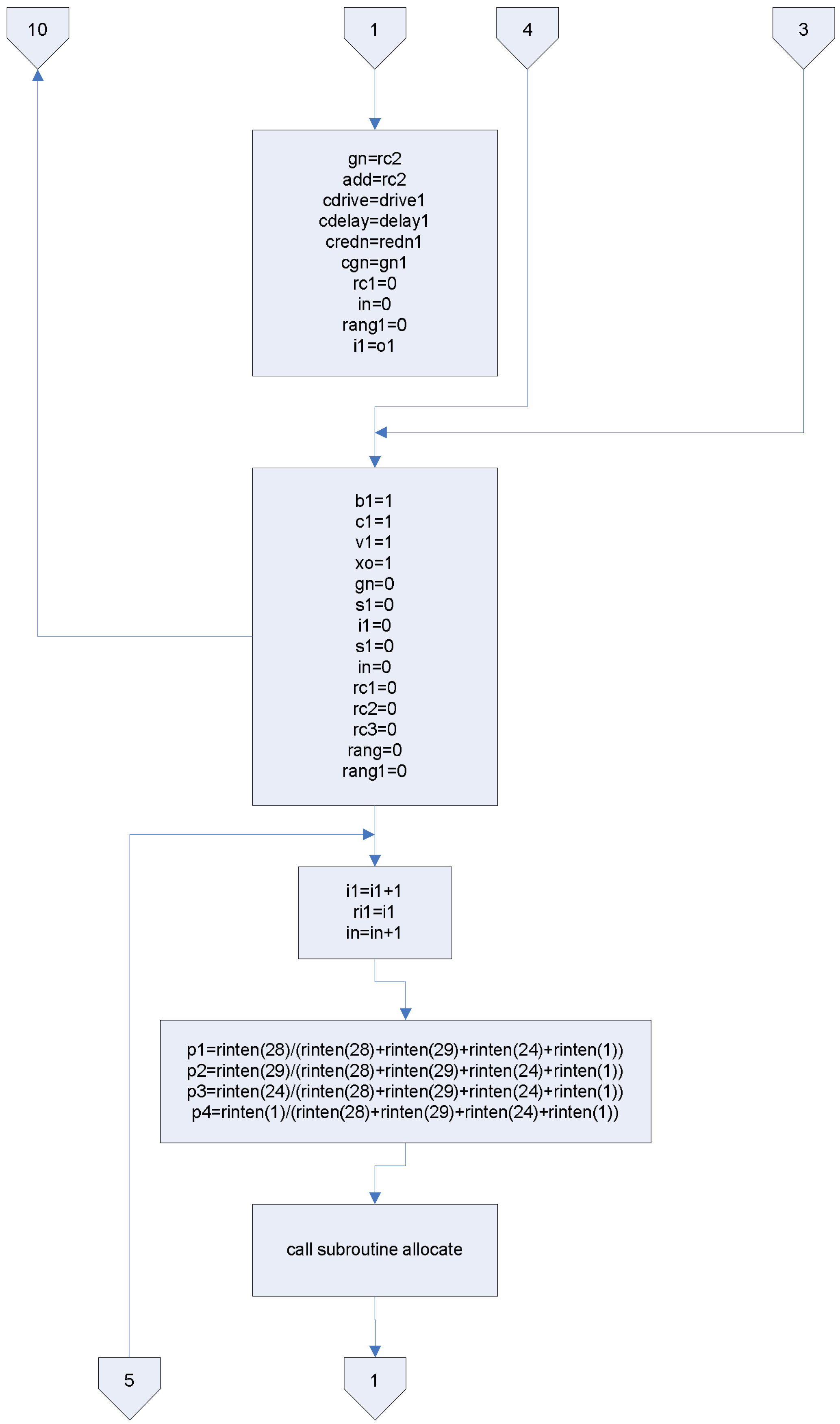
10
1
4
2
3
no
j=in
yes
driv(in)=sumg
cut(in)=0
wase(in)=0
que=2
yes
rc2=rc2+1
no
beta=1/rlink(17)
call subroutine expo
a(in)=bx
scut=0
j=0
j=j+1
10
1
6
2
5
3

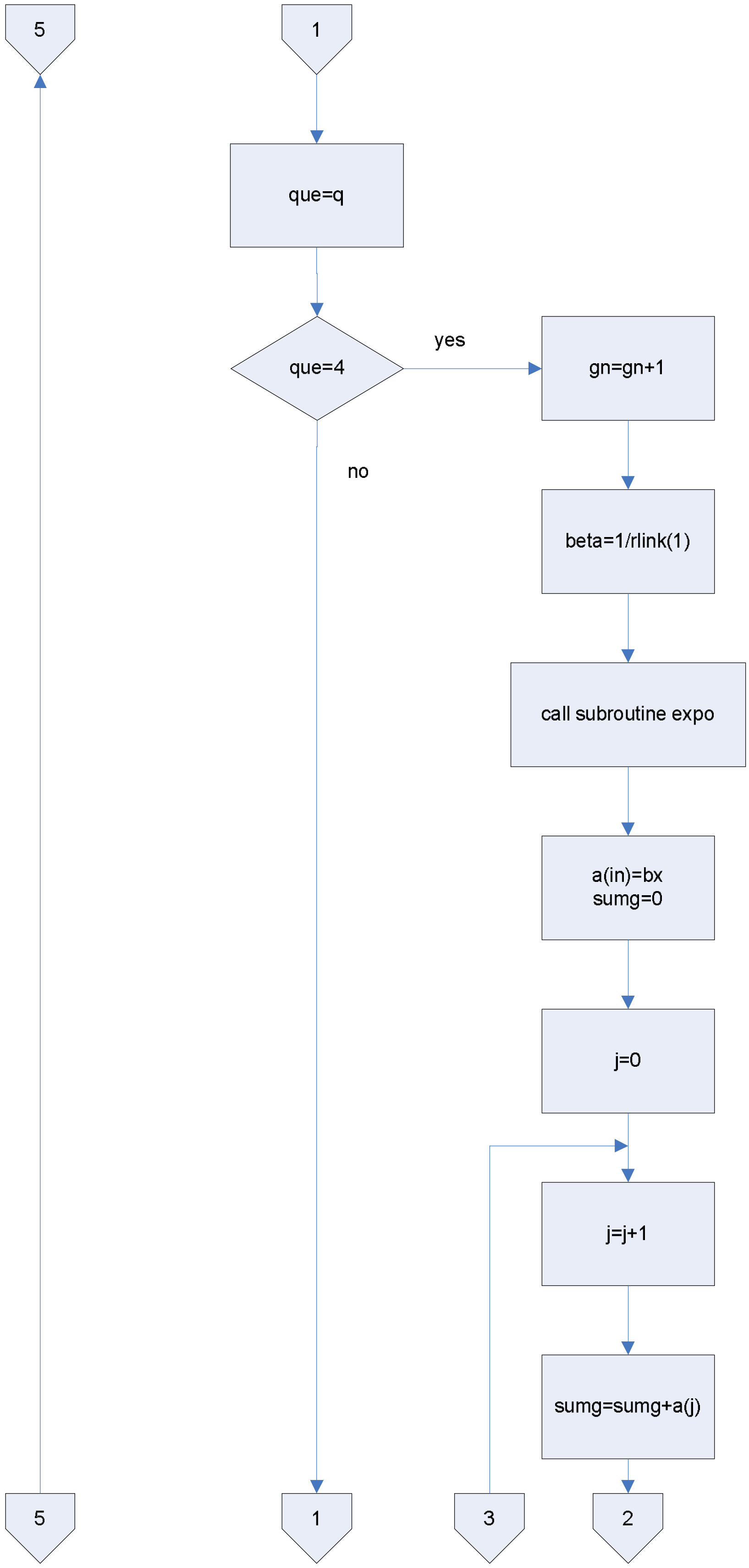

10
1
6
2
5
3
no
j=in
yes
driv(in)=0
cut(in)=scut
que=3
yes
rc3=rc3+1
no
beta=1/rlink(37)
call subroutine expo
a(in)=bx
scut=0
j=0
j=j+1
10
1
7
2
5
3

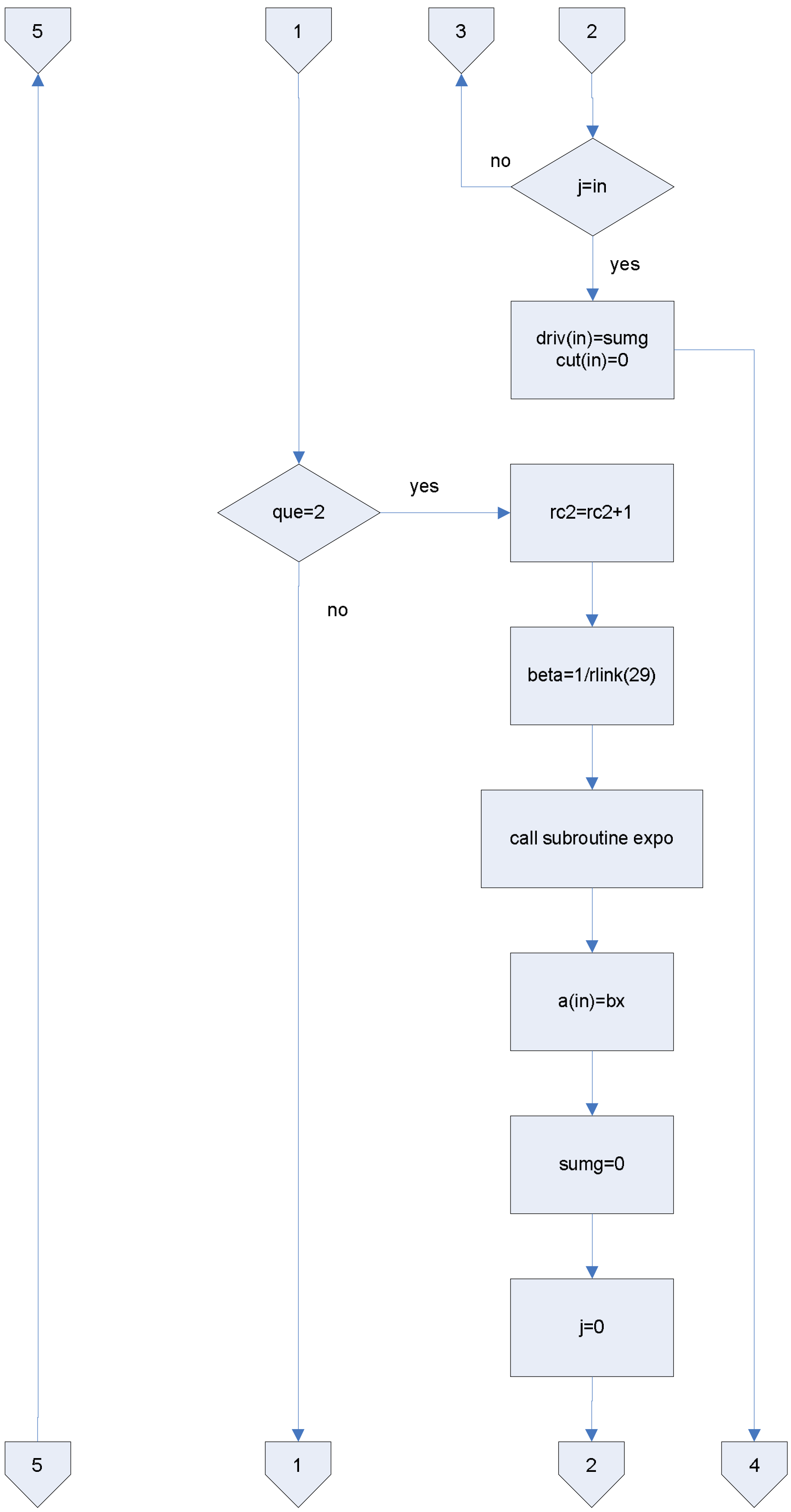

10
1
7
2
5
3
scut=scut+a(j)
no
j=in
yes
driv(in)=0
cut(in)=scut
end if
rang=rang+a(in)
rang3=rang3+a(in)
yes
o1-j1=1
no
delay=0
drive=0
k1=0
2
10
1
3

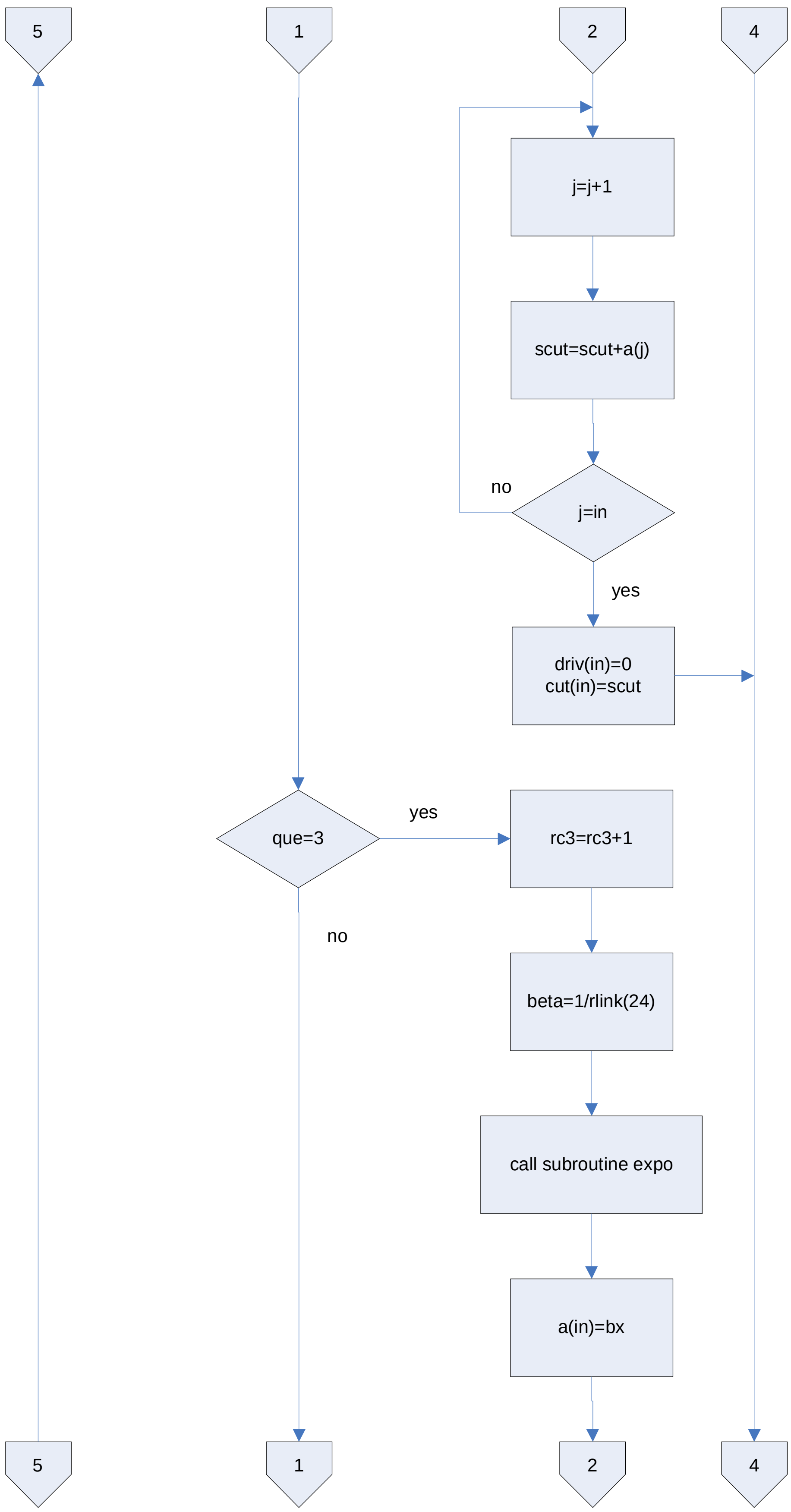

2
10
1
3
k1=k1+1
cut(k1)=0
yes
sumwa(k1)=0
no
cut(k1)>0
yes
sumwa(k1)=rang1-cut(k1)
no
end if
delay=delay+sumwa(k1)
drive=drive+driv(k1)
no
k1=in
yes
drive=drive+add*a(j1+1)
2
10
1
3

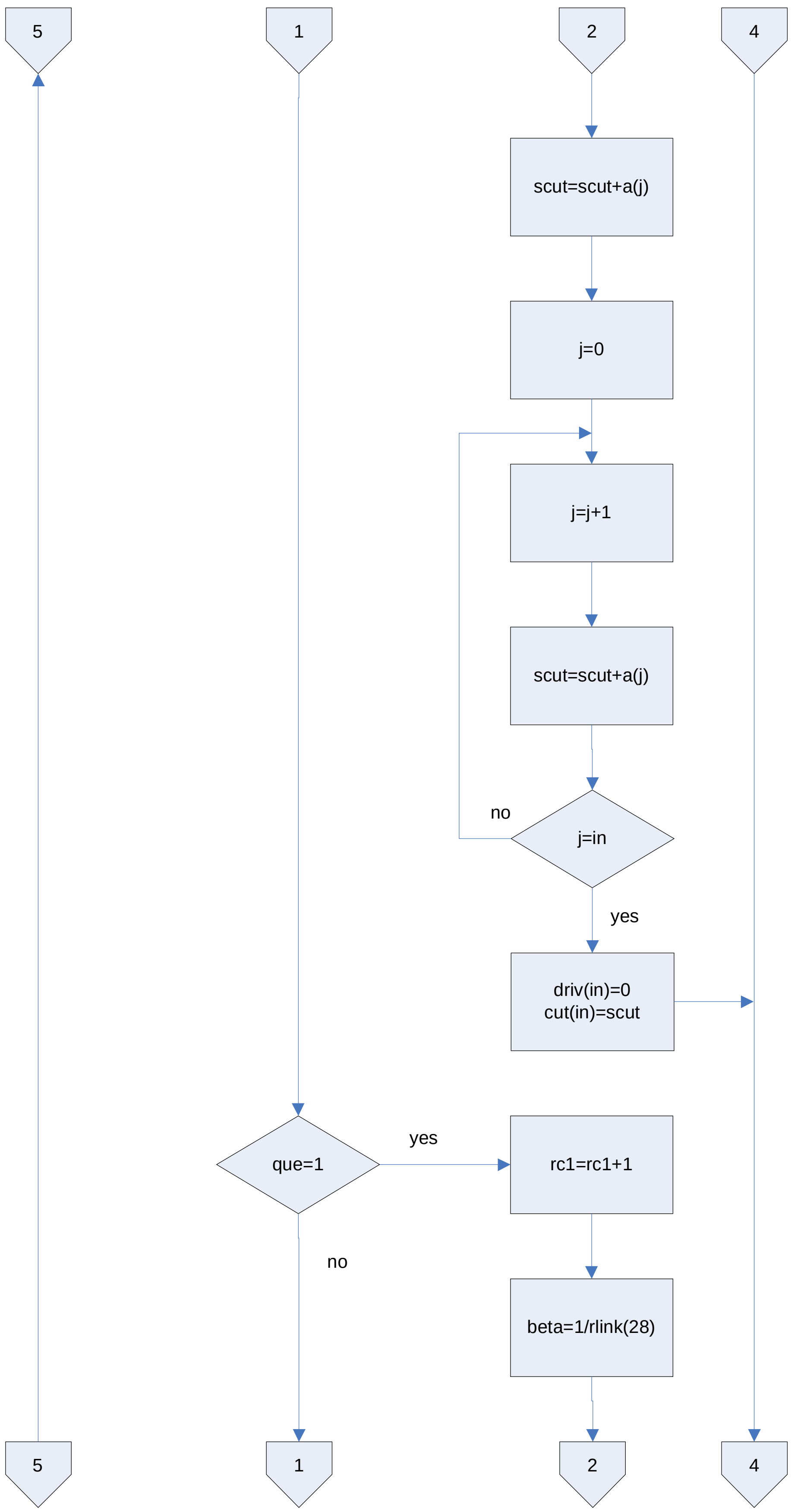

2
10
1
3
redn=rc2+rc3
delay3=delay/redn
g=2*gn/rang3
red=4*redn/rang3
wait=rang3
drive1=cdrive+drive
delay1=cdelay+delay
redn1=credn+redn
gn1=cgn+gn
cost=(delay1*gn1*(redn+gn))/(100*redn1*gn*drive1)
call subroutine fuzzy
degree=mu
no
degree=1
yes
cp3(ai)=wait
ai=ai+1
rang>1800
yes
no
10
1
4
3

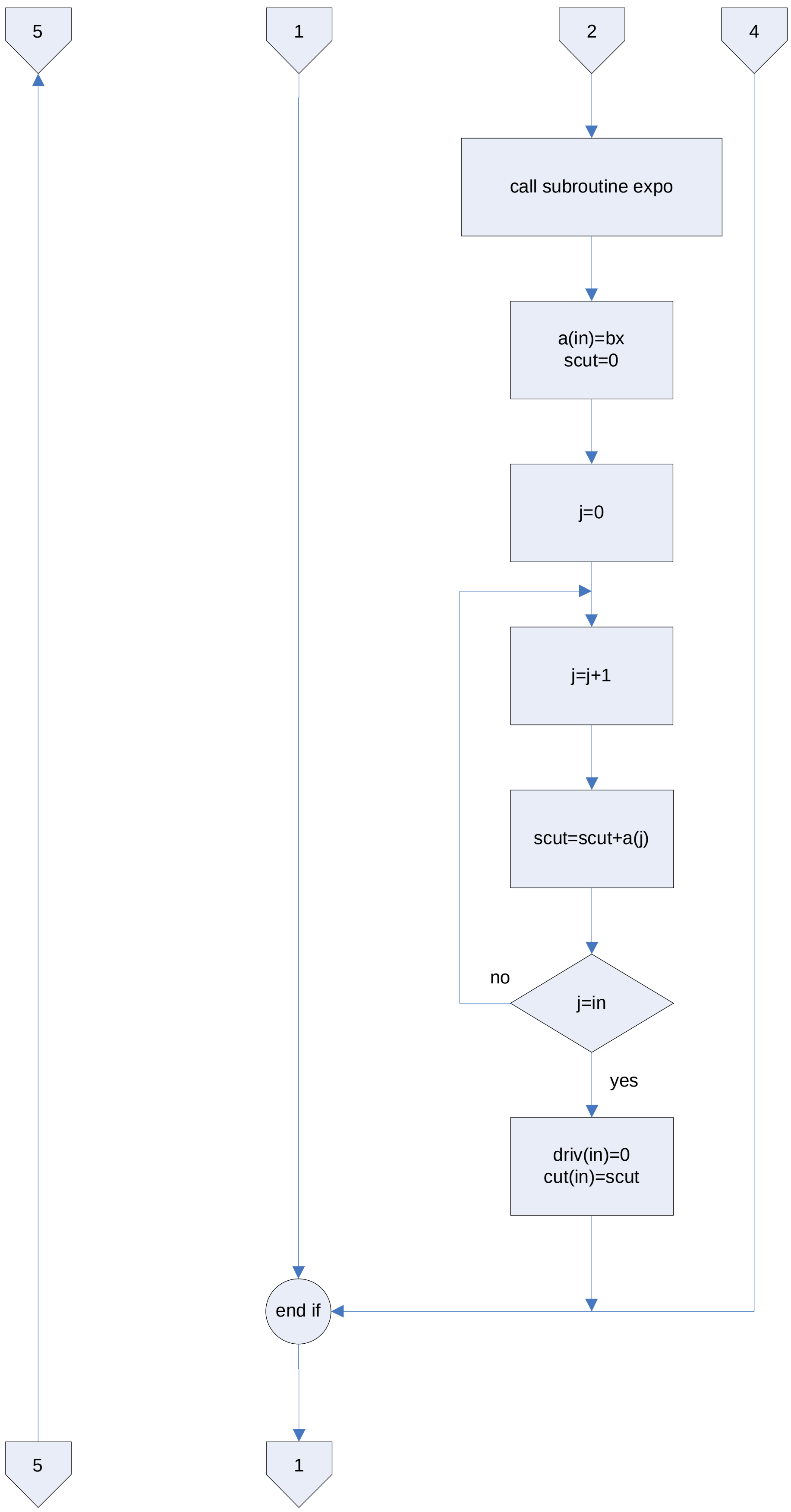

10
1
4
3
gn=rc2
add=rc2
cdrive=drive1
cdelay=delay1
credn=redn1
cgn=gn1
rc1=0
in=0
rang1=0
i1=o1
b1=1
c1=1
v1=1
xo=1
gn=0
s1=0
i1=0
s1=0
in=0
rc1=0
rc2=0
rc3=0
rang=0
rang1=0
i1=i1+1
ri1=i1
in=in+1
p1=rinten(28)/(rinten(28)+rinten(29)+rinten(24)+rinten(1))
p2=rinten(29)/(rinten(28)+rinten(29)+rinten(24)+rinten(1))
p3=rinten(24)/(rinten(28)+rinten(29)+rinten(24)+rinten(1))
p4=rinten(1)/(rinten(28)+rinten(29)+rinten(24)+rinten(1))
call subroutine allocate
5
1

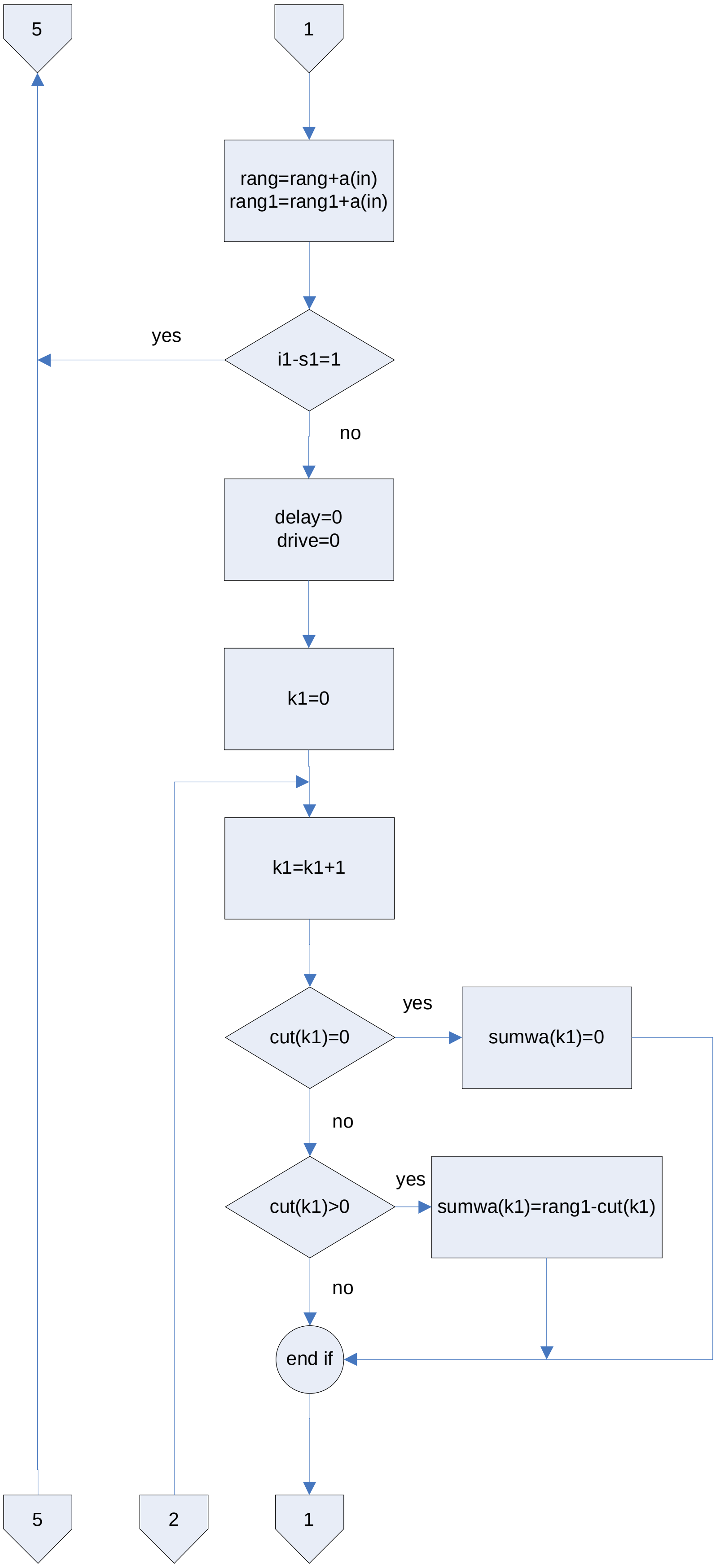

5
1
que=q
que=4
yes
gn=gn+1
no
beta=1/rlink(1)
call subroutine expo
a(in)=bx
sumg=0
j=0
j=j+1
sumg=sumg+a(j)
5
1
3
2

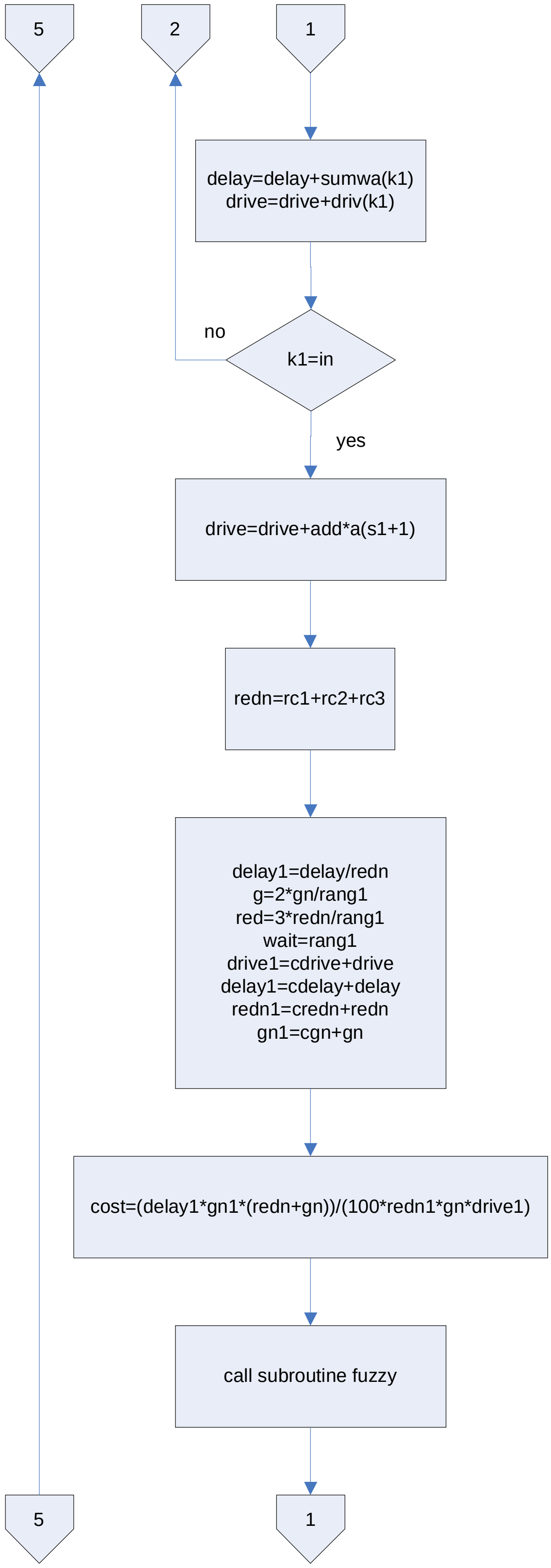

5
1
3
2
no
j=in
yes
driv(in)=sumg
cut(in)=0
que=2
yes
rc2=rc2+1
no
beta=1/rlink(29)
call subroutine expo
a(in)=bx
sumg=0
j=0
5
1
2
4

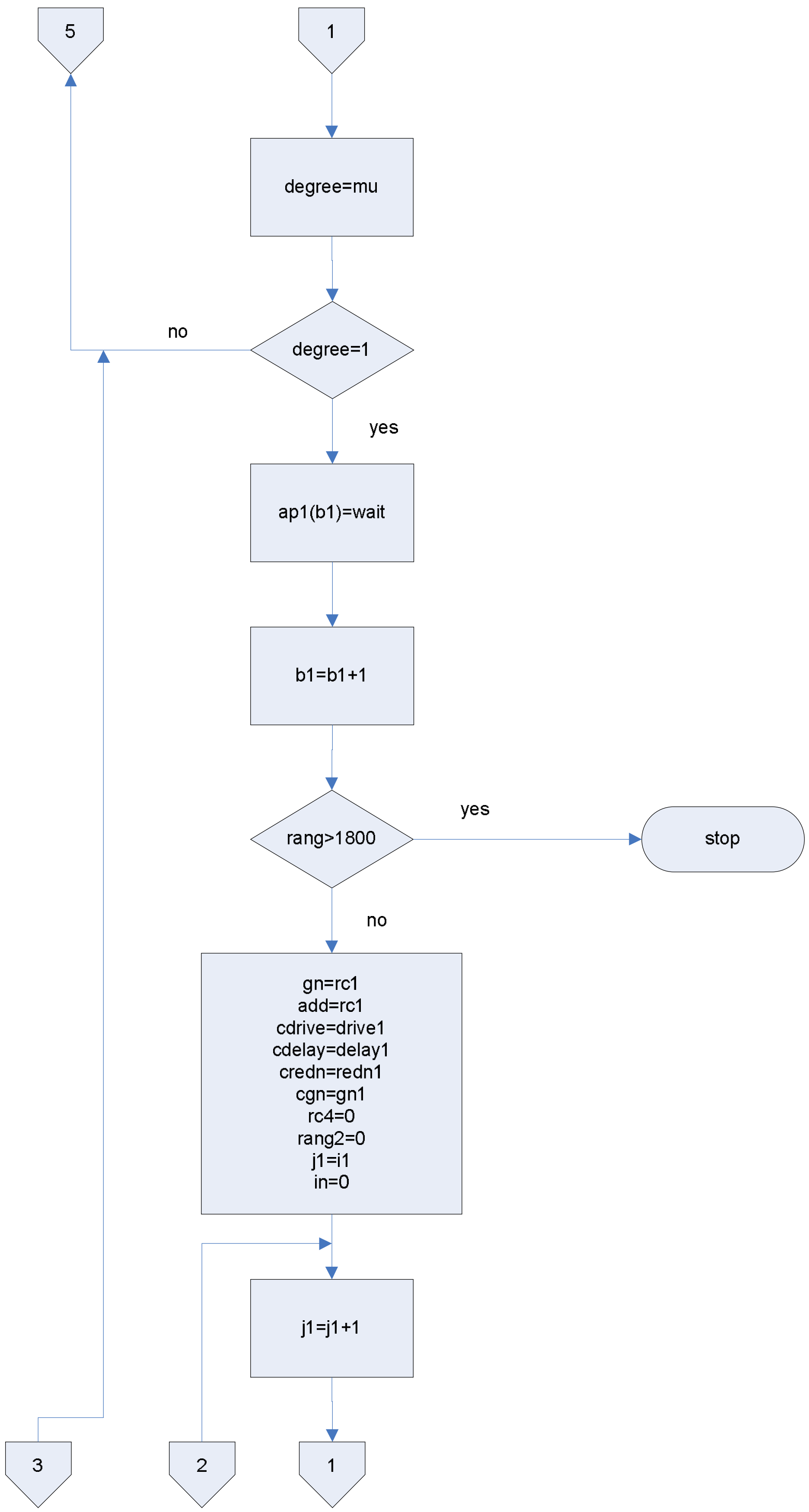

5
1
2
4
j=j+1
scut=scut+a(j)
no
j=in
yes
driv(in)=0
cut(in)=scut
que=3
yes
rc3=rc3+1
no
beta=1/rlink(24)
call subroutine expo
a(in)=bx
5
1
2
4

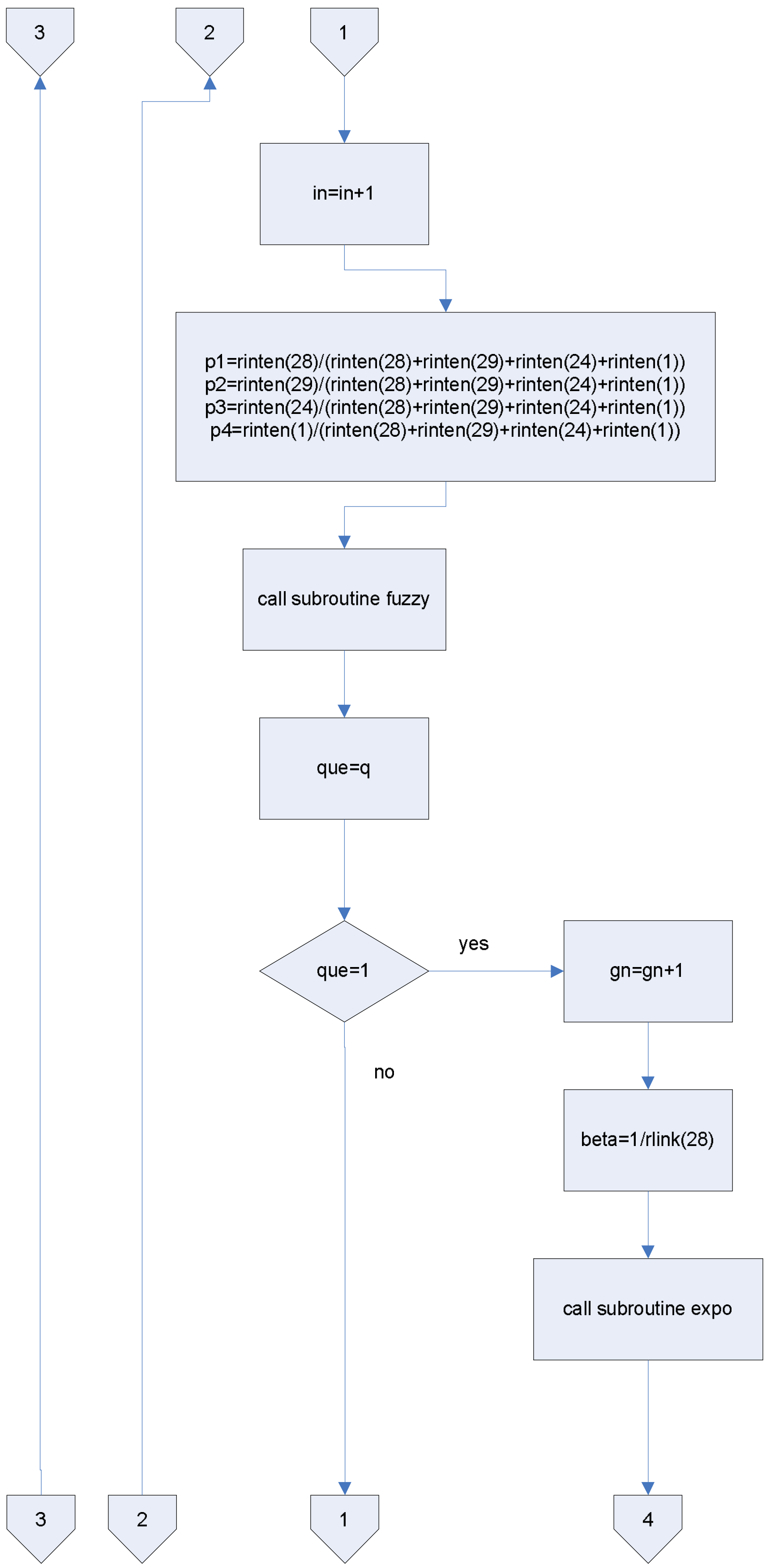

5
1
2
4
scut=scut+a(j)
j=0
j=j+1
scut=scut+a(j)
no
j=in
yes
driv(in)=0
cut(in)=scut
yes
que=1
rc1=rc1+1
no
beta=1/rlink(28)
5
1
2
4

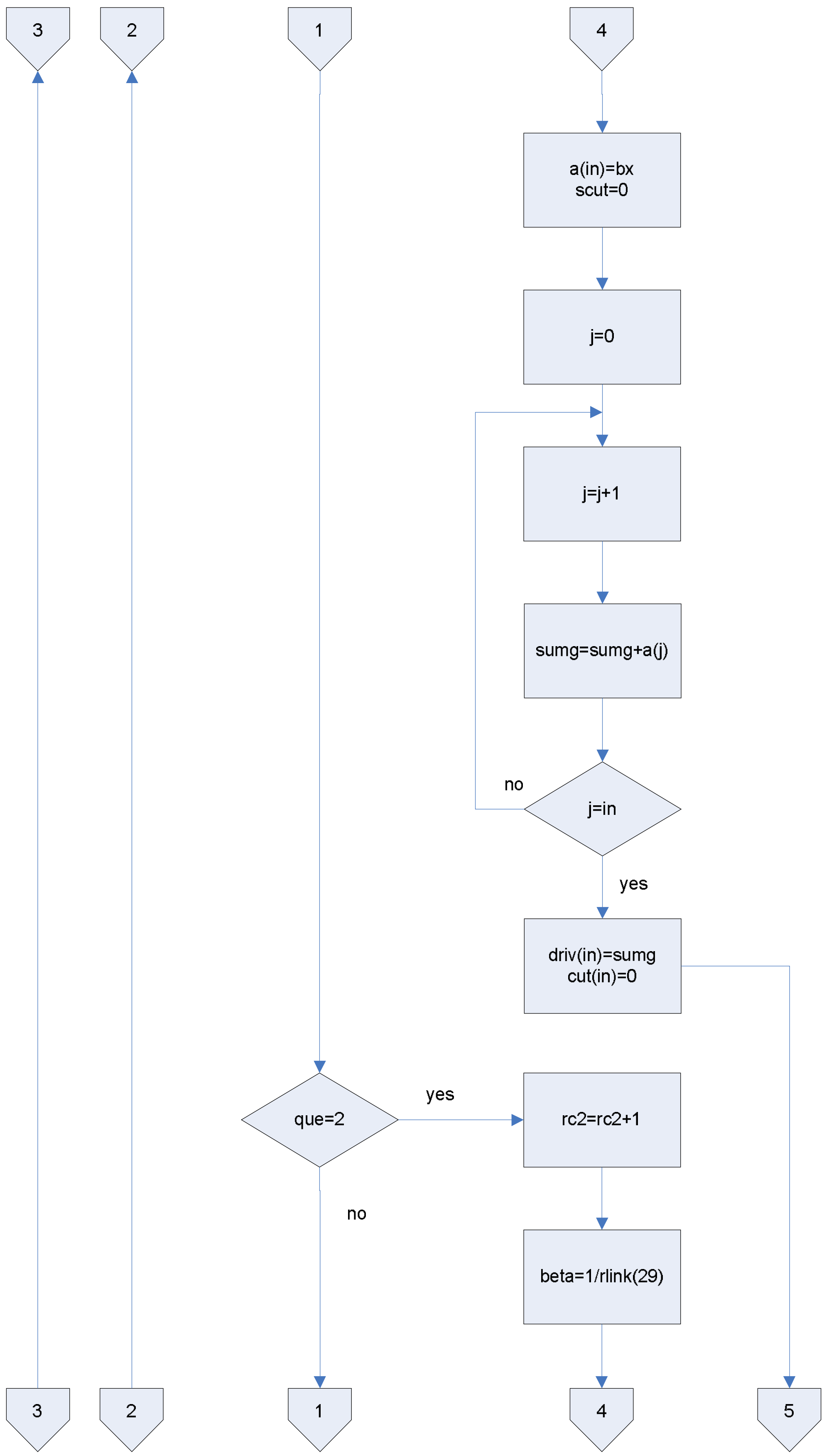

5
1
2
4
call subroutine expo
a(in)=bx
scut=0
j=0
j=j+1
scut=scut+a(j)
no
j=in
yes
driv(in)=0
cut(in)=scut
end if
5
1

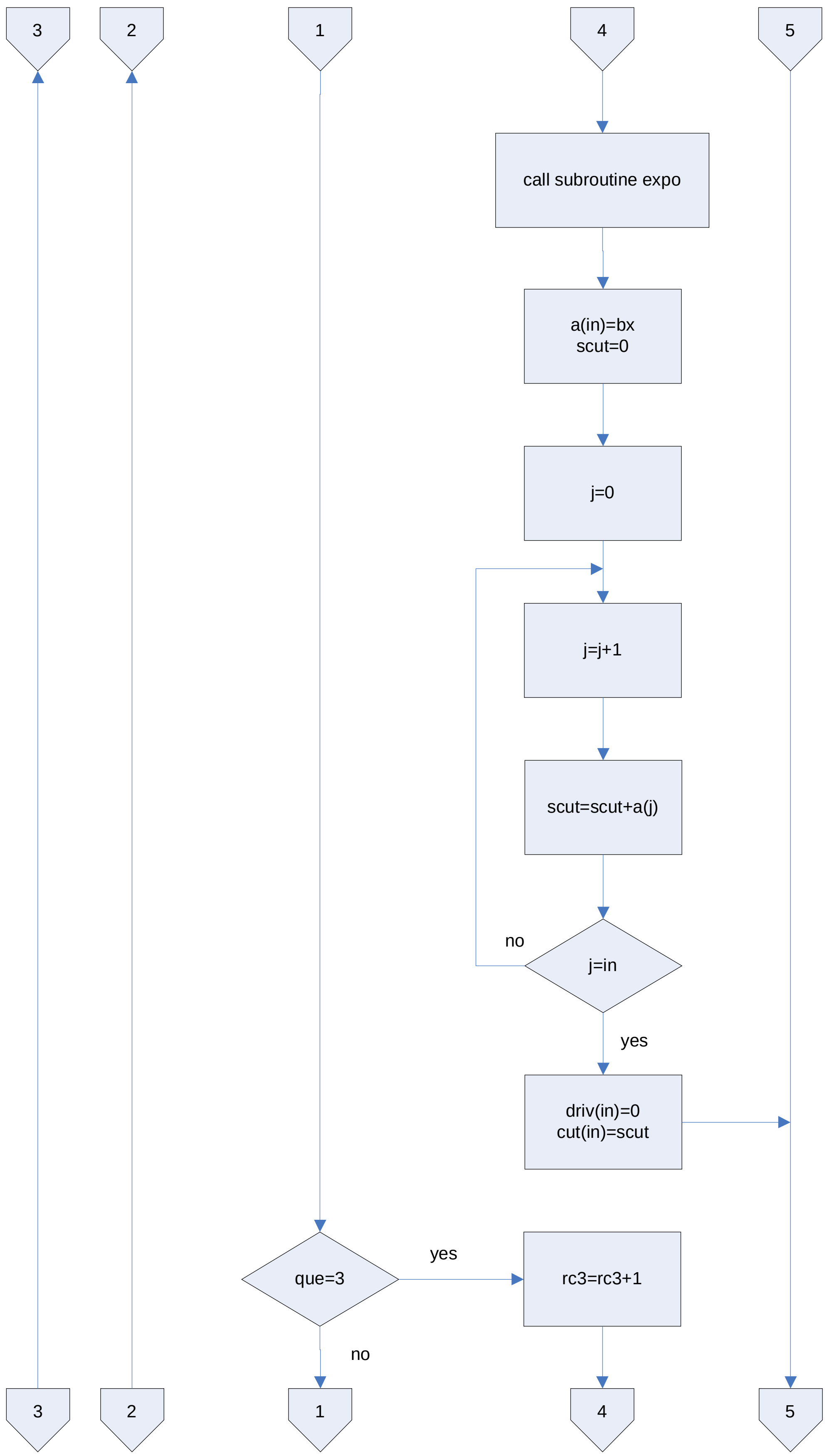

5
1
rang=rang+a(in)
rang1=rang1+a(in)
yes
i1-s1=1
no
delay=0
drive=0
k1=0
k1=k1+1
yes
cut(k1)=0
sumwa(k1)=0
no
yes
cut(k1)>0
sumwa(k1)=rang1-cut(k1)
no
end if
5
2
1

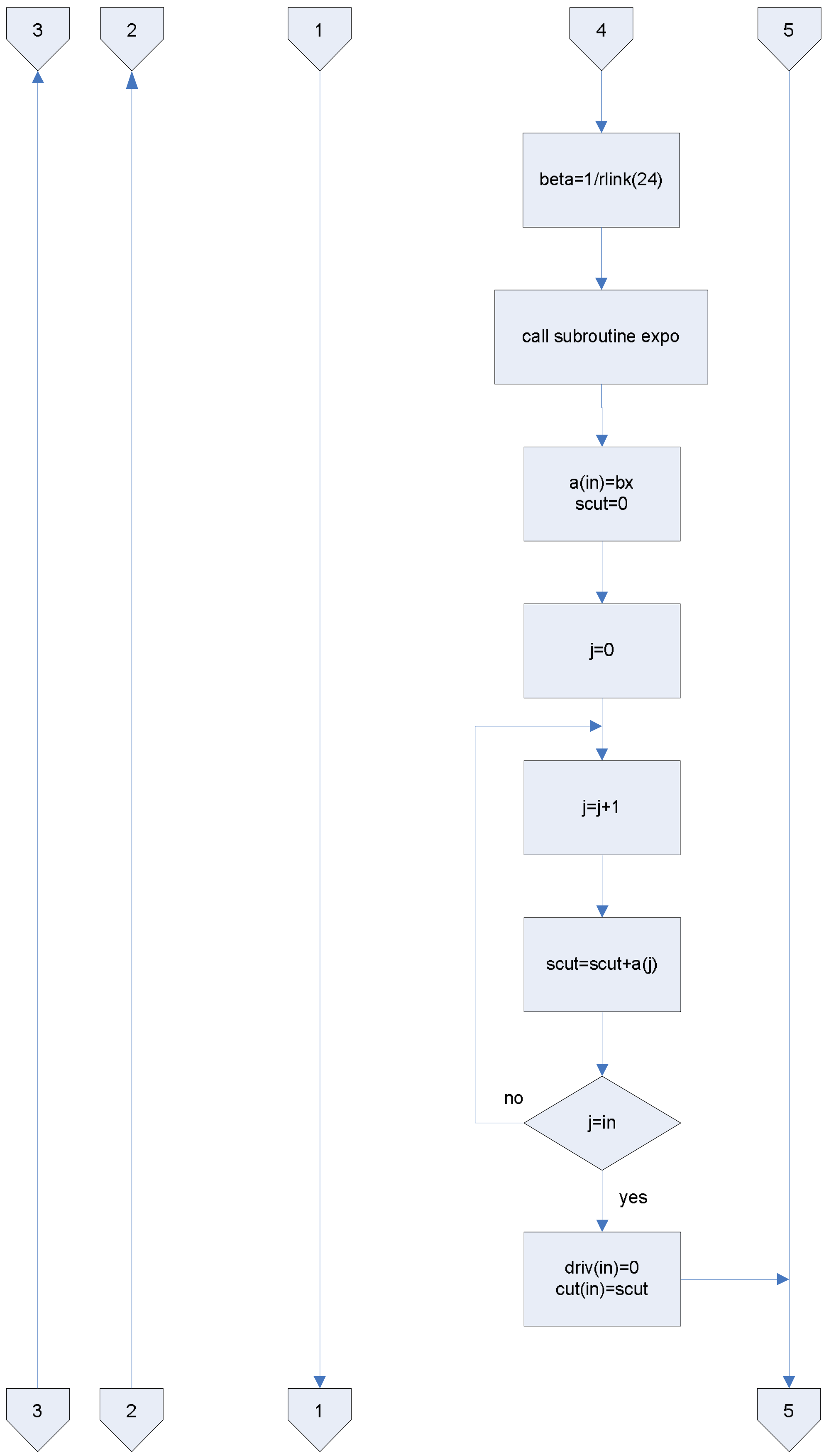

5
2
1
delay=delay+sumwa(k1)
drive=drive+driv(k1)
no
k1=in
yes
drive=drive+add*a(s1+1)
redn=rc1+rc2+rc3
delay1=delay/redn
g=2*gn/rang1
red=3*redn/rang1
wait=rang1
drive1=cdrive+drive
delay1=cdelay+delay
redn1=credn+redn
gn1=cgn+gn
cost=(delay1*gn1*(redn+gn))/(100*redn1*gn*drive1)
call subroutine fuzzy
5
1

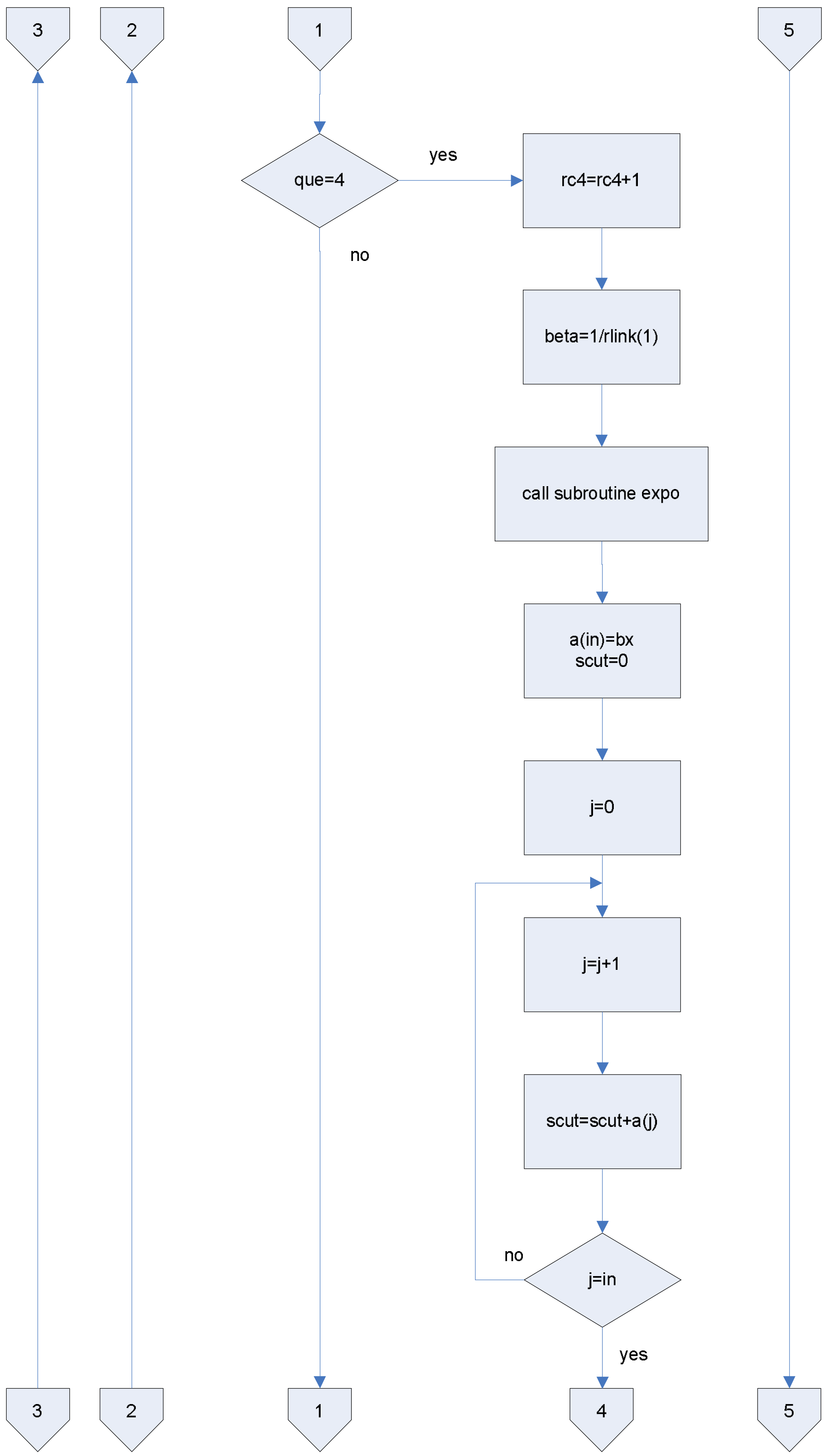

5
1
degree=mu
no
degree=1
yes
ap1(b1)=wait
b1=b1+1
yes
rang>1800
stop
no
gn=rc1
add=rc1
cdrive=drive1
cdelay=delay1
credn=redn1
cgn=gn1
rc4=0
rang2=0
j1=i1
in=0
j1=j1+1
3
2
1

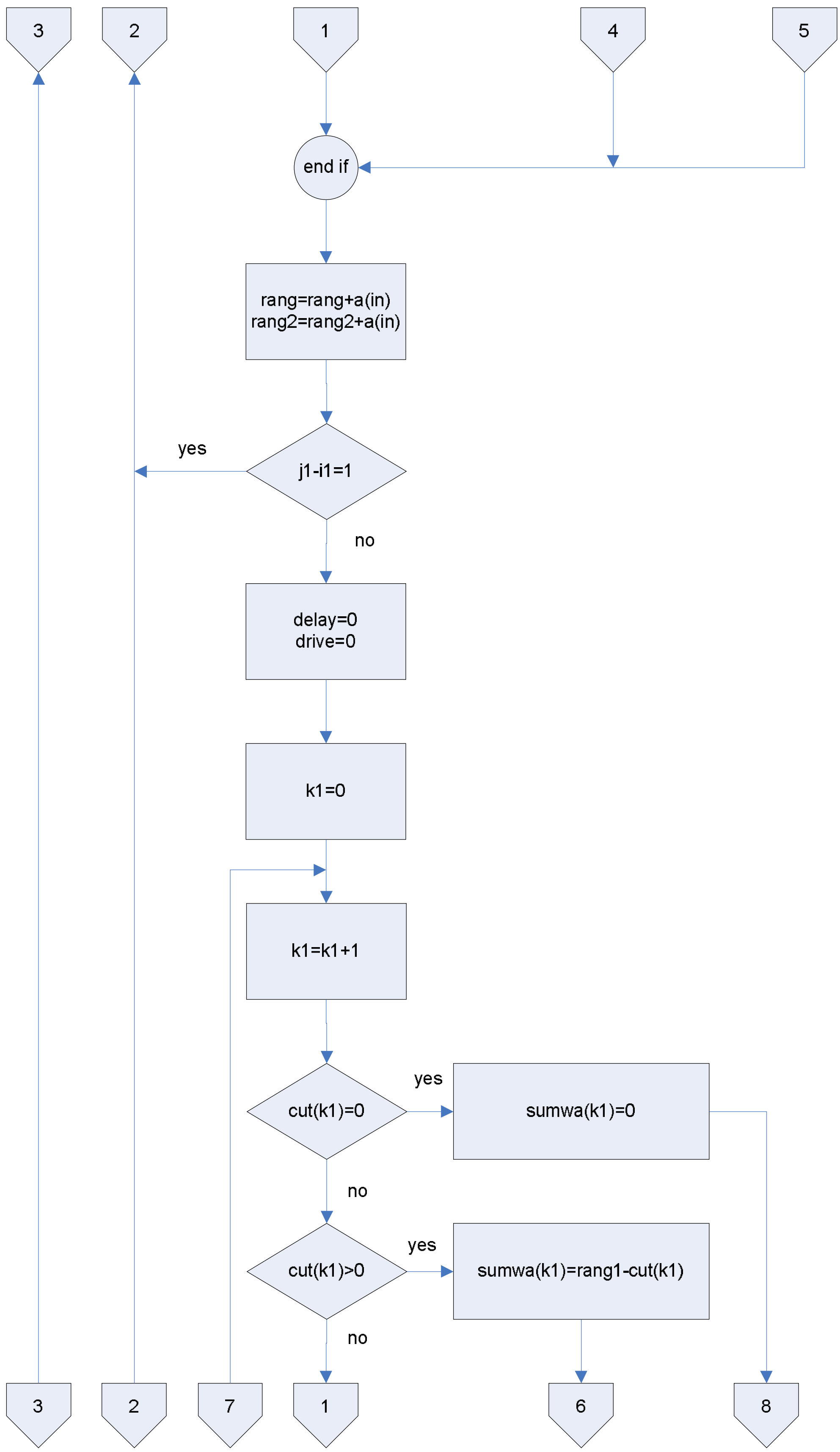

3
2
1
in=in+1
p1=rinten(28)/(rinten(28)+rinten(29)+rinten(24)+rinten(1))
p2=rinten(29)/(rinten(28)+rinten(29)+rinten(24)+rinten(1))
p3=rinten(24)/(rinten(28)+rinten(29)+rinten(24)+rinten(1))
p4=rinten(1)/(rinten(28)+rinten(29)+rinten(24)+rinten(1))
call subroutine fuzzy
que=q
que=1
yes
gn=gn+1
no
beta=1/rlink(28)
call subroutine expo
3
2
1
4

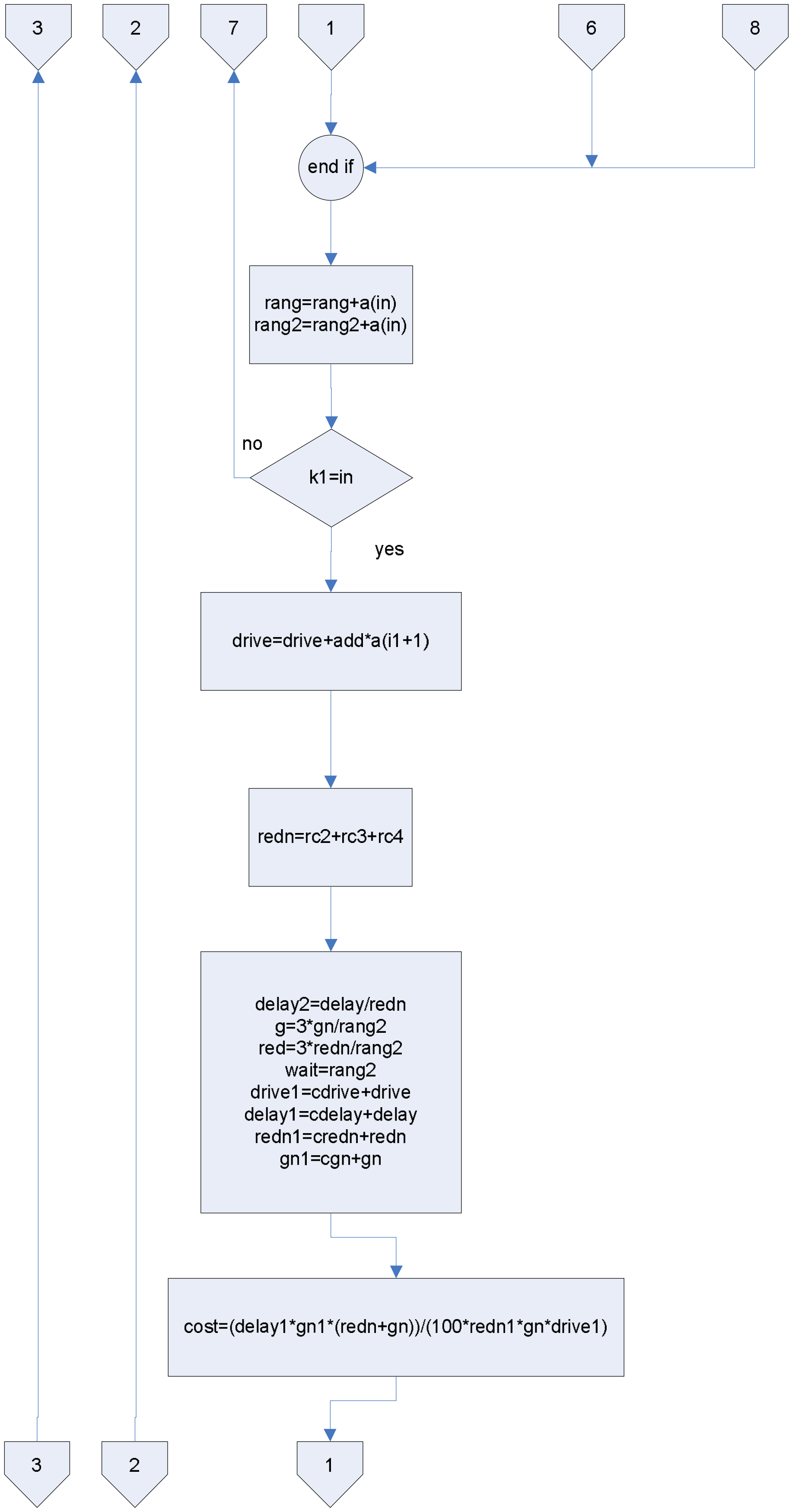

3
2
1
4
a(in)=bx
scut=0
j=0
j=j+1
sumg=sumg+a(j)
no
j=in
yes
driv(in)=sumg
cut(in)=0
que=2
yes
rc2=rc2+1
no
beta=1/rlink(29)
3
2
1
4
5

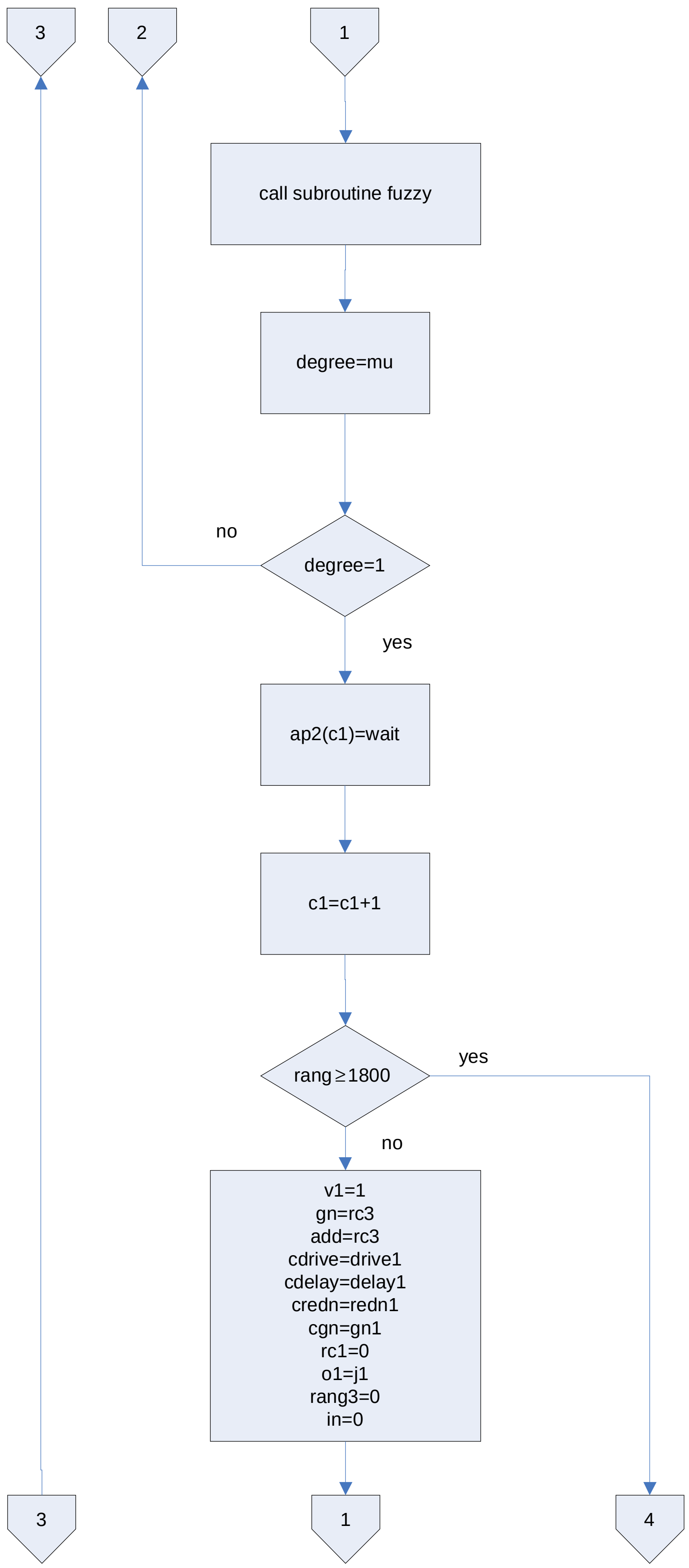

3
2
1
4
5
call subroutine expo
a(in)=bx
scut=0
j=0
j=j+1
scut=scut+a(j)
no
j=in
yes
driv(in)=0
cut(in)=scut
que=3
yes
rc3=rc3+1
no
3
2
1
4
5

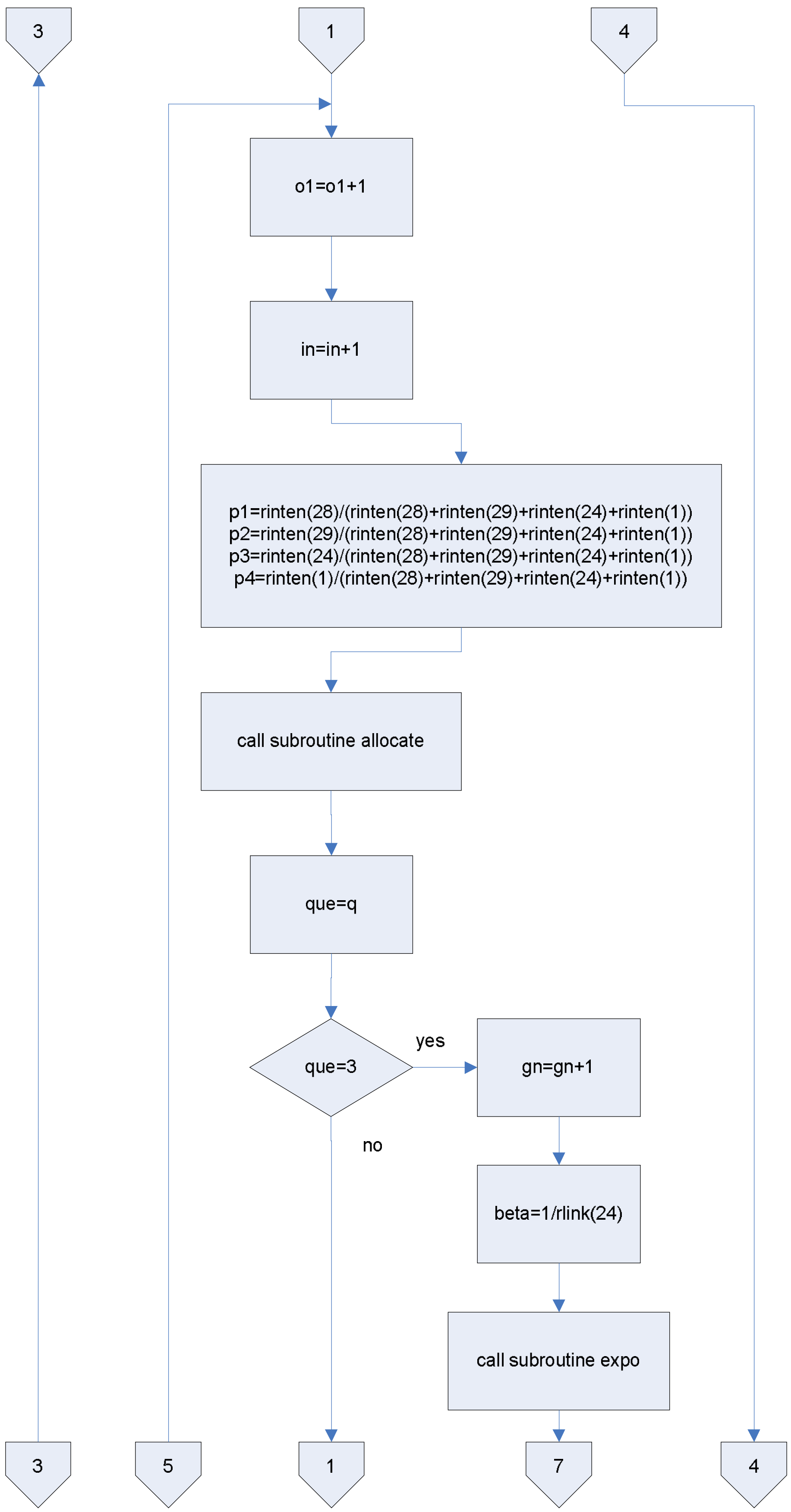

3
2
1
4
5
beta=1/rlink(24)
call subroutine expo
a(in)=bx
scut=0
j=0
j=j+1
scut=scut+a(j)
no
j=in
yes
driv(in)=0
cut(in)=scut
3
2
1
5

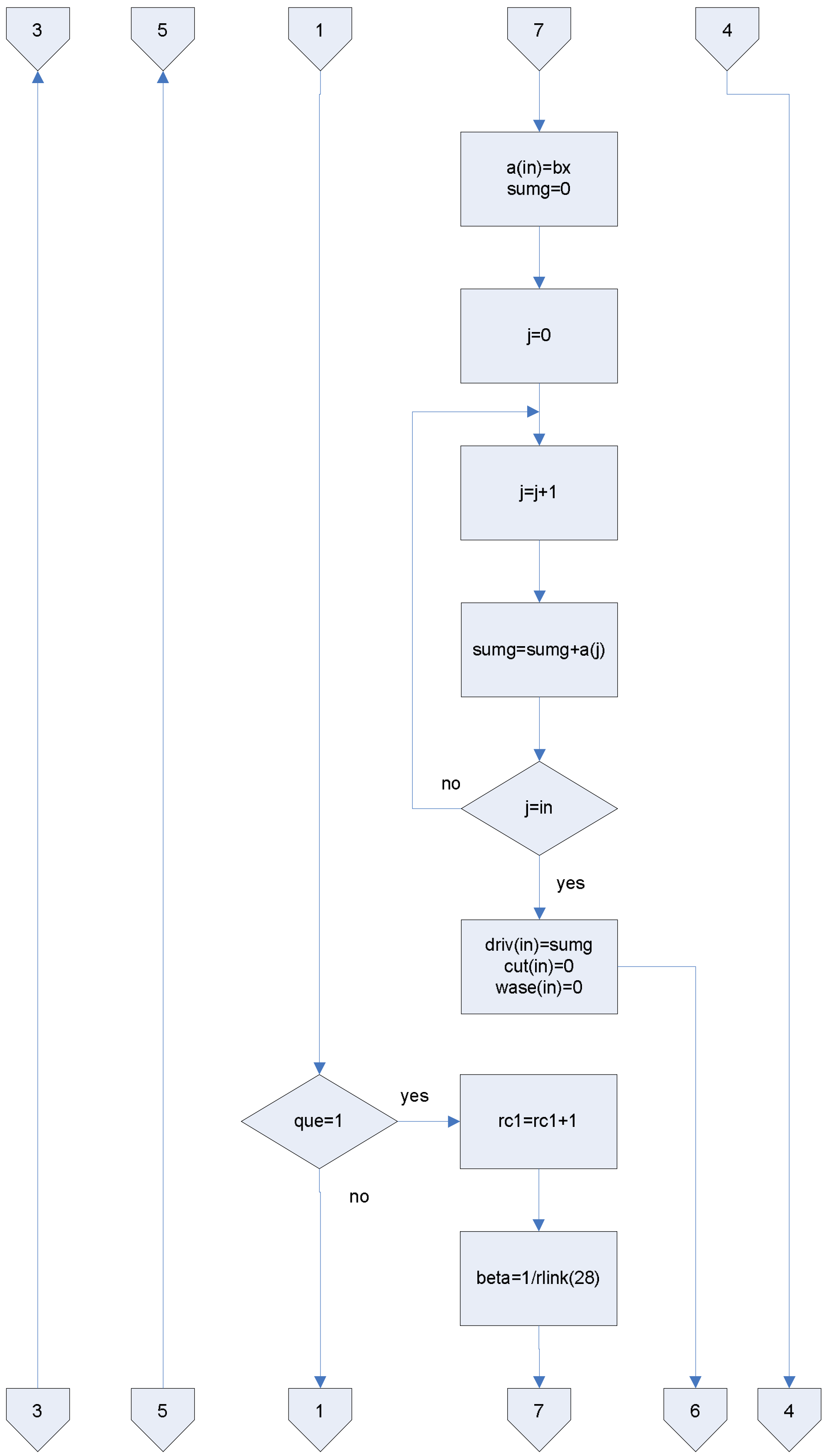

3
2
1
5
que=4
yes
no
rc4=rc4+1
beta=1/rlink(1)
call subroutine expo
a(in)=bx
scut=0
j=0
j=j+1
scut=scut+a(j)
no
j=in
yes
3
2
1
4
5


3
2
1
4
5
end if
rang=rang+a(in)
rang2=rang2+a(in)
yes
j1-i1=1
no
delay=0
drive=0
k1=0
k1=k1+1
yes
cut(k1)=0
sumwa(k1)=0
no
yes
cut(k1)>0
sumwa(k1)=rang1-cut(k1)
no
3
2
7
1
6
8

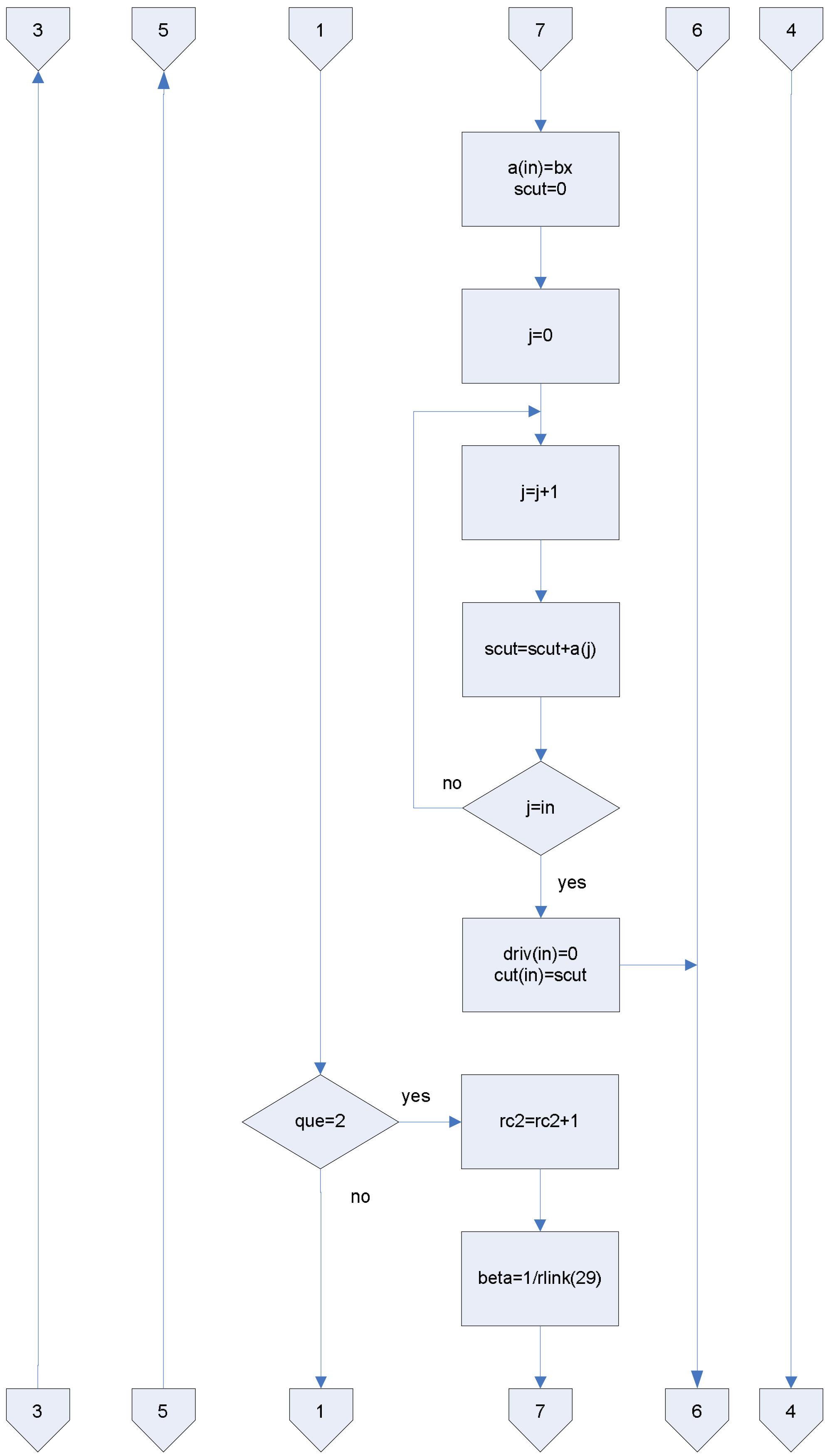

3
2
7
1
6
8
end if
rang=rang+a(in)
rang2=rang2+a(in)
no
k1=in
yes
drive=drive+add*a(i1+1)
redn=rc2+rc3+rc4
delay2=delay/redn
g=3*gn/rang2
red=3*redn/rang2
wait=rang2
drive1=cdrive+drive
delay1=cdelay+delay
redn1=credn+redn
gn1=cgn+gn
cost=(delay1*gn1*(redn+gn))/(100*redn1*gn*drive1)
3
2
1

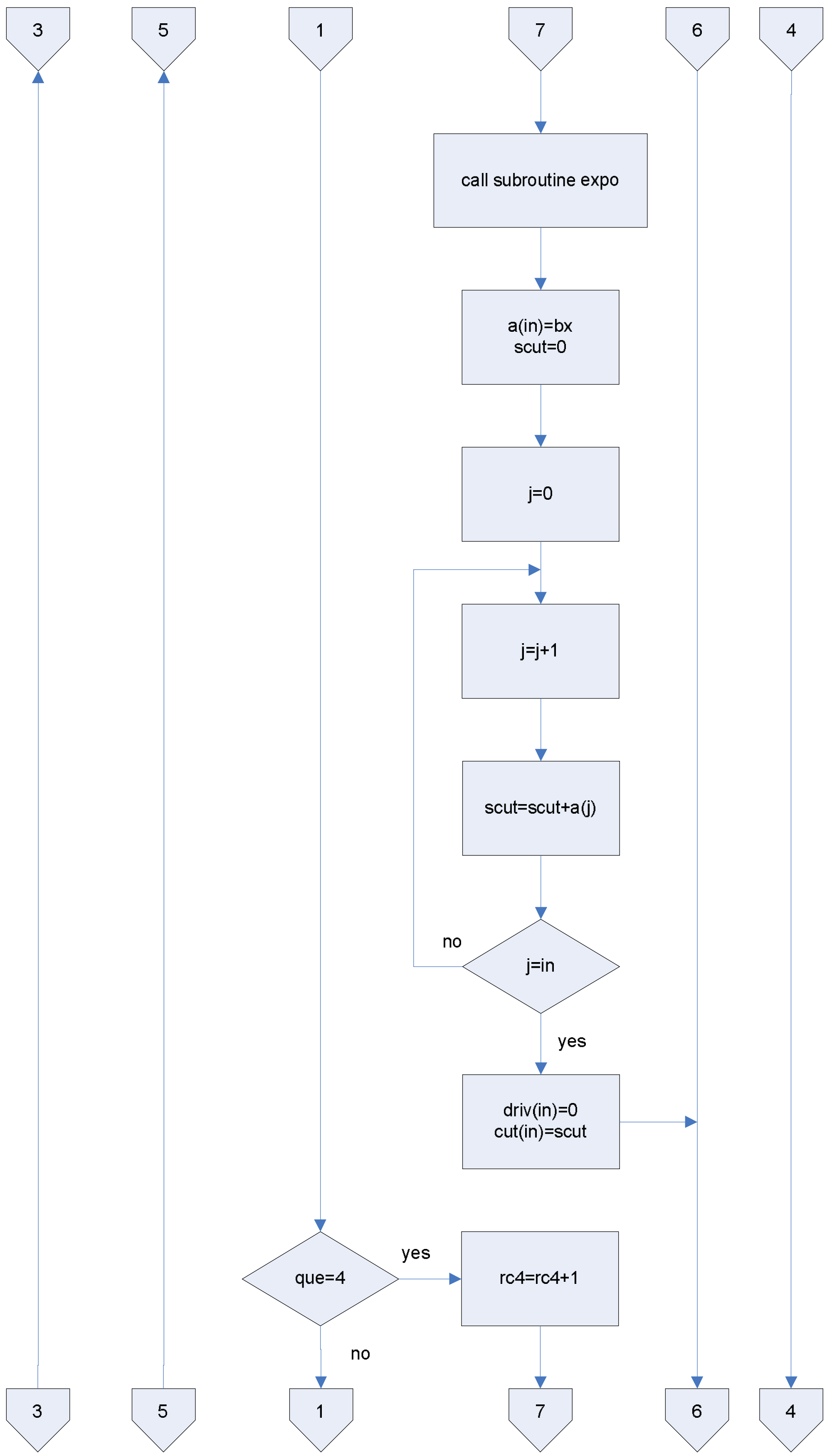

3
2
1
call subroutine fuzzy
degree=mu
no
degree=1
yes
ap2(c1)=wait
c1=c1+1
rang≥1800
yes
no
v1=1
gn=rc3
add=rc3
cdrive=drive1
cdelay=delay1
credn=redn1
cgn=gn1
rc1=0
o1=j1
rang3=0
in=0
3
1
4


3
1
4
o1=o1+1
in=in+1
p1=rinten(28)/(rinten(28)+rinten(29)+rinten(24)+rinten(1))
p2=rinten(29)/(rinten(28)+rinten(29)+rinten(24)+rinten(1))
p3=rinten(24)/(rinten(28)+rinten(29)+rinten(24)+rinten(1))
p4=rinten(1)/(rinten(28)+rinten(29)+rinten(24)+rinten(1))
call subroutine allocate
que=q
que=3
yes
no
gn=gn+1
beta=1/rlink(24)
call subroutine expo
3
5
1
7
4

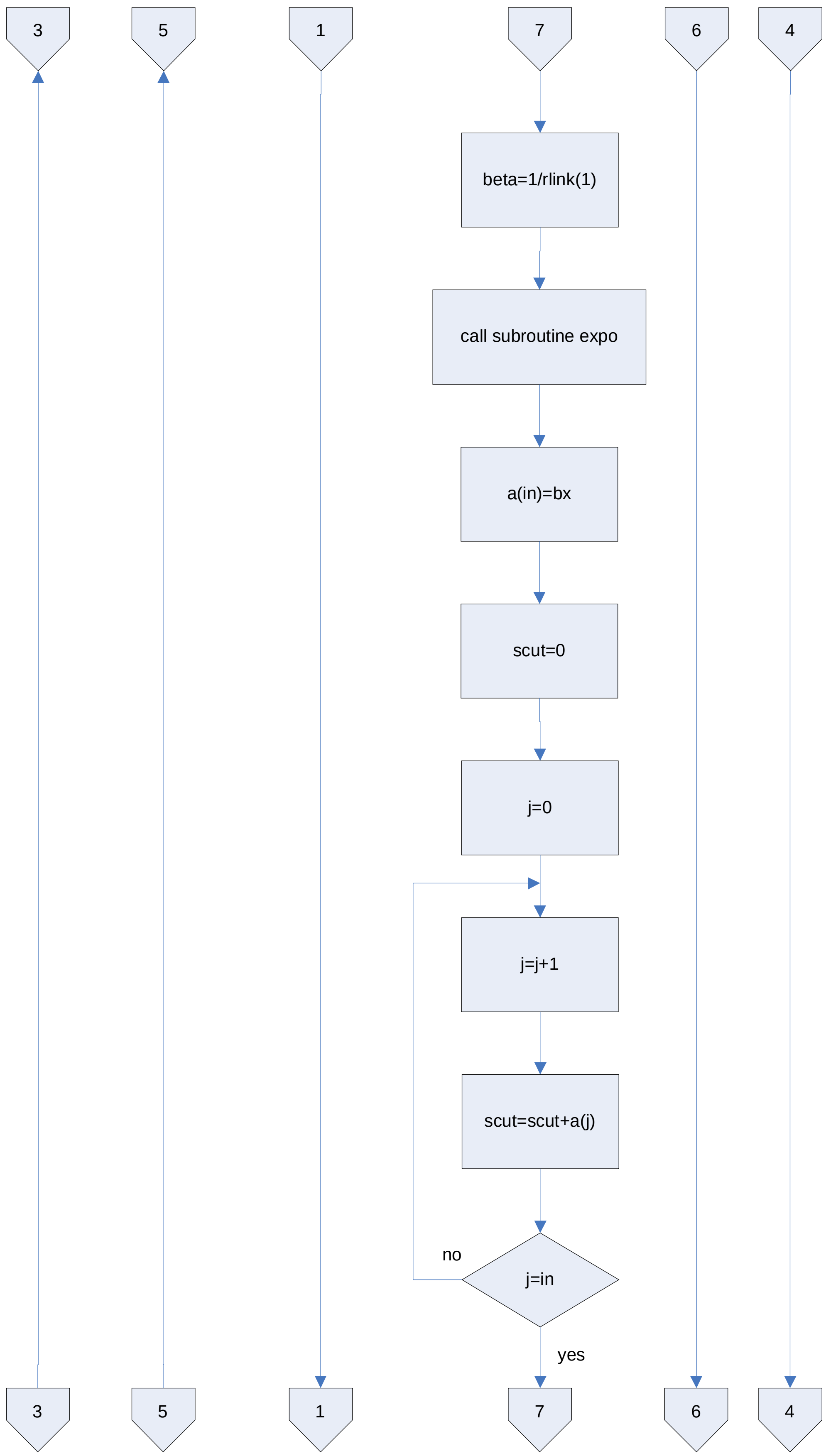


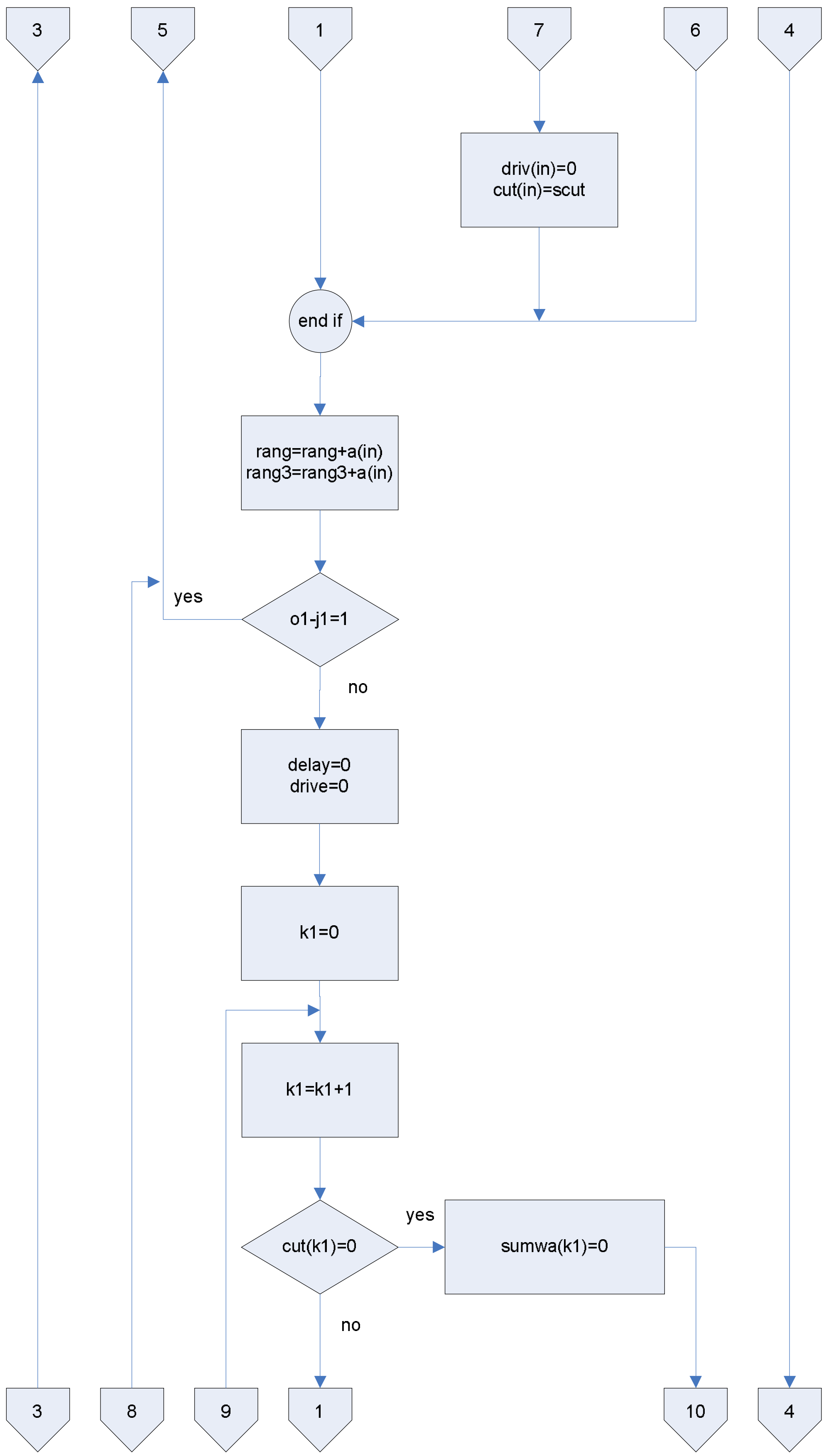
3
5
1
7
4
a(in)=bx
sumg=0
j=0
j=j+1
sumg=sumg+a(j)
no
j=in
yes
driv(in)=sumg
cut(in)=0
wase(in)=0
yes
que=1
rc1=rc1+1
no
beta=1/rlink(28)
3
5
1
7
6
4


3
5
1
7
6
4
a(in)=bx
scut=0
j=0
j=j+1
scut=scut+a(j)
no
j=in
yes
driv(in)=0
cut(in)=scut
que=2
yes
rc2=rc2+1
no
beta=1/rlink(29)
3
5
1
7
6
4

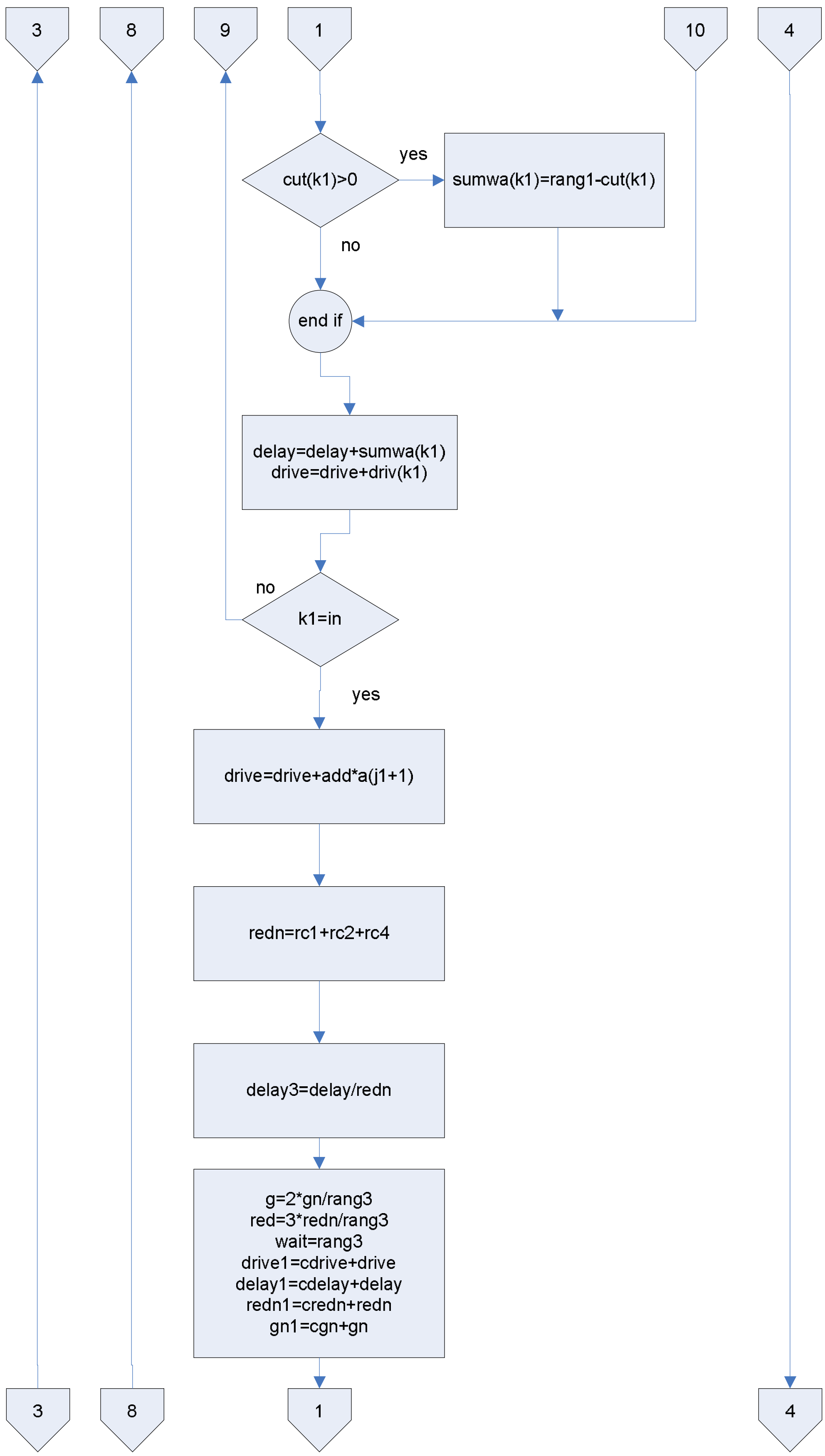


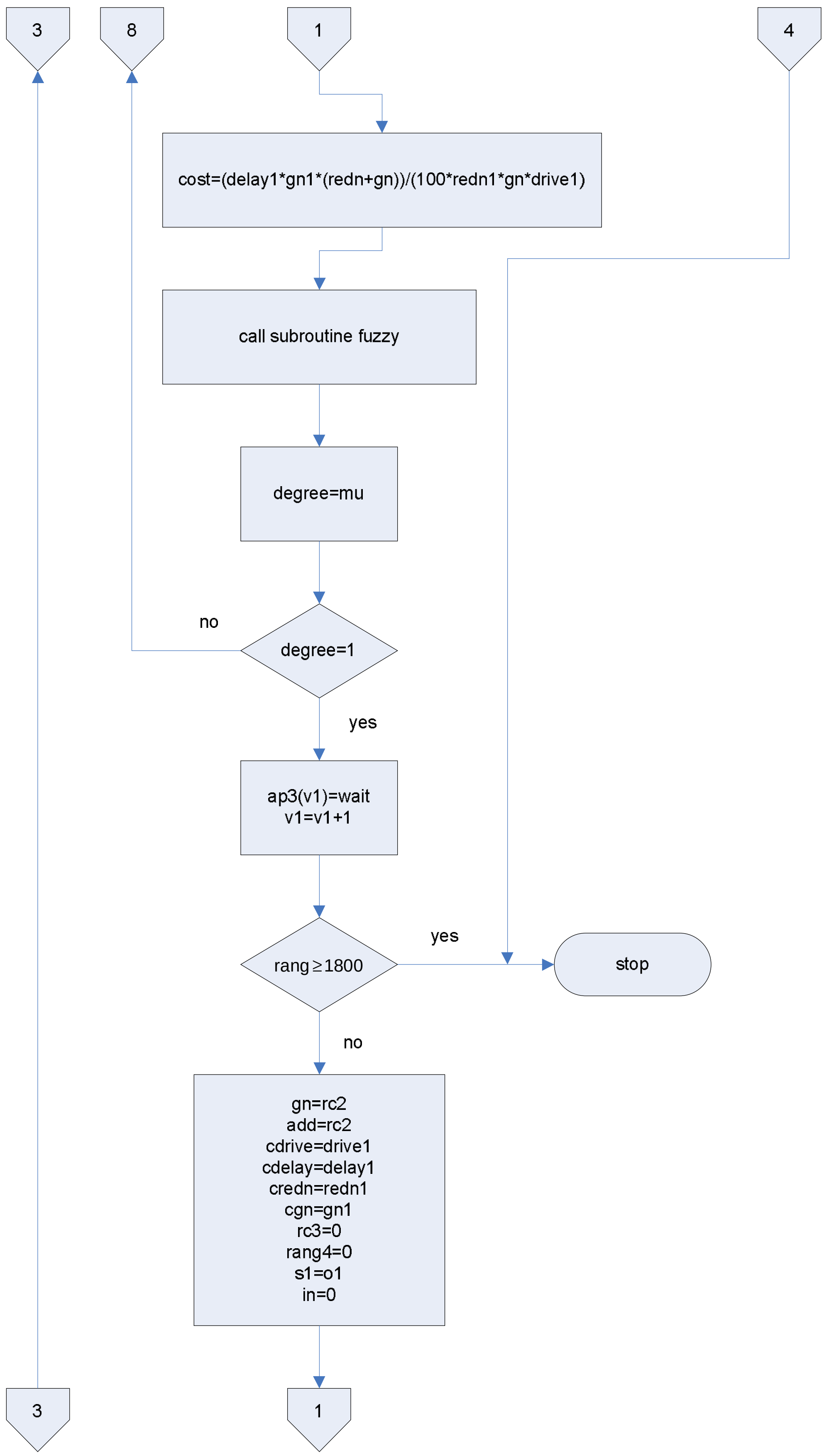
3
5
1
7
6
4
call subroutine expo
a(in)=bx
scut=0
j=0
j=j+1
scut=scut+a(j)
j=in
no
yes
driv(in)=0
cut(in)=scut
que=4
yes
rc4=rc4+1
no
3
5
1
7
6
4

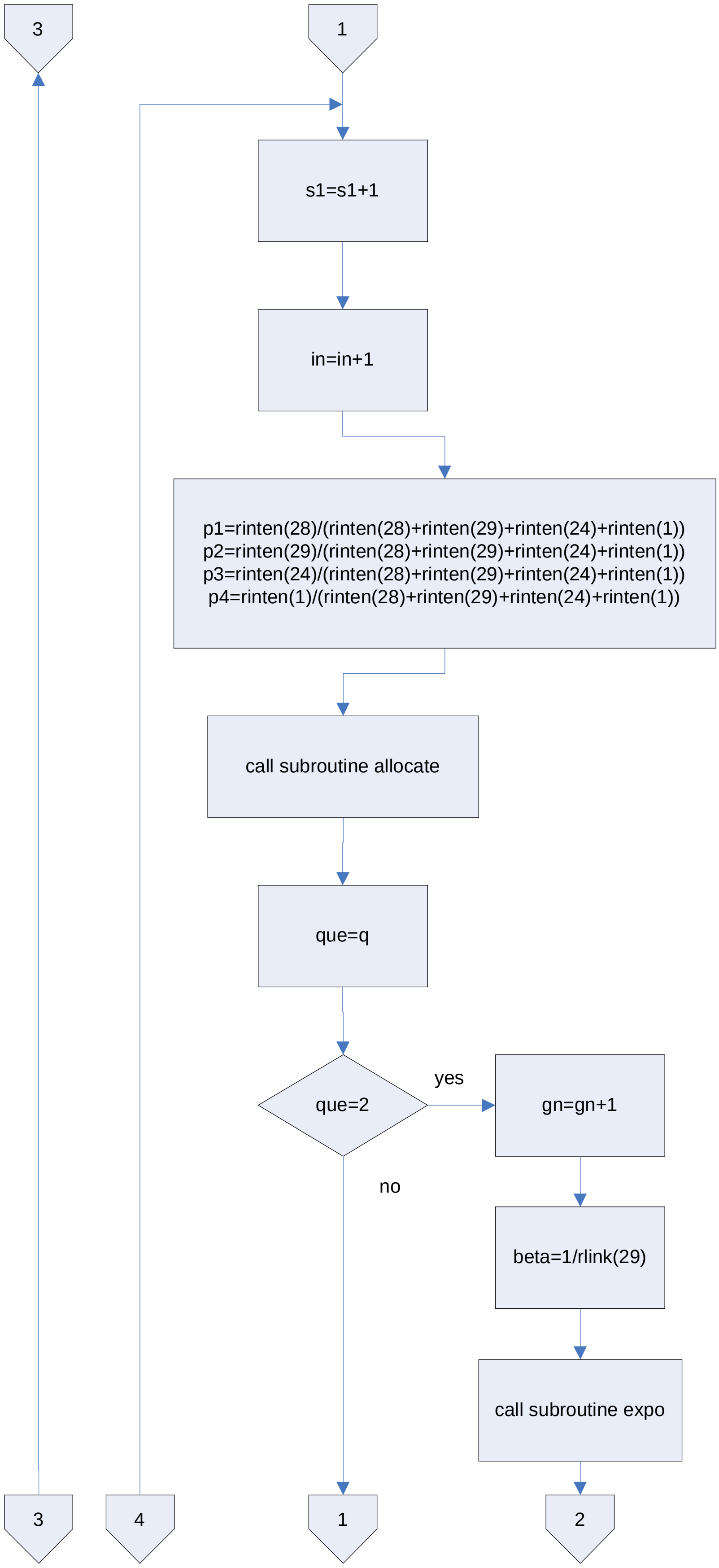

3
5
1
7
6
4
beta=1/rlink(1)
call subroutine expo
a(in)=bx
scut=0
j=0
j=j+1
scut=scut+a(j)
no
j=in
yes
3
5
1
7
6
4

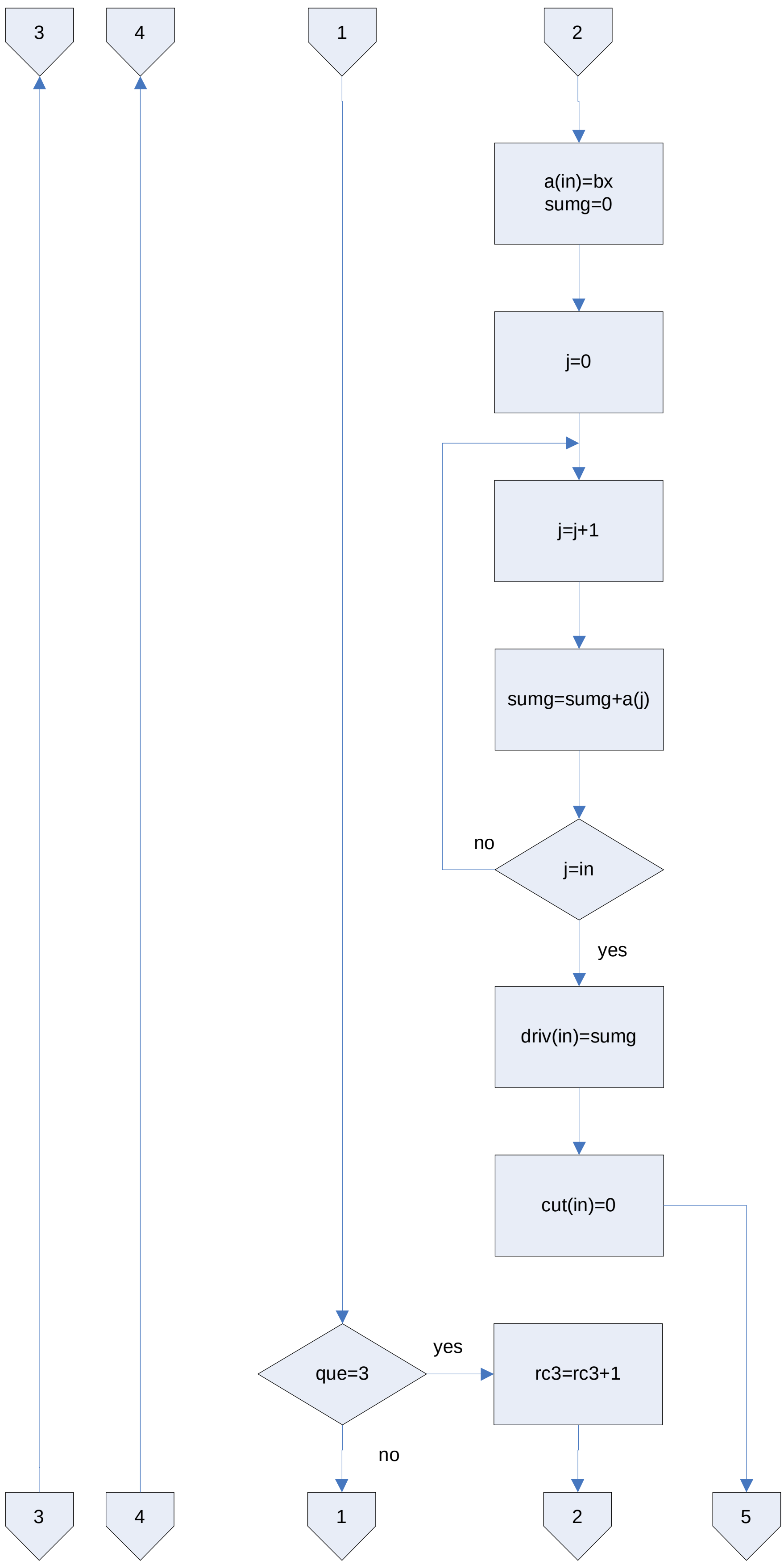


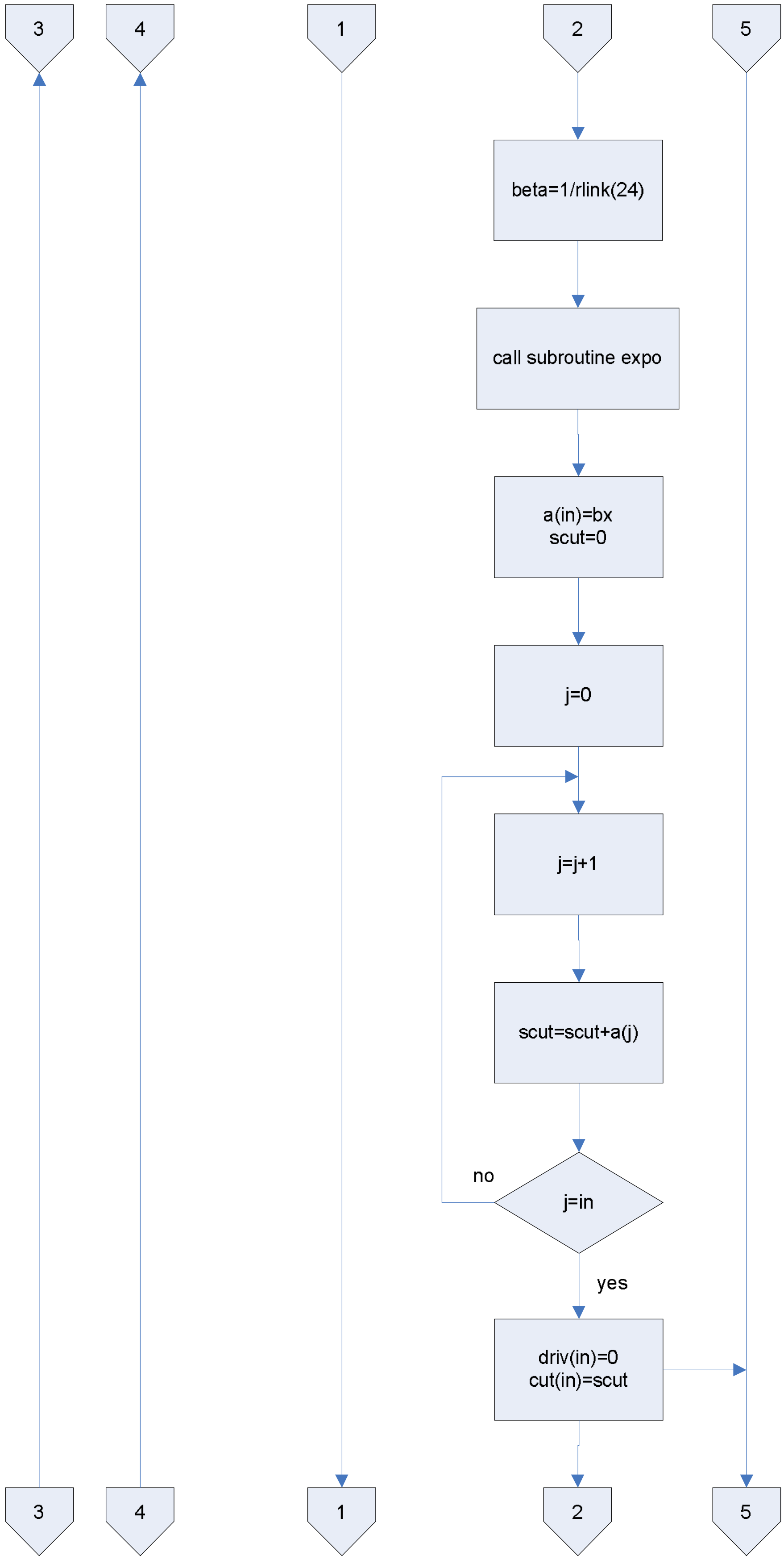
3
5
1
7
6
4
driv(in)=0
cut(in)=scut
end if
rang=rang+a(in)
rang3=rang3+a(in)
yes
o1-j1=1
no
delay=0
drive=0
k1=0
k1=k1+1
yes
cut(k1)=0
sumwa(k1)=0
no
3
8
9
1
10
4

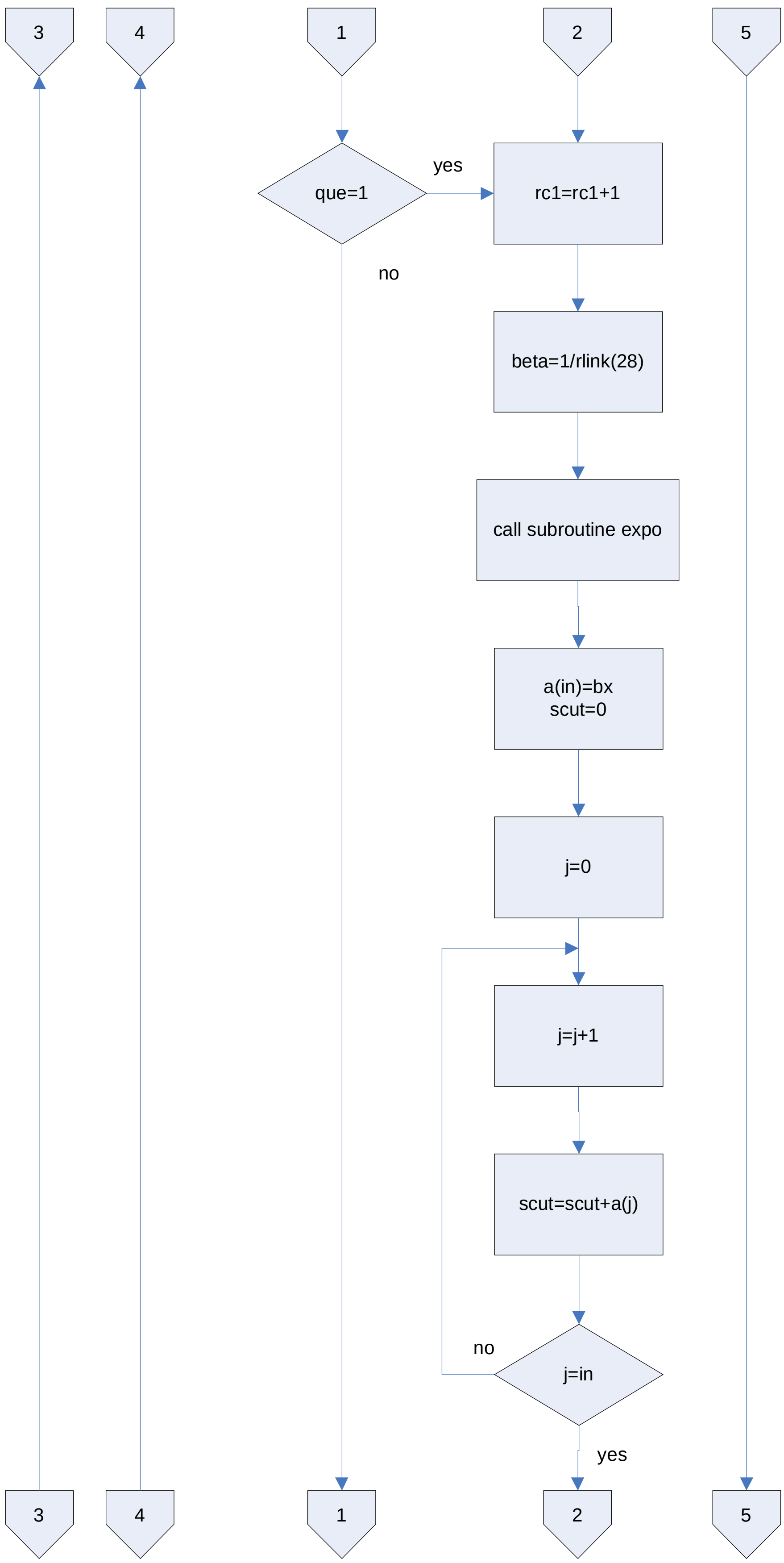

3
8
9
1
10
4
cut(k1)>0
yes
sumwa(k1)=rang1-cut(k1)
no
end if
delay=delay+sumwa(k1)
drive=drive+driv(k1)
no
k1=in
yes
drive=drive+add*a(j1+1)
redn=rc1+rc2+rc4
delay3=delay/redn
g=2*gn/rang3
red=3*redn/rang3
wait=rang3
drive1=cdrive+drive
delay1=cdelay+delay
redn1=credn+redn
gn1=cgn+gn
3
8
1
4

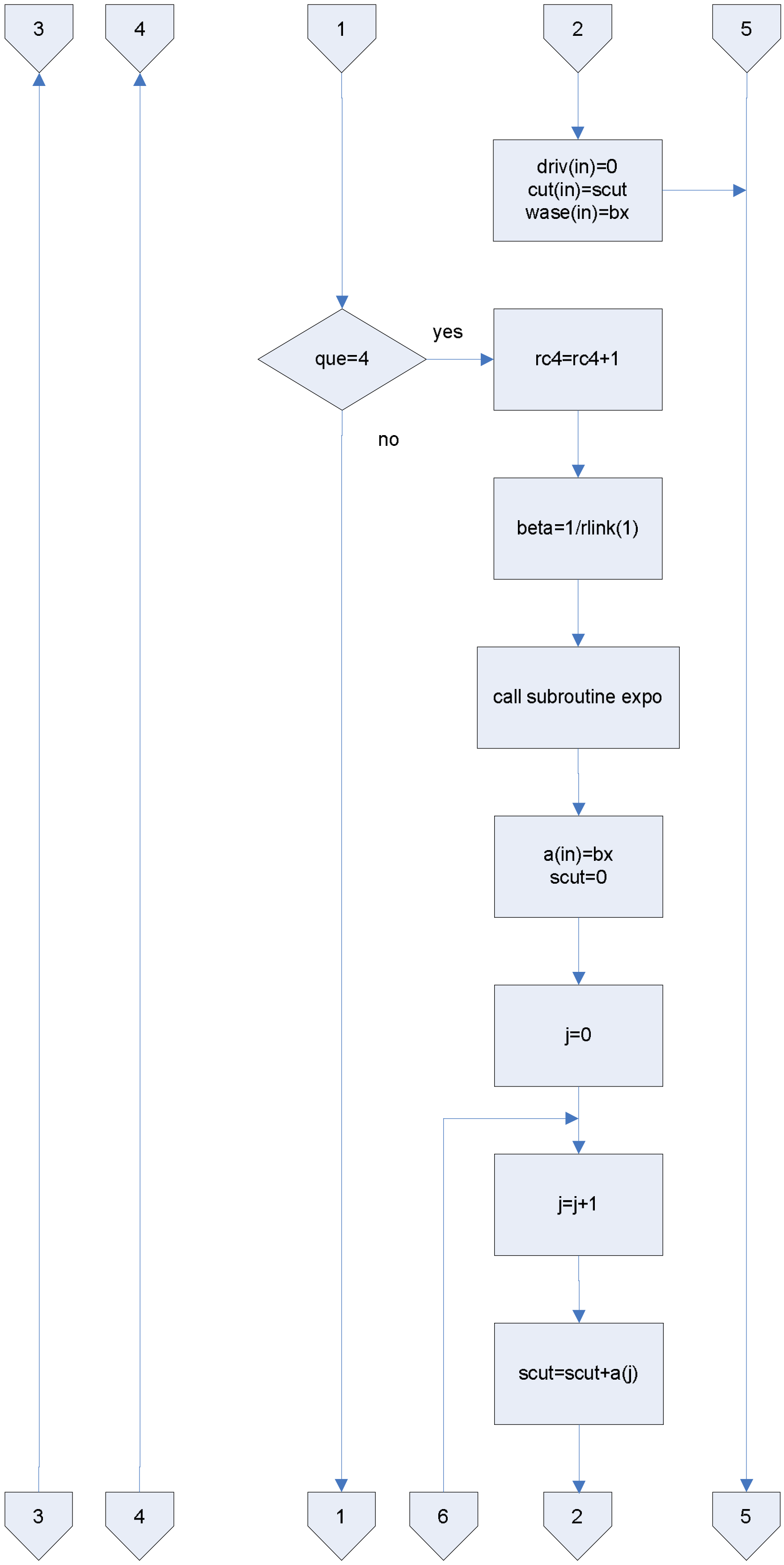

3
8
1
4
cost=(delay1*gn1*(redn+gn))/(100*redn1*gn*drive1)
call subroutine fuzzy
degree=mu
no
degree=1
yes
ap3(v1)=wait
v1=v1+1
yes
rang≥1800
stop
no
gn=rc2
add=rc2
cdrive=drive1
cdelay=delay1
credn=redn1
cgn=gn1
rc3=0
rang4=0
s1=o1
in=0
3
1

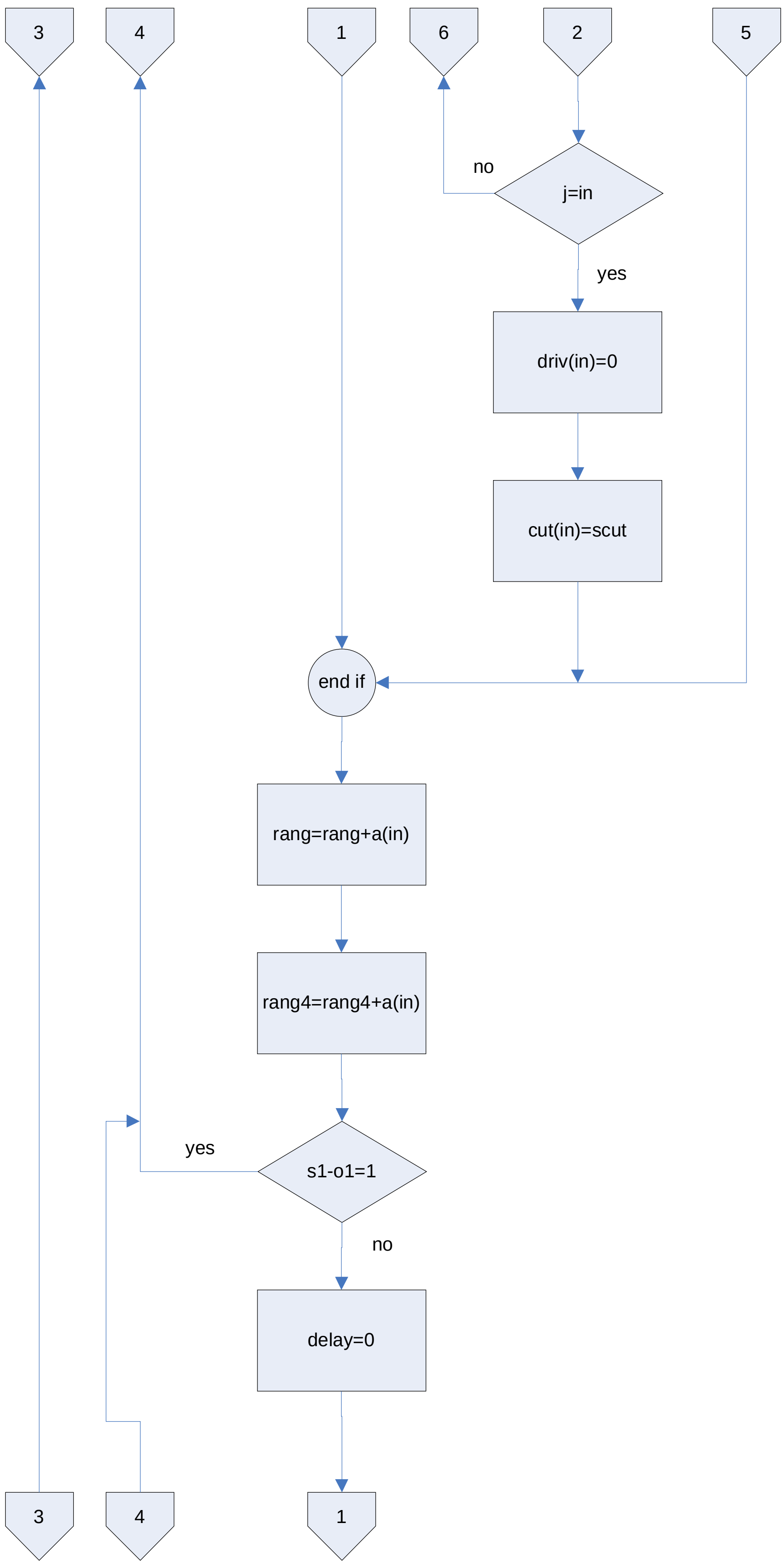

3
1
s1=s1+1
in=in+1
p1=rinten(28)/(rinten(28)+rinten(29)+rinten(24)+rinten(1))
p2=rinten(29)/(rinten(28)+rinten(29)+rinten(24)+rinten(1))
p3=rinten(24)/(rinten(28)+rinten(29)+rinten(24)+rinten(1))
p4=rinten(1)/(rinten(28)+rinten(29)+rinten(24)+rinten(1))
call subroutine allocate
que=q
que=2
yes
no
gn=gn+1
beta=1/rlink(29)
call subroutine expo
3
4
1
2

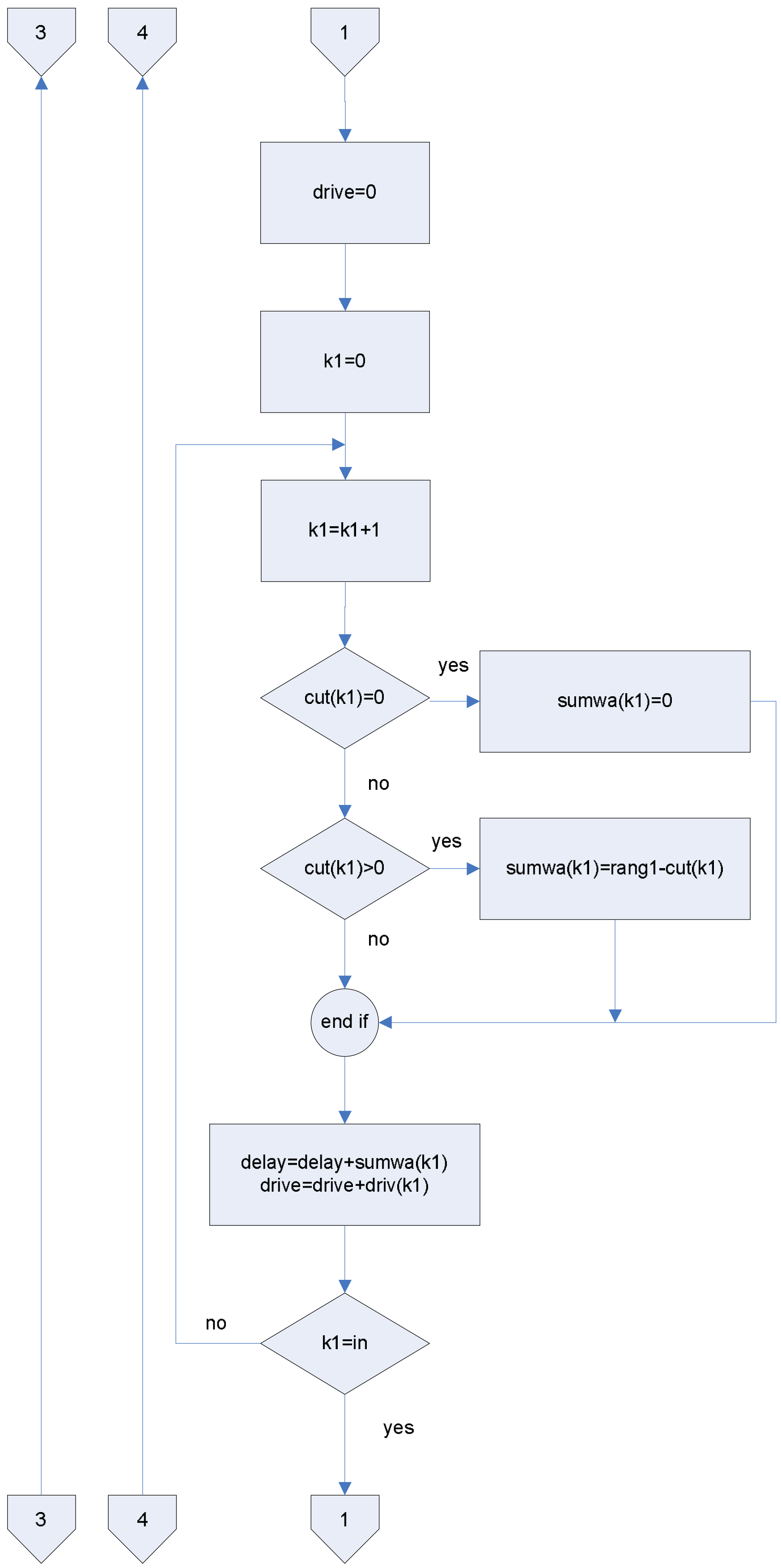

3
4
1
2
a(in)=bx
sumg=0
j=0
j=j+1
sumg=sumg+a(j)
no
j=in
yes
driv(in)=sumg
cut(in)=0
que=3
yes
rc3=rc3+1
no
3
4
1
2
5

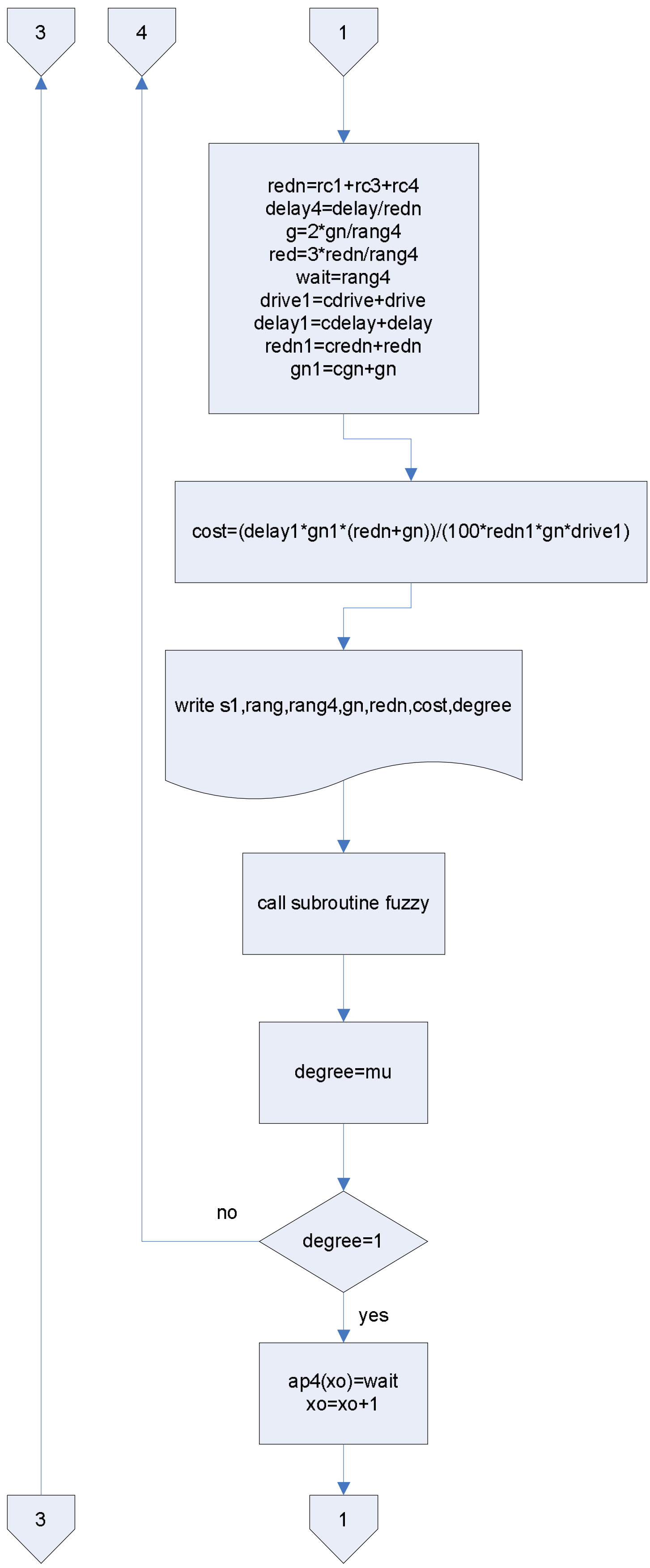

3
4
1
2
5
beta=1/rlink(24)
call subroutine expo
a(in)=bx
scut=0
j=0
j=j+1
scut=scut+a(j)
no
j=in
yes
driv(in)=0
cut(in)=scut
3
4
1
2
5


3
4
1
2
5
que=1
yes
no
rc1=rc1+1
beta=1/rlink(28)
call subroutine expo
a(in)=bx
scut=0
j=0
j=j+1
scut=scut+a(j)
no
j=in
yes
3
4
1
2
5

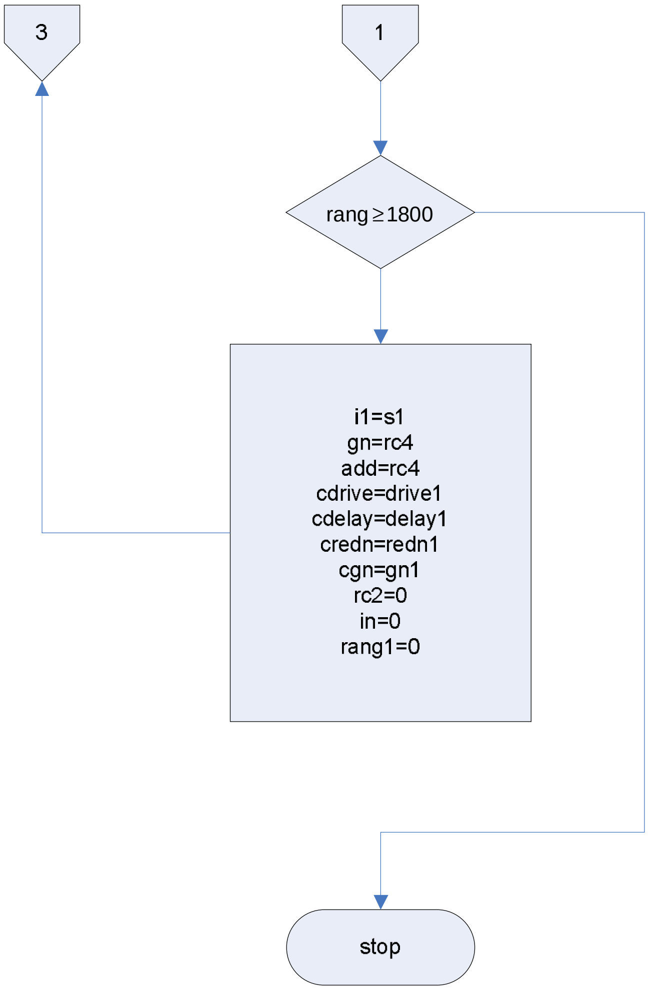

3
4
1
2
5
driv(in)=0
cut(in)=scut
wase(in)=bx
que=4
yes
no
rc4=rc4+1
beta=1/rlink(1)
call subroutine expo
a(in)=bx
scut=0
j=0
j=j+1
scut=scut+a(j)
3
4
1
6
2
5


3
4
1
6
2
5
no
j=in
yes
driv(in)=0
cut(in)=scut
end if
rang=rang+a(in)
rang4=rang4+a(in)
yes
s1-o1=1
no
delay=0
3
4
1


3
4
1
drive=0
k1=0
k1=k1+1
cut(k1)=0
yes
sumwa(k1)=0
no
cut(k1)>0
yes
sumwa(k1)=rang1-cut(k1)
no
end if
delay=delay+sumwa(k1)
drive=drive+driv(k1)
no
k1=in
yes
3
4
1

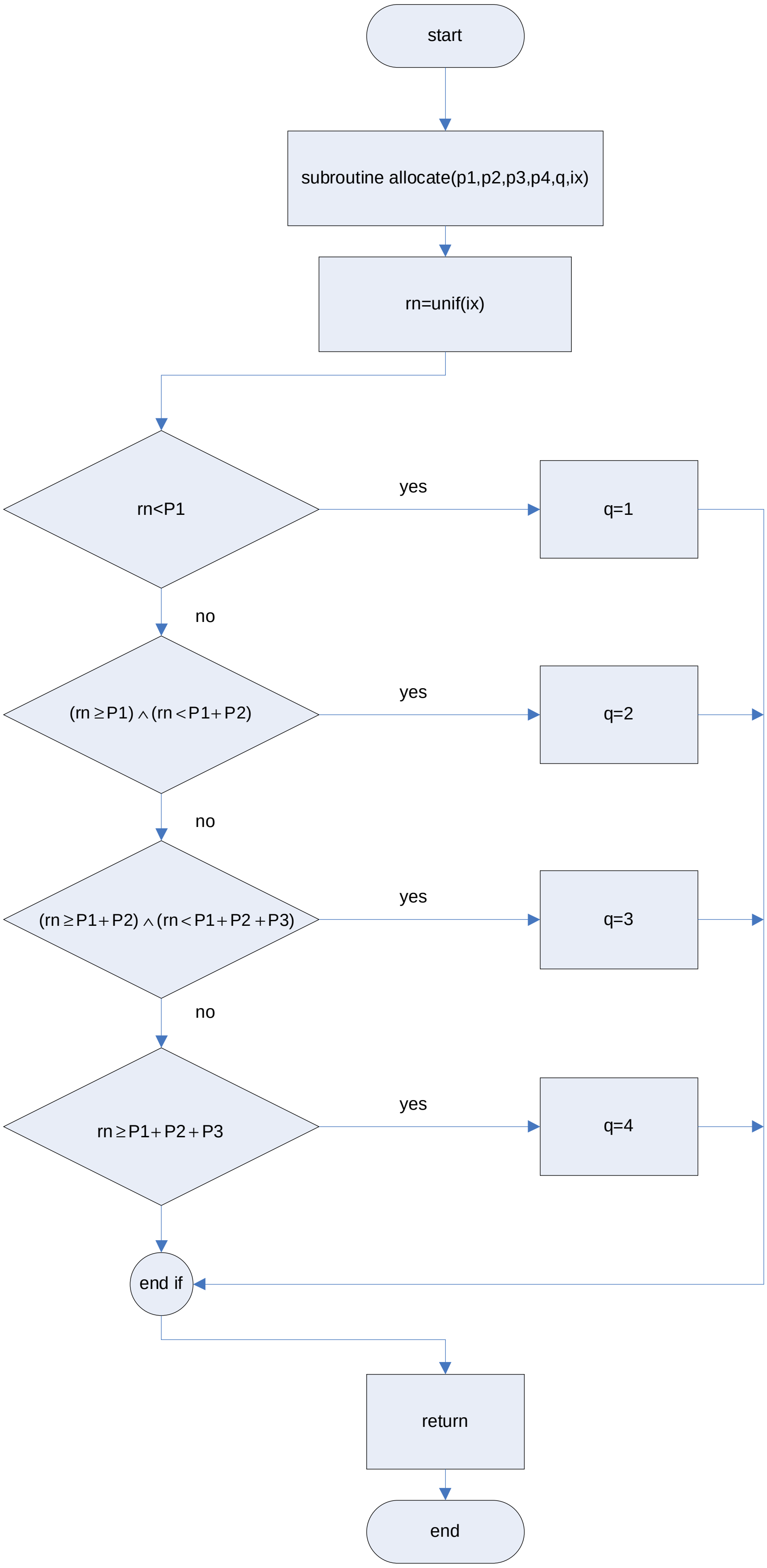


3
4
1
redn=rc1+rc3+rc4
delay4=delay/redn
g=2*gn/rang4
red=3*redn/rang4
wait=rang4
drive1=cdrive+drive
delay1=cdelay+delay
redn1=credn+redn
gn1=cgn+gn
cost=(delay1*gn1*(redn+gn))/(100*redn1*gn*drive1)
write s1,rang,rang4,gn,redn,cost,degree
call subroutine fuzzy
degree=mu
no
degree=1
yes
ap4(xo)=wait
xo=xo+1
3
1

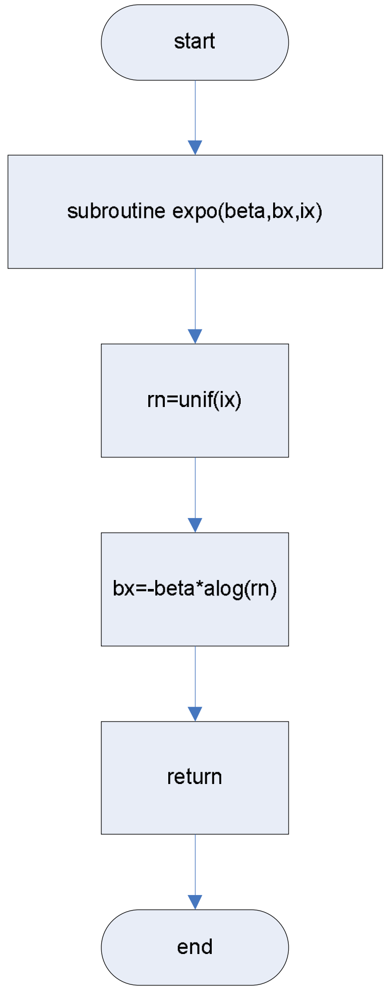

3
1
rang≥1800
i1=s1
gn=rc4
add=rc4
cdrive=drive1
cdelay=delay1
credn=redn1
cgn=gn1
rc2=0
in=0
rang1=0
stop

The flowchart for subprogram

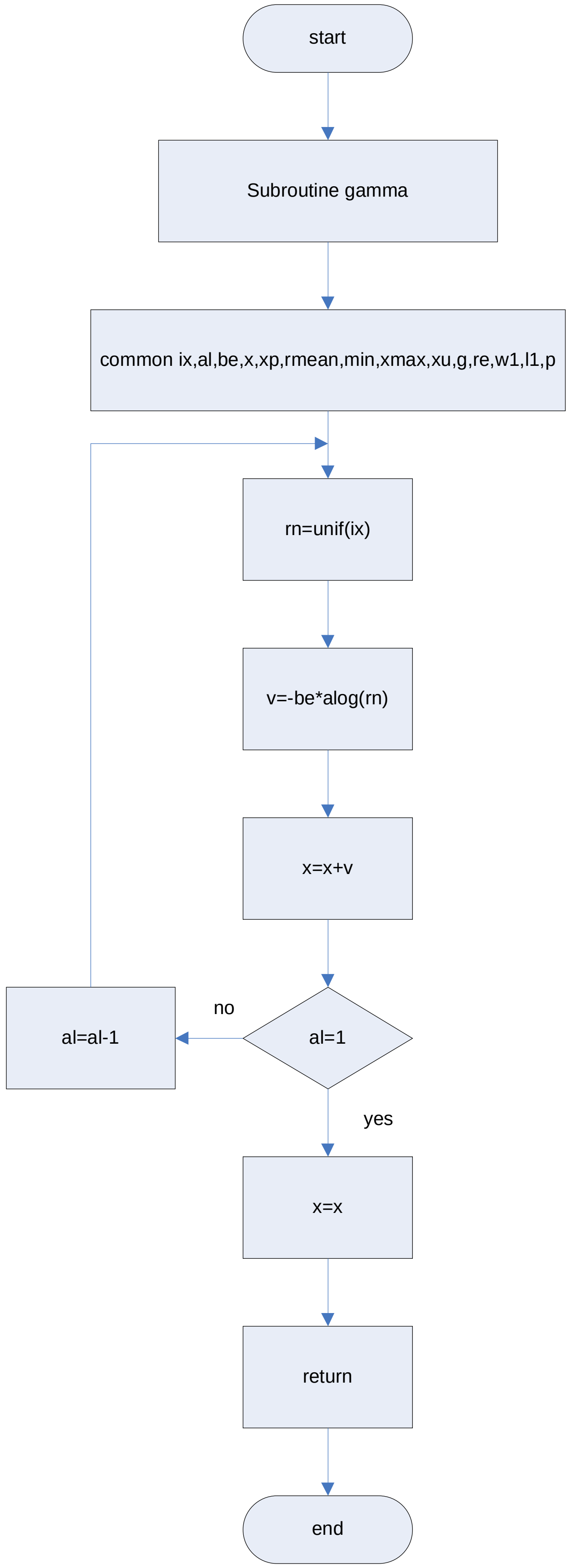
1) Subroutine for allocate car to each branch



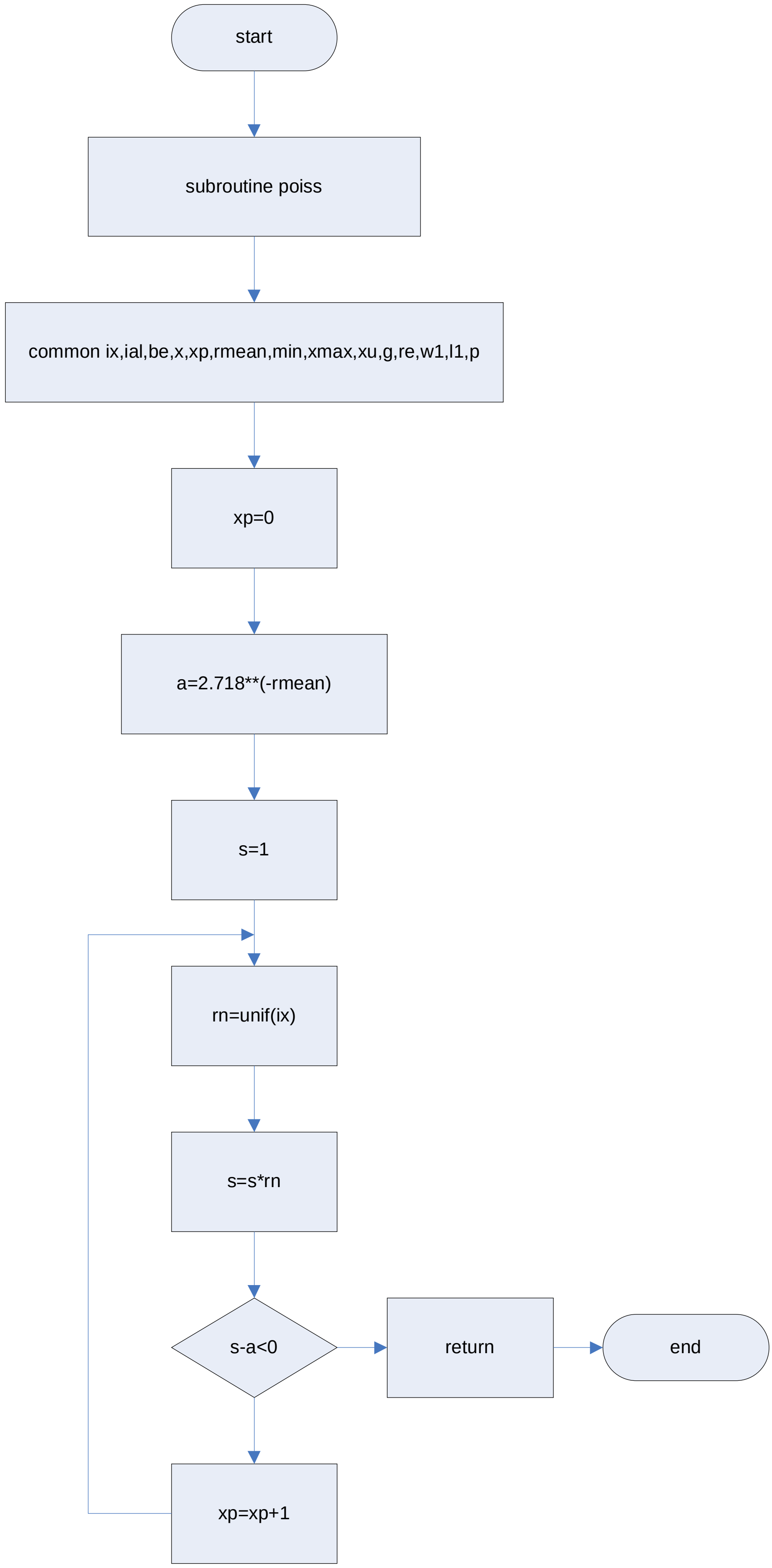
2) Subroutine for generate exponential random variable



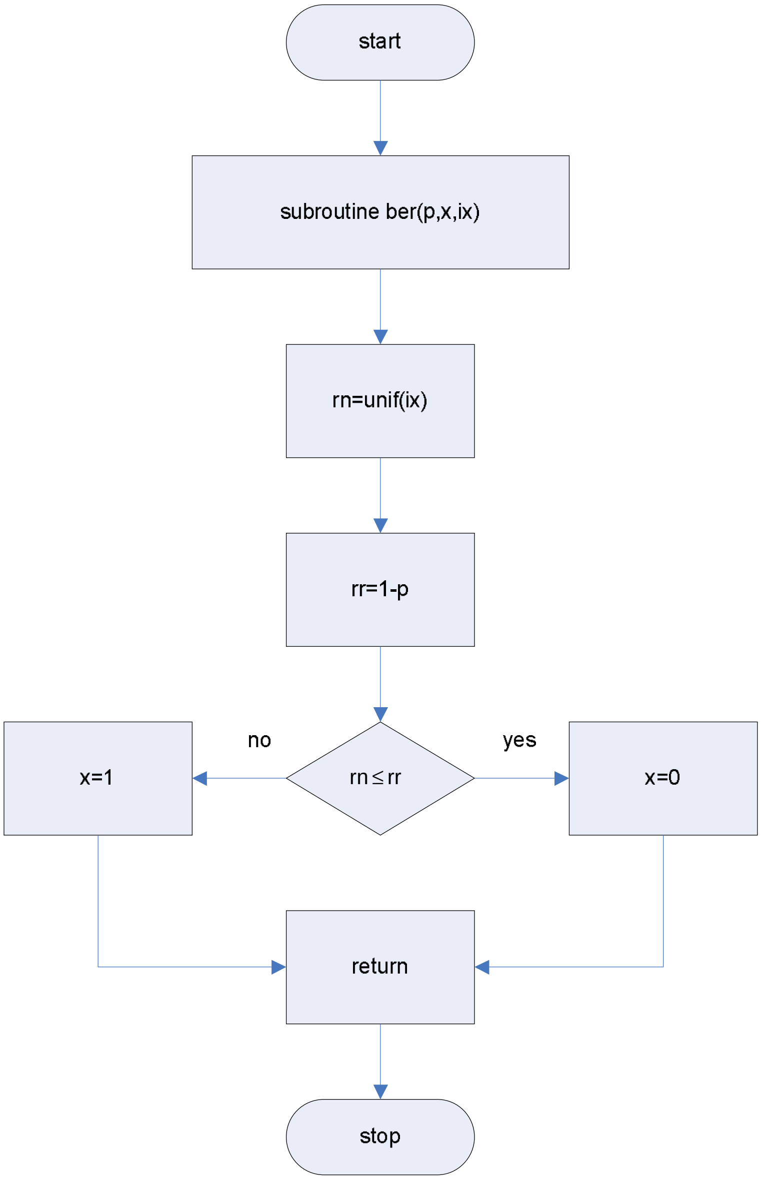
3) Subroutine for generate gamma random variable



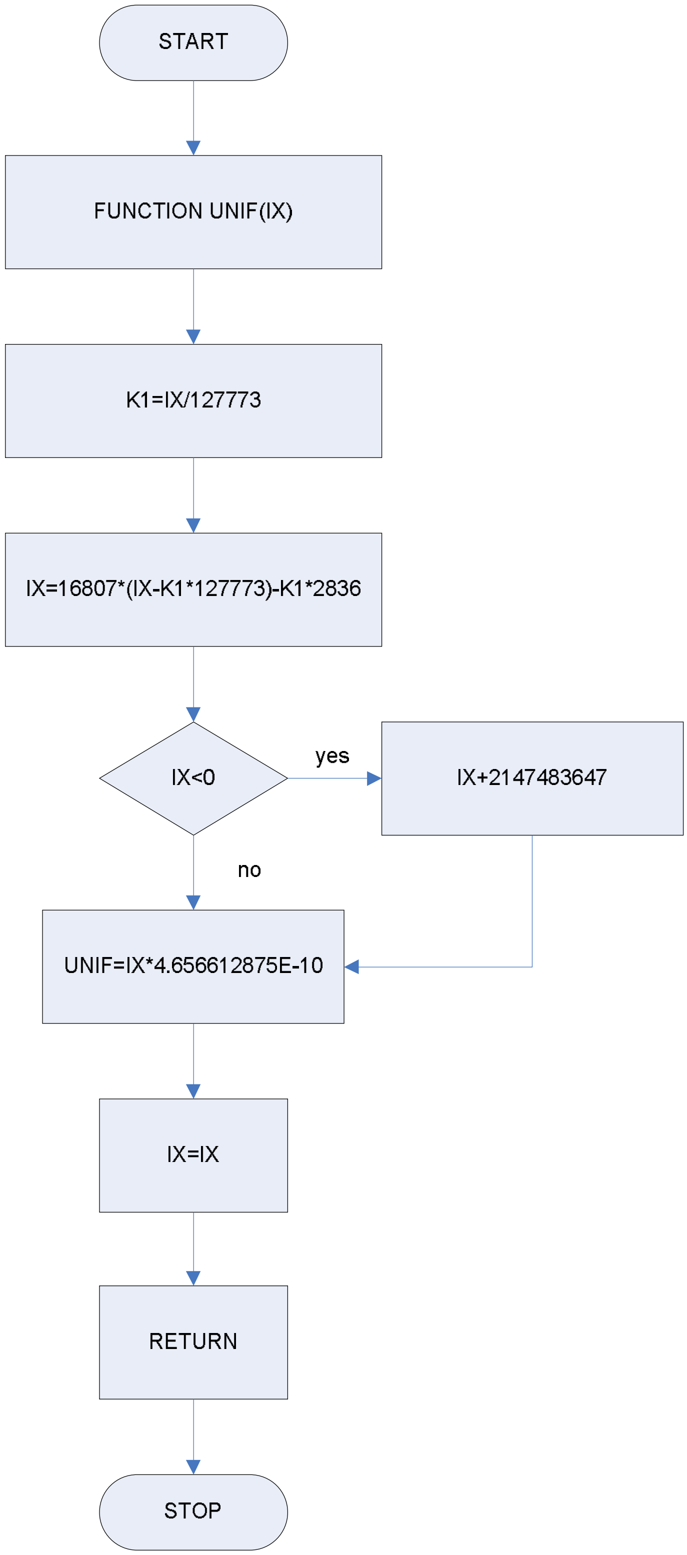
4) Subroutine for generate poisson random variable



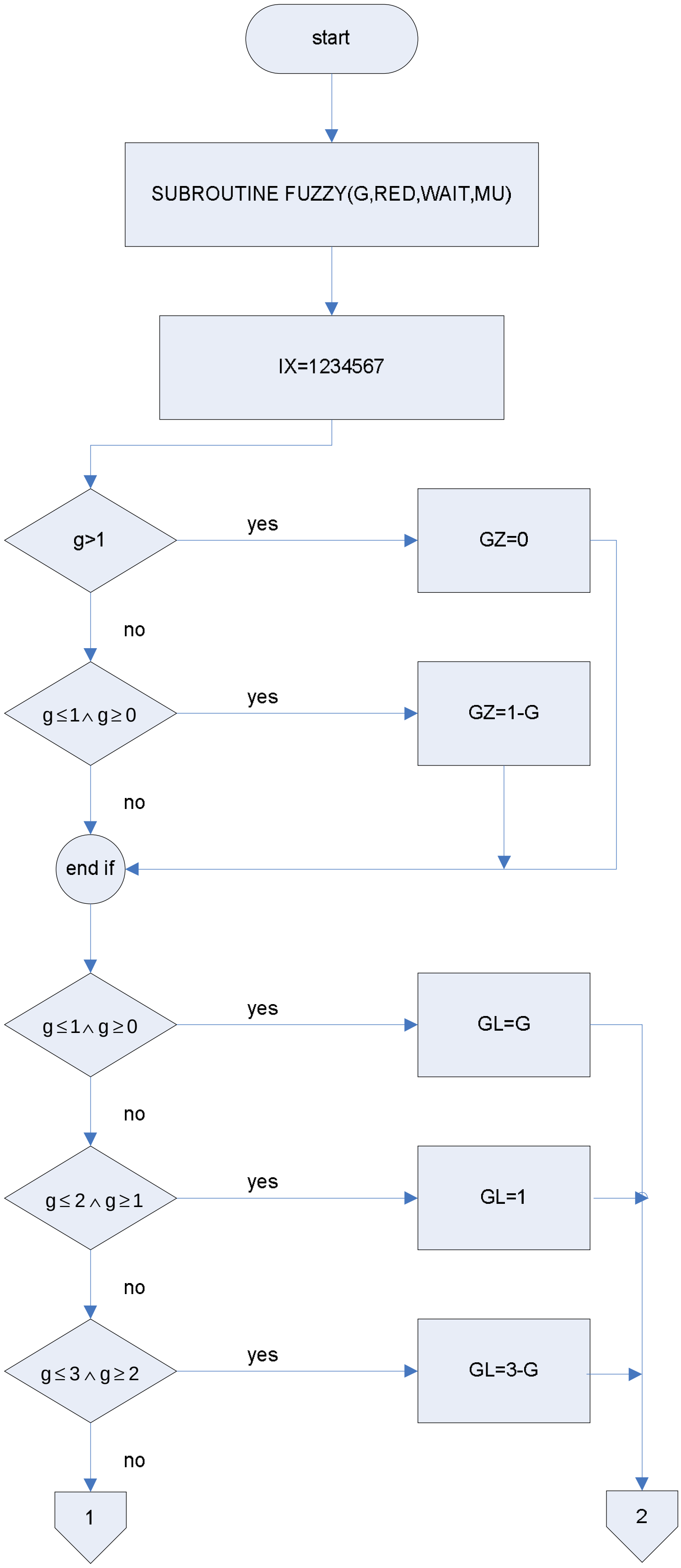
5) Subroutine for generate bernoulie random variable

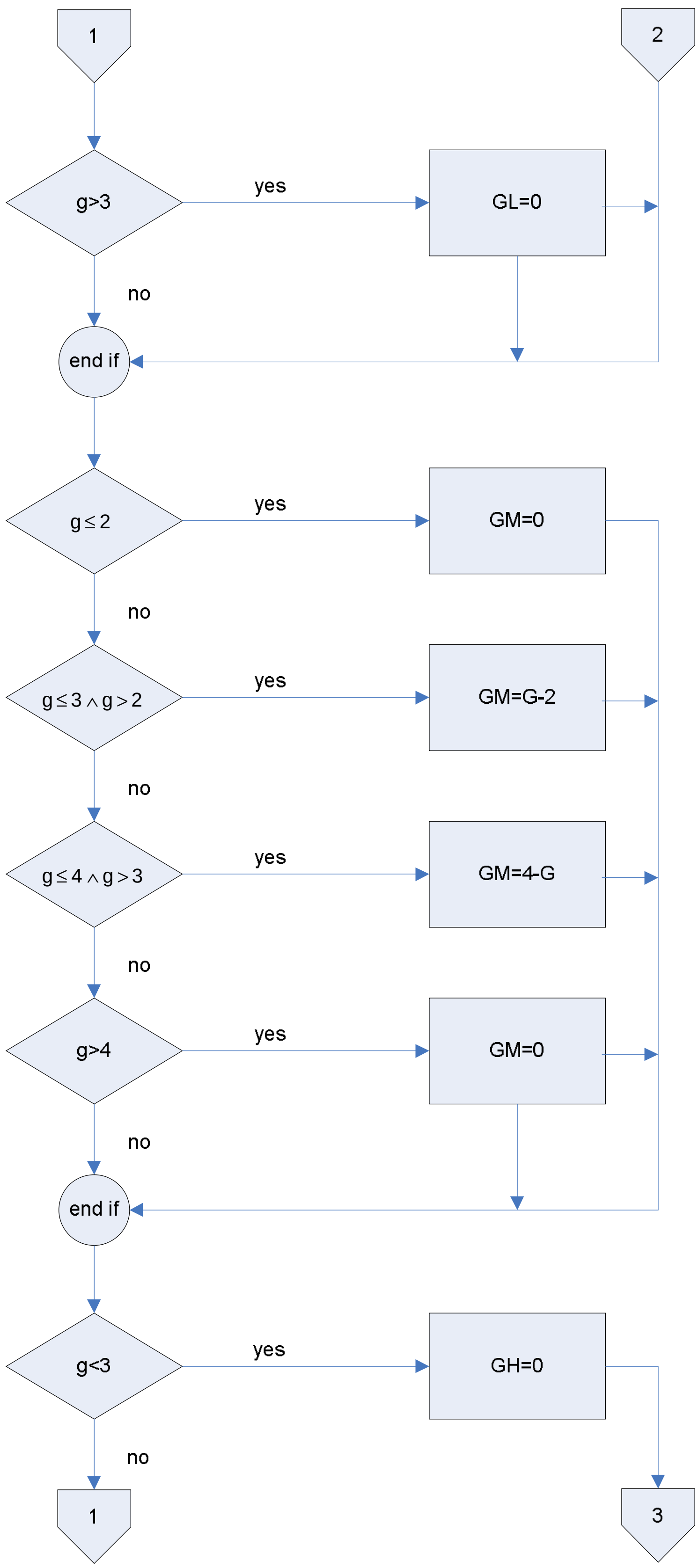


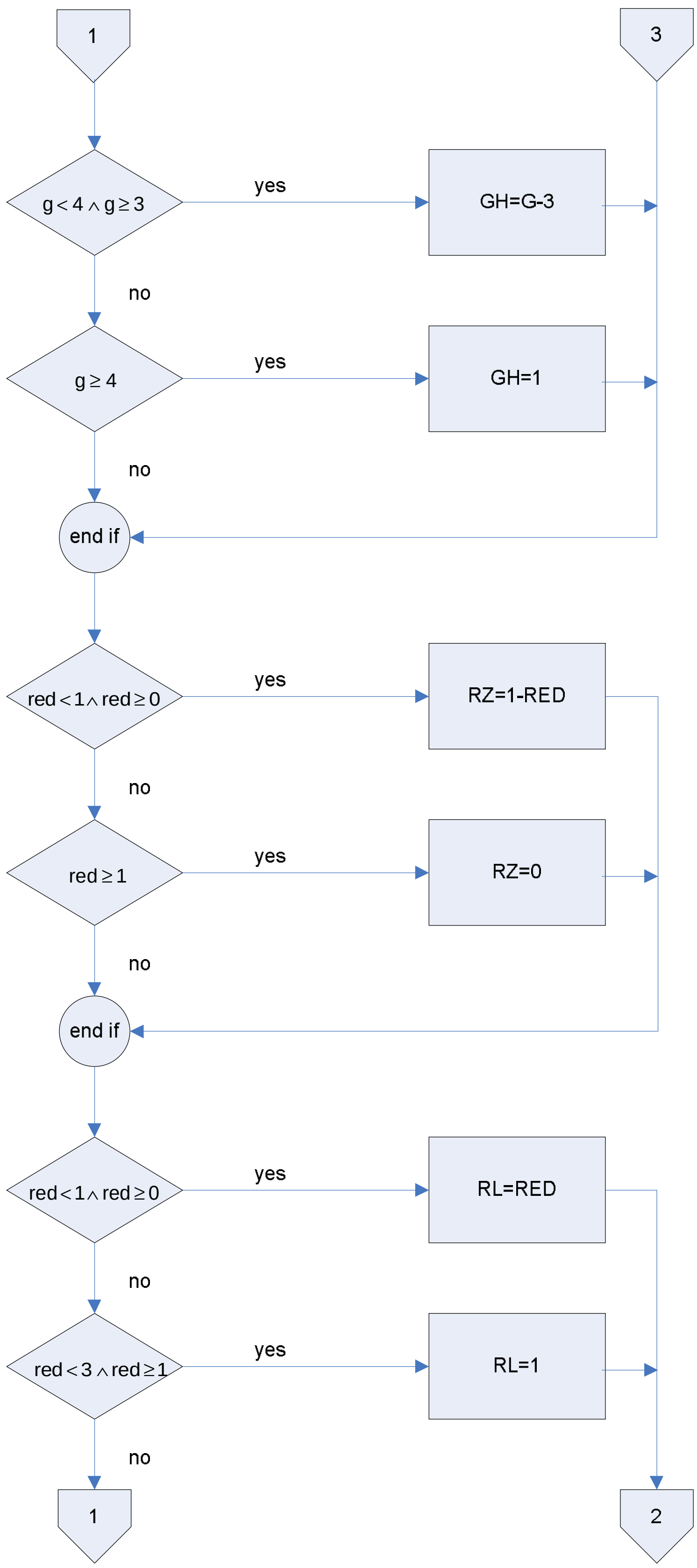
6) Function for generate random number

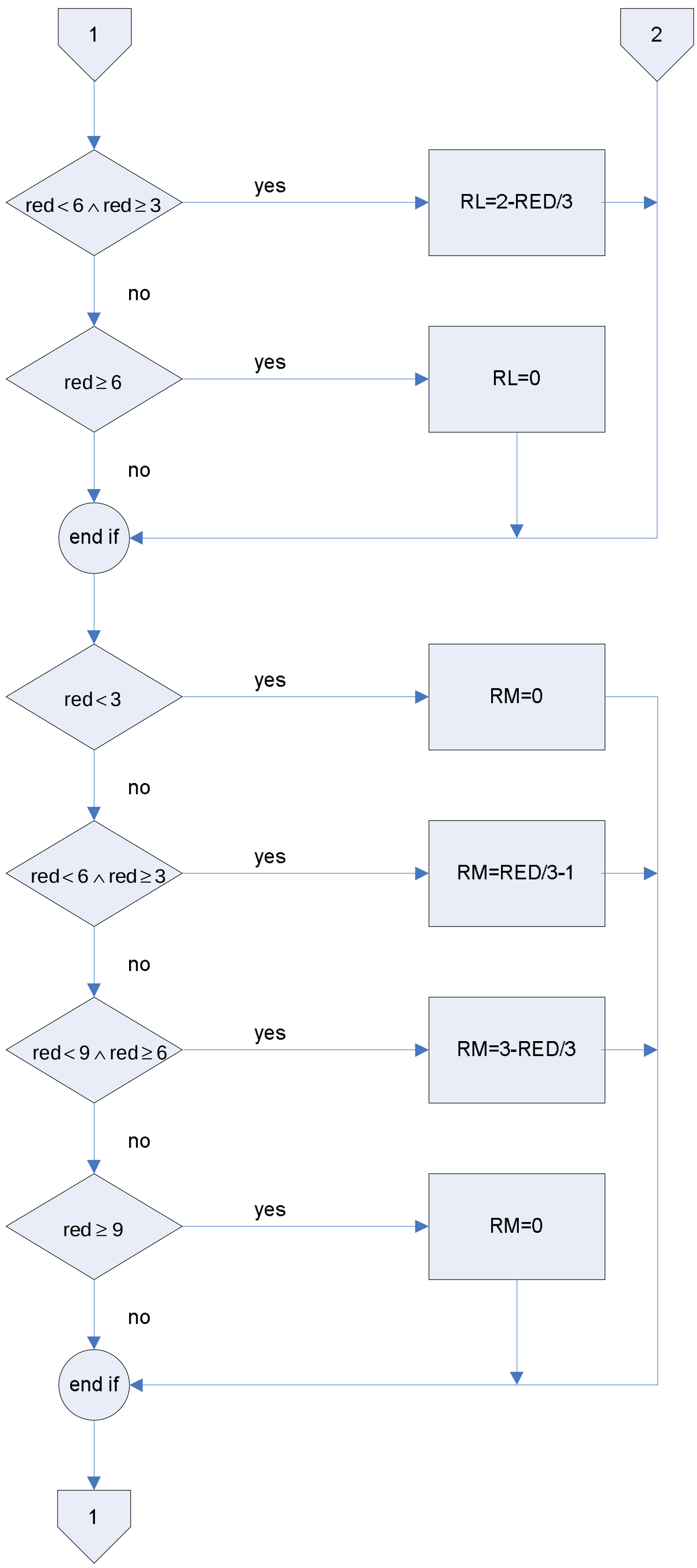


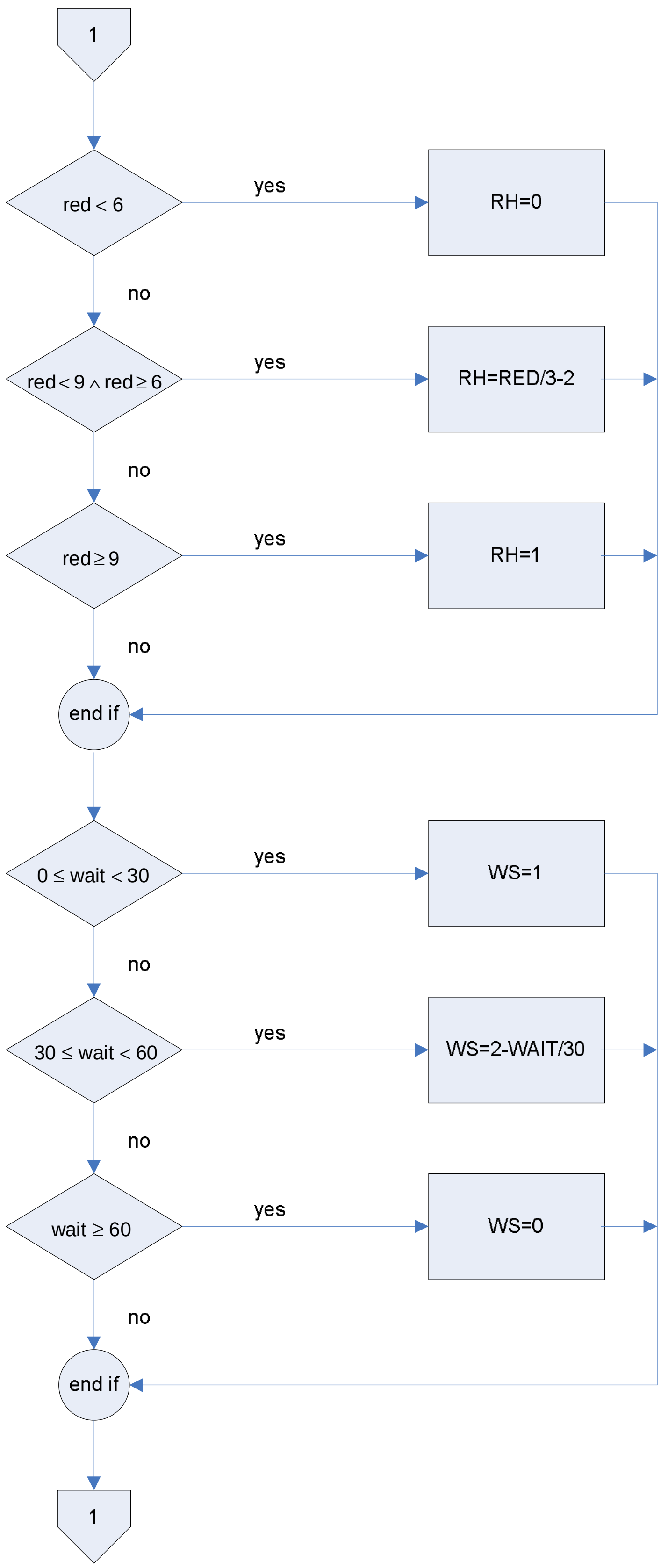
7) Subroutine for fuzzy logic system

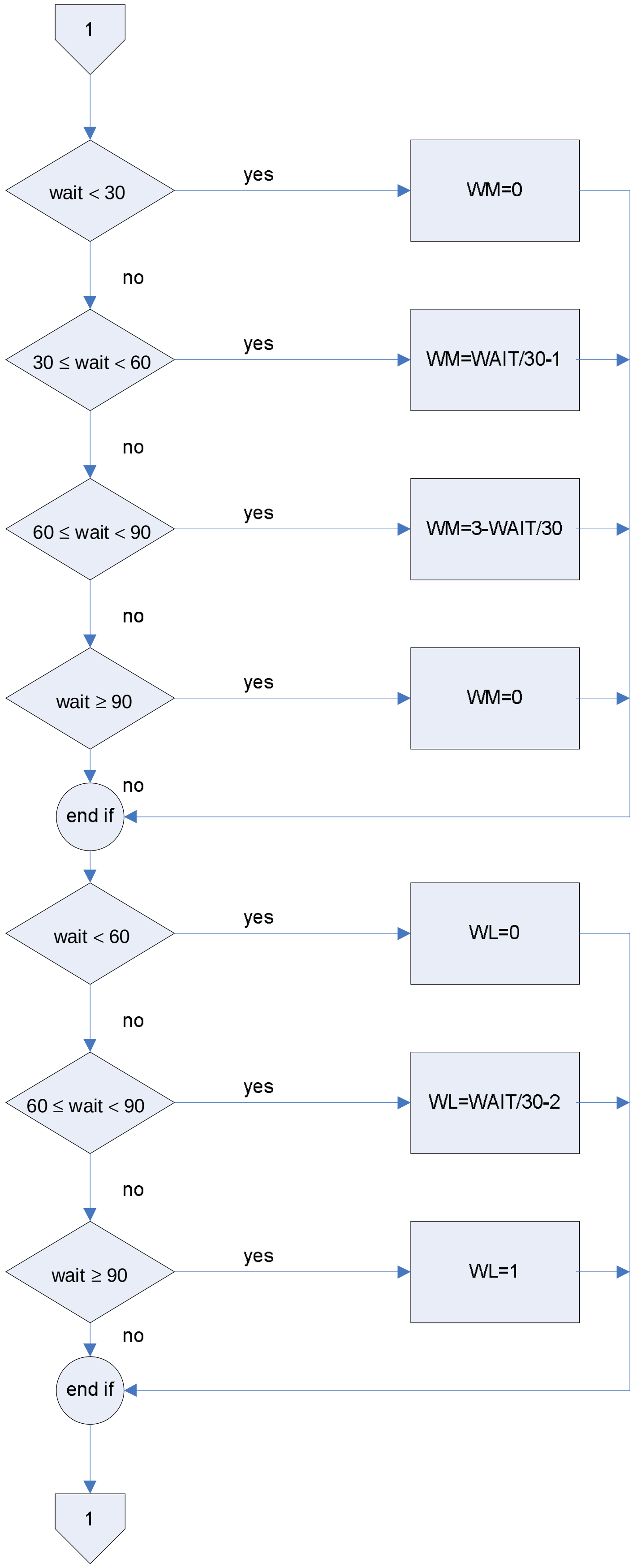


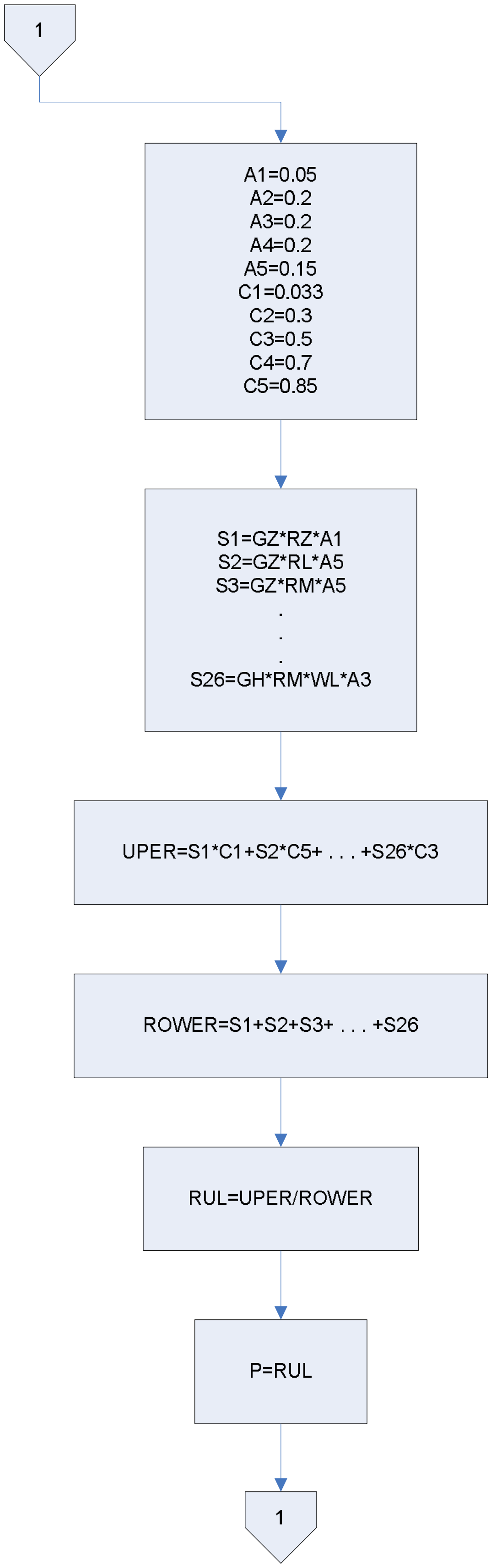

1
2
g>3
yes
GL=0
no
end if
g≤2
yes
GM=0
no
g≤3∧g>2
yes
GM=G-2
no
g≤4∧g>3
yes
GM=4-G
no
g>4
yes
GM=0
no
end if
g<3
yes
GH=0
no
1
3

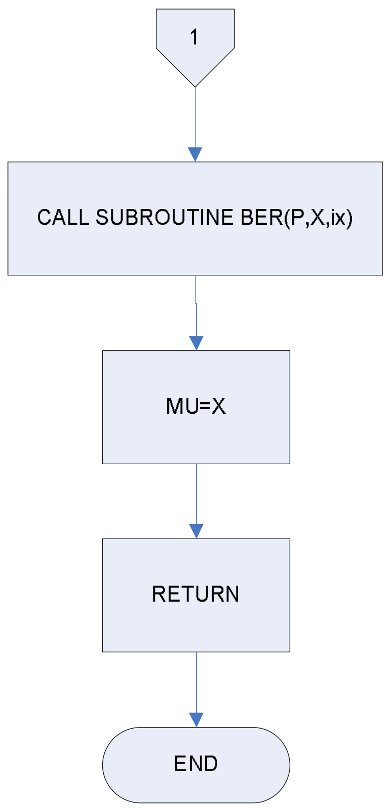

1
3
g < 4 ∧ g ≥ 3
yes
GH=G-3
no
g ≥ 4
yes
GH=1
no
end if
red < 1 ∧ red ≥ 0
yes
RZ=1-RED
no
red ≥ 1
yes
RZ=0
no
end if
red < 1 ∧ red ≥ 0
yes
RL=RED
no
red < 3 ∧ red ≥ 1
yes
RL=1
no
1
2


1
2
red < 6 ∧ red ≥ 3
yes
RL=2-RED/3
no
red ≥ 6
yes
RL=0
no
end if
red < 3
yes
RM=0
no
red < 6 ∧ red ≥ 3
yes
RM=RED/3-1
no
red < 9 ∧ red ≥ 6
yes
RM=3-RED/3
no
red ≥ 9
yes
RM=0
no
end if
1


1
red < 6
yes
RH=0
no
red < 9 ∧ red ≥ 6
yes
RH=RED/3-2
no
red ≥ 9
yes
RH=1
no
end if
0 ≤ wait < 30
yes
WS=1
no
30 ≤ wait < 60
yes
WS=2-WAIT/30
no
wait ≥ 60
yes
WS=0
no
end if
1


1
wait < 30
yes
WM=0
no
30 ≤ wait < 60
yes
WM=WAIT/30-1
no
60 ≤ wait < 90
yes
WM=3-WAIT/30
no
wait ≥ 90
yes
WM=0
no
end if
wait < 60
yes
WL=0
no
60 ≤ wait < 90
yes
WL=WAIT/30-2
no
wait ≥ 90
yes
WL=1
no
end if
1


1
A1=0.05
A2=0.2
A3=0.2
A4=0.2
A5=0.15
C1=0.033
C2=0.3
C3=0.5
C4=0.7
C5=0.85
S1=GZ*RZ*A1
S2=GZ*RL*A5
S3=GZ*RM*A5
.
.
.
S26=GH*RM*WL*A3
UPER=S1*C1+S2*C5+ . . . +S26*C3
ROWER=S1+S2+S3+ . . . +S26
RUL=UPER/ROWER
P=RUL
1


1
CALL SUBROUTINE BER(P,X,ix)
MU=X
RETURN
END

```
C***************************************************
C                      1.  Main Program
C***************************************************
      common ix,al,be,x,xp,rmean,min,xmax,xu,g,re,w1,rl1,p1,p2,p3,p4
     &beta,bx,p,q
      dimension ybar(18),ram(5000,72,10),isum(72),ia(18,72),
     &rmu(72),rmar(18),x2(100,72),iy(100,18),ramda(50,72),da(72),
     &X3(5000,72,10),max(18),z(30),w(72),b(72),r(72),rl(4,4),x1(72),
     &uml(5000,72,10),umu(5000,72,10),rlo(5000,72,10),rinten(72),
     &u(5000,72,10),jy(20,18),count(72),rlink(72),ib(18,72),dan(72)
     &,ymin(72),ymax(72),ax(72),an(72),gy(72),w1(1000),ub1(100),
     &a(1000),para(50,72),rmeanp(72),ub2(100),ub3(100),ch1(100),
     &ch2(100),ch3(100),cp1(100),cp2(100),cp3(100),ap1(100),
     &ap2(100),ap3(100),ap4(100),bar(72),ymean(72),ramd(72),
     &wase(1000),driv(500),cut(500),sumwa(500)
c***************************************************
c   1.1 Program for estimate traffic intensity by EM algorithm
c***************************************************
c    The program read the observe daily data on direct link
c     for 20 day
      open(5,file='input.dat',status='old')
      open(6,file='output.out',status='new')
      do 10 i=1,18
10     read(5,15)(ia(i,j),j=1,72)
15     format(72i1)
      do 20 m=1,20
20     read(5,25)(iy(m,n),n=1,18)
25     format(18i3)
      do 21 i=1,20
      do 21 j=1,18
21     jy(i,j)=iy(i,j)/30
      ix=45673874
5      do 35 j=1,18
      sumy=0
```

```
      do 40 i=1,20
      sumy=sumy+iy(i,j)
40    continue
```

c The program calculate $\bar{Y}_i = \dfrac{\sum_{k=1}^{20} Y_i^{(k)}}{20}$

```
      ybar(j)=sumy/20
      ymean(j)=ybar(j)/30
35    continue
      ix=45673874
      do 30 i=1,72
      x=0.0
      al=80.0
      be=2.0
      zi=i
      ri=i
```

c The program let positive mean population of number of cars

c that travel on direct route on traffic network

c m **=** $(m_1, \mathbf{L}, m_{72})$ **; arbitrary.**

```
      call gamma
      rmu(i)=x
30    continue
      rmu(1)=ybar(1)/30
      rmu(3)=ybar(7)/30
      rmu(5)=ybar(9)/30
      rmu(9)=ybar(2)/30
      rmu(11)=ybar(8)/30
      rmu(13)=ybar(10)/30
      rmu(17)=ybar(3)/30
      rmu(21)=ybar(17)/30
      rmu(22)=ybar(15)/30
      rmu(24)=ybar(4)/30
      rmu(28)=ybar(18)/30
      rmu(29)=ybar(16)/30
      rmu(31)=ybar(5)/30
```

```
      rmu(36)=ybar(13)/30
      rmu(37)=ybar(6)/30
      rmu(42)=ybar(14)/30
      rmu(43)=ybar(11)/30
      rmu(48)=ybar(12)/30
```

c **The program generate daily data on direct links for 100 days**

c $\boldsymbol{Y}^{(1)} \equiv \left(Y_1^{(1)}, Y_2^{(1)}, \mathbf{K}, Y_{18}^{1}\right)$

c $\boldsymbol{Y}^{(2)} \circ \left(Y_1^{(2)}, Y_2^{(2)}, \mathbf{K}, Y_{18}^{(2)}\right)$

c M M

c $\boldsymbol{Y}^{(100)} \circ \left(Y_1^{(100)}, Y_2^{(100)}, \mathbf{K}, Y_{18}^{(100)}\right)$

c **Calculate** $\bar{Y}_i = \dfrac{\sum_{k=1}^{20} Y_i^{(k)}}{100}$

```
      do 600 n1=1,50
      do 605 k1=1,18
      gys=0
```

c **The program generate** X_j **from Poisson distribution**

c **with parameter** m_j, 1,2,$\mathbf{K}$,72 **for 100 day**

```
      do 610 l1=1,100
      rv=l1
      rk=k1
      rn=n1
      rmean=ybar(k1)/30
      call poiss
      gy(k1)=xp
      gys=gys+gy(k1)
610   continue
      bar(k1)=gys/100
605   continue
```

c **The program calculate** m **by** $\hat{m}=(\hat{m}_1,\hat{m}_2,\ldots,\hat{m}_{72})^{/}$ **based on**

c **applied algorithm**

$$m_{\mathrm{j}} \leftarrow \frac{m_j}{\sum_{i=1}^{18} a_{ij}} \sum_{i=1}^{18} \frac{a_{ij}\overline{Y}_i}{\sum_{k=1}^{72} a_{ik} m_k}$$

```
      do 45 j=1,72
      isuma=0
      do 50 i=1,18
50    isuma=isuma+ia(i,j)
      isum(j)=isuma
45    continue
      do 615 t1=1,1000
      do 55 i=1,18
      sumar=0
      do 60 j=1,72
      ri=i
      rj=j
      rt=t1
60    sumar=sumar+ia(i,j)*rmu(j)
      rmar(i)=sumar
55    continue
      do 65 j=1,72
      sumd=0
      sumtest=0
      do 70 i=1,18
      rj=j
      ri=i
      ratio=bar(i)/rmar(i)
      rmuti=ia(i,j)*ratio
      sumtest=sumtest+rmuti
70      devide=sumtest/isum(j)
      ramda(t1,j)=rmu(j)*devide
```

```
c   The program calculate m̂ 50 times to get m̂(1), m̂(2), ..., m̂(50)
      para(n1,j)=ramda(t1,j)
      rmu(j)=ramda(t1,j)
65    continue
615   continue
600   continue
      do 620 j1=1,72
      sump=0
      rj=j1
      do 625 n1=1,50
      sump=sump+para(n1,j1)
625   continue
c   The program calculate mean vector ; m̄̂ = 1/50 Σ_{k=1}^{50} m̂(k) based
c    on  50 estimations. Then m̄̂ is the unbiased estimator of m ,
c    route count.
      rmeanp(j1)=sump/50
c****************************************************
c 1.2  Program for estimate traffic intensity by Gibb sampling
c****************************************************
      t=0.0
      s=0.0
105   t=t+1.0
      s=s+1.0
c   The program generate 10 vectors X from 72 independent
c    Poisson distributions with parameter vector m
      do 111 j=1,10
      do 110 i=1,72
      rj=j
      ri=i
      if(t.eq.1.0)then
      ram(t,i,j)=rmeanp(i)
      x3(t-1,i,j)=rmeanp(i)
      if(rmeanp(i).le.103)then
```

In the comment lines: $\hat{m}^{(1)}, \hat{m}^{(2)}, \ldots, \hat{m}^{(50)}$; $\bar{\hat{m}} = \frac{1}{50}\sum_{k=1}^{50} \hat{m}^{(k)}$.

```
      rmean=ram(t,i,j)
      else if(rmeanp(i).gt.103)then
      rmean=103
      end if
      call poiss
      x3(t,i,j)=xp
      else if(t.gt.1)then
```

c **The program draw sample value of 10 parameter vectors** l

c **from 72 conditionally independent posterior distributions,**

c $p(l_j | X_j)$**, that is Gamma distribution with shape parameter**

c $X_j + 1$ **and scale parameter 1;** $j = 1, 2, \mathbf{K}, 72$**.**

```
      al=int(abs(x3(t-1,i,j))+1)
      be=1.0
      x=0.0
      call gamma
      ram(t,i,j)=x
      rmean=ram(t,i,j)
      if(x.le.103)then
      rmean=ram(t,i,j)
      else if(x.gt.103)then
      rmean=103
      end if
      call poiss
      x3(t,i,j)=xp
      end if
```

c **The program draw a candidate** X_j^* **from Poisson distribution**

c **function For each parameter vector** l **at iterationt as below.**

c X_j^* **~ Poisson(** $X_j^* | \boldsymbol{X}_{-j}^{t-1}$) **;**

c **Where** $\boldsymbol{X}_{-j}^{t-1}$ **represents all the element of** X **except** X_j**, at their**

c **current values:**

c $\boldsymbol{X}_{-j}^{t-1}$ **=** $(X_1^t, \mathbf{K}, X_{j-1}^t, X_{j+1}^{t-1}, \mathbf{K}, X_{72}^{t-1})$

c **set** $X_j^t = \begin{cases} X_j^* & \textit{with probability} \quad \min(r,1) \\ X_j^{t-1} & \textit{otherwise} \end{cases}$

```
c       r = P(X_j^*)U(X_j^{t-1}) / P(X_j^{t-1})U(X_j^*)
c   where   P(X_j) = e^{-l_j} l_j^{x_j} / x_j! ,   U(X_j) = e^{-m_j} m_j^{x_j} / x_j!
    run=1.0
    mf=int(x3(t,i,j))
    do 175 m=1,mf
175 run=run*rmeanp(i)/m
    u(t,i,j)=run/2.718**rmeanp(i)
    if(t.gt.1)go to 172
    n=0
    k=0
    umu(t,i,j)=1.0
    rlo(t,i,j)=1
    go to 173
172 rlo(t,i,j)=u(t-1,i,j)
    umu(t,i,j)=uml(t-1,i,j)
173 n=0
    ifact=1
    run=1.0
    mf=int(x3(t,i,j))
    do 176 m=1,mf
176 run=run*ram(t,i,j)/m
    uml(t,i,j)=run/2.718**ram(t,i,j)
    p=(u(t,i,j)*umu(t,i,j))/(rlo(t,i,j)*uml(t,i,j))
    if(p.ge.1)then
    un=1
    else if(p.lt.1)then
    call ber(p,x,ix)
    un=x
    end if
    if(un.eq.1)then
    x3(t,i,j)=x3(t,i,j)
    else if(un.eq.0) then
```

```
      x3(t,i,j)=x3(t-1,i,j)
      end if
110   continue
c     The program directly compute the element of Y  by Y = AX
      x3(t,1,j)=x3(t,1,j)+x3(t,6,j)+x3(t,25,j)+x3(t,26,j)
     &+x3(t,27,j)+x3(t,57,j)+x3(t,58,j)+x3(t,69,j)+x3(t,70,j)
      x3(t,9,j)=x3(t,9,j)+x3(t,10,j)+x3(t,14,j)+x3(t,15,j)+
     &x3(t,20,j)+x3(t,33,j)+x3(t,51,j)+x3(t,60,j)+x3(t,61,j)
      x3(t,17,j)=x3(t,17,j)+x3(t,18,j)+x3(t,19,j)+x3(t,23,j)+
     &x3(t,40,j)+x3(t,41,j)+x3(t,54,j)-x3(t,55,j)+x3(t,64,j)+
     &x3(t,67,j)
      x3(t,24,j)=x3(t,10,j)+x3(t,24,j)+x3(t,30,j)+x3(t,33,j)+
     &x3(t,34,j)+x3(t,35,j)+x3(t,46,j)+x3(t,66,j)+x3(t,68,j)
      x3(t,31,j)=x3(t,16,j)+x3(t,18,j)+x3(t,19,j)+x3(t,31,j)+
     &x3(t,32,j)+x3(t,50,j)+x3(t,52,j)+x3(t,54,j)+x3(t,55,j)+
     &x3(t,56,j)+x3(t,62,j)
      x3(t,37,j)=x3(t,2,j)+x3(t,8,j)+x3(t,37,j)+x3(t,38,j)+
     &x3(t,39,j)+x3(t,45,j)+x3(t,46,j)+x3(t,47,j)+x3(t,59,j)+
     &x3(t,71,j)
      x3(t,3,j)=x3(t,2,j)+x3(t,3,j)+x3(t,4,j)+x3(t,8,j)+
     &x3(t,39,j)+x3(t,49,j)+x3(t,51,j)+x3(t,59,j)+x3(t,72,j)
      x3(t,11,j)=x3(t,11,j)+x3(t,12,j)+x3(t,16,j)+x3(t,26,j)+
     &x3(t,44,j)+x3(t,53,j)+x3(t,62,j)+x3(t,69,j)+x3(t,70,j)
      x3(t,5,j)=x3(t,5,j)+x3(t,20,j)+x3(t,33,j)+x3(t,44,j)+
     &x3(t,53,j)+x3(t,66,j)+x3(t,61,j)+x3(t,62,j)
      x3(t,13,j)=x3(t,13,j)+x3(t,27,j)+x3(t,39,j)+x3(t,49,j)+
     &x3(t,57,j)+x3(t,58,j)+x3(t,72,j)
      x3(t,43,j)=x3(t,4,j)+x3(t,19,j)+x3(t,32,j)+x3(t,43,j)+
     &x3(t,55,j)+x3(t,56,j)+x3(t,72,j)
      x3(t,48,j)=x3(t,12,j)+x3(t,26,j)+x3(t,38,j)+x3(t,53,j)+
     &x3(t,68,j)+x3(t,69,j)+x3(t,70,j)+x3(t,71,j)
      x3(t,36,j)=x3(t,8,j)+x3(t,23,j)+x3(t,36,j)+x3(t,47,j)+
     &x3(t,59,j)+x3(t,64,j)+x3(t,67,j)+x3(t,71,j)
      x3(t,42,j)=x3(t,16,j)+x3(t,30,j)+x3(t,42,j)+x3(t,52,j)+
     &x3(t,56,j)+x3(t,62,j)+x3(t,66,j)+x3(t,68,j)
```

```
 x3(t,22,j)=x3(t,7,j)+x3(t,22,j)+x3(t,35,j)+x3(t,46,j)+
&x3(t,58,j)+x3(t,63,j)+x3(t,68,j)+x3(t,70,j)
 x3(t,29,j)=x3(t,15,j)+x3(t,29,j)+x3(t,41,j)+x3(t,51,j)+
&x3(t,55,j)+x3(t,61,j)+x3(t,65,j)+x3(t,65,j)+x3(t,67,j)
 x3(t,21,j)=x3(t,6,j)+x3(t,21,j)+x3(t,34,j)+x3(t,57,j)+
&x3(t,65,j)+x3(t,66,j)+x3(t,69,j)
 x3(t,28,j)=x3(t,14,j)+x3(t,28,j)+x3(t,40,j)+x3(t,54,j)+
&x3(t,60,j)+x3(t,63,j)+x3(t,64,j)
111 continue
 if(t.eq.1.0) go to 105
```

c **The program let** X_{tj}^{k} **be the drawn from 10 parallel sequences**

c **of iteration t of the k**[th] **element of** X

c $(t=1,2,\ldots,n; \quad j=1,2,\ldots,10)$ **, compute** B **and** W**, the between**

c **and within-sequence variances for each k**[th]**:**

c $B=\frac{n}{9}\sum_{j=1}^{10}(\overline{X}_{.j}-\overline{X}_{..})^2$ **, where** $\overline{X}_{.j}=\frac{1}{n}\sum_{i=1}^{n}X_{ij}^{k}$ **,** $\overline{X}_{..}=\frac{1}{10}\sum_{i=1}^{10}\overline{X}_{.j}$

c $W=\frac{1}{10}\sum_{j=1}^{10}S_j^2$ **, where** $S_j^2=\frac{1}{n-1}\sum_{i=1}^{n}(X_{ij}^{k}-\overline{X}_{.j})^2$

c **and** $\hat{R}=\frac{1}{n}(\frac{B}{W}+n-1)$

```
 t=s
 time=t
 do 215 i=1,72
 sb=0
 ssb=0
 ss=0
 do 220 j=1,10
 sw=0
 ssw=0
 do 225 t=1,time
 sw=sw+x3(t,i,j)
225 ssw=ssw+x3(t,i,j)**2
 w(j)=(t*ssw-sw**2)/t*(t-1)
 sb=sb+sw/t
```

```
      ssb=ssb+(sw/t)**2
220   ss=ss+w(j)
      w(i)=ss/10
      b(i)=(t/9)*(ssb-sb**2/10)

215   r(i)=sqrt((b(i)/w(i)+t-1)/t)
      t=s
```

c **The program iterate until $\sqrt{\hat{R}} \rightarrow 1$ for all k^th element.**

```
      do 221 ir=1,72
      if((r(ir).le.0.999.or.r(ir).ge.1.001) goto 105
221   continue
```

c **The program calculate route count for each direct route by**

c $\hat{X}_k = \frac{1}{10}\sum_{j=1}^{10} X_{nj}^{k}$ **,** $k = 1,2,\mathbf{K},72$

c **where** $\hat{X}_k$ **is the estimator of route count for direct route** k^{th}

c X_{nj}^{k} **is the latest draw for parallel** j

```
222   do 226 i=1,72
      sum1=0
      do 231 j=1,10
231   sum1=sum1+x3(t,i,j)
      rlink(i)=sum1/600
      rinten(i)=1800*rlink(i)
226   continue
      write(6,311)rlink(1),rlink(2),rlink(3),rlink(4),rlink(5),rlink(6)
      write(6,3122)rlink(7),rlink(8),rlink(9),rlink(10),rlink(11),
     &rlink(12)
      write(6,314)rlink(13),rlink(14),rlink(15),rlink(16),rlink(17),
     &rlink(18)
      write(6,316)rlink(19),rlink(20),rlink(21),rlink(22),rlink(23),
     &rlink(24)
      write(6,3137)rlink(25),rlink(26),rlink(27),rlink(28),rlink(29),
     &rlink(30)
      write(6,3118)rlink(31),rlink(32),rlink(33),rlink(34),rlink(35),
     &rlink(36)
```

```
      write(6,3119)rlink(37),rlink(38),rlink(39),rlink(40),rlink(41),
     &rlink(42)
      write(6,3111)rlink(43),rlink(44),rlink(45),rlink(46),rlink(47),
     &rlink(48)
      write(6,3112)rlink(49),rlink(50),rlink(51),rlink(52),rlink(53),
     &rlink(54)
      write(6,3114)rlink(55),rlink(56),rlink(57),rlink(58),rlink(59),
     &rlink(60)
      write(6,3116)rlink(61),rlink(62),rlink(63),rlink(64),rlink(65),
     &rlink(66)
      write(6,3127)rlink(67),rlink(68),rlink(69),rlink(70),rlink(71),
     &rlink(72)
311    format(6f10.4)
3122    format(6f10.4)
314    format(6f10.4)
316    format(6f10.4)
3137    format(6f10.4)
3118    format(6f10.4)
3119    format(6f10.4)
3111    format(6f10.4)
3112    format(6f10.4)
3114    format(6f10.4)
3116    format(6f10.4)
3127    format(6f10.4)
```

c

c   1.3 Program for calculate optimal length

c

c   The program set the start phase of traffic signal cycle at the

c   intersection.

```
      t1=1
      c1=1
      d1=1
      i1=0
      gn=0
      o1=0
```

```
      rc2=0
      rc3=0
      rc4=0
      in=0
      rang=0
      rang1=0
650   i1=i1+1
      ri1=i1
      in=in+1
c     The program create cars and find the probability, which is
c     emerged from the calculation of route counts , for each
c     of the created car in order to randomise its moving from each
c     branch of the intersection.
      p1=rinten(11)/(rinten(11)+rinten(9)+rinten(13))
      p2=rinten(9)/(rinten(11)+rinten(9)+rinten(13))
      p3=(0.38*rinten(13))/(rinten(11)+rinten(9)+rinten(13))
      p4=(0.62*rinten(13))/(rinten(11)+rinten(9)+rinten(13))
      call allocate(p1,p2,p3,p4,q,ix)
      que=q
      if(que.eq.1)then
      beta=1/rlink(11)
c     The program generate interarrival time of each car by
c     exponential distribution with parameter beta that is fixed by
c     traffic intensity in the part of input process.
      call expo(beta,bx,ix)
      a(in)=bx
      gn=gn+1
      sumg=0
c     The program compute the important parameter of simulation
c     process, input of fuzzy logic system such as:
c     x1' : number of cars that pass the green light.
c     x1' : number of cars from the branch that are allowed to pass the
c           intersection by the green light.
c     x2' : number of car that stop behind the red light.
```

c x_2' : number of cars from the branch that are prohibited passing

c the intersection by the red light.

c x_3' : the current cycle time.

c x_3' : summation of interarrival time.

```
      do 4 j=1,in
      sumg=sumg+a(j)
4     continue
      driv(in)=sumg
      cut(in)=0
      else if(que.eq.2)then
      beta=1/rlink(9)
      call expo(beta,bx,ix)
      a(in)=bx
      rc2=rc2+1
      scut=0
      do 1 j=1,in
      scut=scut+a(j)
1     continue
      driv(in)=0
      cut(in)=scut
      else if(que.eq.3)then
      beta=1/(0.38*rlink(13))
      call expo(beta,bx,ix)
      a(in)=bx
      rc3=rc3+1
      scut=0
      do 2 j=1,in
      scut=scut+a(j)
2     continue
      driv(in)=0
      cut(in)=scut
      else if(que.eq.4)then
      beta=1/(0.62*rlink(13))
      call expo(beta,bx,ix)
```

```
      a(in)=bx
      rc4=rc4+1
      scut=0
      do 3 j=1,in
      scut=scut+a(j)
 3     continue
      driv(in)=0
      cut(in)=scut
      end if
      rang1=rang1+a(in)
      rang=rang+a(in)
      if(i1-o1.eq.1)go to 650
      delay=0
      drive=0
      do 6 k1=1,in
      if(cut(k1).eq.0)then
      sumwa(k1)=0
      else if(cut(k1).gt.0)then
      sumwa(k1)=rang1-cut(k1)
      end if
      delay=delay+sumwa(k1)
      drive=drive+driv(k1)
 6     continue
      drive=drive+add*a(o1+1)
 645   redn=rc2+rc3+rc4
      g=2*gn/rang1
      red=6*redn/rang1
      wait=rang1
      drive1=cdrive+drive
      delay1=cdelay+delay
      redn1=credn+redn
      gn1=cgn+gn
c     The program caculate the value of the cost function.
      cost=(delay1*gn1*(redn+gn))/(100*redn1*gn*drive1)
c     The program calculate degree of change by using
```

```
c      fuzzy logic system.
       call fuzzy(g,red,wait,mu)
       degree=mu
       if(degree.eq.1) go to 655
       go to 650
c    The program iterate until length of time is complete and all
c     intersections are covered.
655   ub1(t1)=wait
       t1=t1+1
       if(rang.gt.1800) go to 730
       gn=rc2+rc4
       add=rc2+rc4
       cdrive=drive1
       cdelay=delay1
       credn=redn1
       cgn=gn1
       rc1=0
       in=0
       rang2=0
       j1=i1
660   j1=j1+1
       in=in+1
       rj1=j1
       p1=rinten(11)/(rinten(11)+rinten(9)+rinten(13))
       p2=rinten(9)/(rinten(11)+rinten(9)+rinten(13))
       p3=(0.38*rinten(13))/(rinten(11)+rinten(9)+rinten(13))
       p4=(0.62*rinten(13))/(rinten(11)+rinten(9)+rinten(13))
       call allocate(p1,p2,p3,p4,q,ix)
       que=q
       if(que.eq.2)then
       beta=1/rlink(9)
       call expo(beta,bx,ix)
       a(in)=bx
       gn=gn+1
       sumg=0
```

```
      do 8 j=1,in
      sumg=sumg+a(j)
8     continue
      driv(in)=sumg
      cut(in)=0
      else if(que.eq.4) then
      beta=1/(0.62*rlink(13))
      call expo(beta,bx,ix)
      a(in)=bx
      gn=gn+1
      sumg=0
      do 9 j=1,in
      sumg=sumg+a(j)
9     continue
      driv(in)=sumg
      cut(in)=0
      else if(que.eq.1)then
      rc1=rc1+1
      beta=1/rlink(11)
      call expo(beta,bx,ix)
      a(in)=bx
      scut=0
      do 11 j=1,in
      scut=scut+a(j)
11    continue
      driv(in)=0
      cut(in)=scut
      else if (que.eq.3)then
      rc3=rc3+1
      beta=1/(0.38*rlink(13))
      call expo(beta,bx,ix)
      a(in)=bx
      scut=0
      do 31 j=1,in
      scut=scut+a(j)
```

```
31   continue
     driv(in)=0
     cut(in)=scut
     end if
     rang=rang+a(in)
     rang2=rang2+a(in)
     if(j1-i1.eq.1)go to 660
     k1=0
     delay=0
     drive=0
     do 12 k1=1,in
     if(cut(k1).eq.0)then
     sumwa(k1)=0
     else if(cut(k1).gt.0)then
     sumwa(k1)=rang2-cut(k1)
     end if
     delay=delay+sumwa(k1)
     drive=drive+driv(k1)
12   continue
     drive=drive+add*a(i1+1)
680  redn=rc1+rc3
     g=3*gn/rang2
     red=6*redn/rang2
     wait=rang2
     drive1=cdrive+drive
     delay1=cdelay+delay
     redn1=credn+redn
     gn1=cgn+gn
     cost=(delay1*gn1*(redn+gn))/(100*redn1*gn*drive1)
     call fuzzy(g,red,wait,mu)
     degree=mu
     if(degree.eq.1) go to 664
     go to 660
664  ub2(c1)=wait
     c1=c1+1
```

```
      if(rang.gt.1800) go to 730
      gn=rc3
      add=rc3
      cdrive=drive1
      cdelay=delay1
      credn=redn1
      cgn=gn1
      rc2=0
      in=0
      rang3=0
      o1=j1
685   o1=o1+1
      in=in+1
      p1=rinten(11)/(rinten(11)+rinten(9)+rinten(13))
      p2=rinten(9)/(rinten(11)+rinten(9)+rinten(13))
      p3=(0.38*rinten(13))/(rinten(11)+rinten(9)+rinten(13))
      p4=(0.62*rinten(13))/(rinten(11)+rinten(9)+rinten(13))
      call allocate(p1,p2,p3,p4,q,ix)
      que=q
      if(que.eq.3) then
      beta=1/(0.38*rlink(13))
      call expo(beta,bx,ix)
      a(in)=bx
      gn=gn+1
      sumg=0
      do 14 j=1,in
      sumg=sumg+a(j)
14    continue
      driv(in)=sumg
      cut(in)=0
      else if(que.eq.4) then
      beta=1/(0.62*rlink(13))
      call expo(beta,bx,ix)
      a(in)=bx
      gn=gn+1
```

```
      sumg=0
      do 17 j=1,in
      sumg=sumg+a(j)
17    continue
      driv(in)=sumg
      cut(in)=0
      else if(que.eq.2) then
      rc2=rc2+1
      beta=1/rlink(9)
      call expo(beta,bx,ix)
      a(in)=bx
      scut=0
      do 18 j=1,in
      scut=scut+a(j)
18    continue
      driv(in)=0
      cut(in)=scut
      else if(que.eq.1)then
      rc1=rc1+1
      beta=1/rlink(11)
      call expo(beta,bx,ix)
      a(in)=bx
      scut=0
      do 19 j=1,in
      scut=scut+a(j)
19    continue
      driv(in)=0
      cut(in)=scut
      end if
      rang3=rang3+a(in)
      rang=rang+a(in)
      if(o1-j1.eq.1)go to 685
      k1=0
      delay=0
      drive=0
```

```
      do 32 k1=1,in
      if(cut(k1).eq.0)then
      sumwa(k1)=0
      else if(cut(k1).gt.0)then
      sumwa(k1)=rang3-cut(k1)
      end if
      delay=delay+sumwa(k1)
      drive=drive+driv(k1)
32    continue
      drive=drive+add*a(j1+1)
710   redn=rc1+rc2
      g=3*gn/rang3
      red=6*redn/rang3
      wait=rang3
      drive1=cdrive+drive
      delay1=cdelay+delay
      redn1=credn+redn
      gn1=cgn+gn
      cost=(delay1*gn1*(redn+gn))/(100*redn1*gn*drive1)
      call fuzzy(g,red,wait,mu)
      degree=mu
      if(degree.eq.1) go to 720
      go to 685
720   ub3(d1)=wait
      d1=d1+1
      if(rang.gt.1800) go to 730
      i1=o1
      gn=rc1
      add=rc1
      cdrive=drive1
      cdelay=delay1
      credn=redn1
      cgn=gn1
      rc=0
      in=0
```

```
      rc3=0
      rc4=0
      rang1=0
      go to 650
730   e1=1
      i1=0
      g=0
      gn=0
      o1=0
      rc2=0
      rc3=0
      rang1=0
      rang=0
      in=0
735   i1=i1+1
      ri1=i1
      in=in+1
      rin=in
      p1=rinten(3)/(rinten(3)+rinten(31)+rinten(48))
      p2=rinten(31)/(rinten(3)+rinten(31)+rinten(48))
      p3=rinten(48)/(rinten(3)+rinten(31)+rinten(48))
      p4=0
      call allocate(p1,p2,p3,p4,q,ix)
      que=q
      if(que.eq.1) then
      gn=gn+1
      beta=1/rlink(3)
      call expo(beta,bx,ix)
      a(in)=bx
      sumg=0
      do 121 j=1,in
      sumg=sumg+a(j)
121    continue
      driv(in)=sumg
      cut(in)=0
```

```
      else if(que.eq.2) then
      rc2=rc2+1
      beta=1/rlink(31)
      call expo(beta,bx,ix)
      a(in)=bx
       scut=0
      do 24 j=1,in
      scut=scut+a(j)
24    continue
      driv(in)=0
      cut(in)=scut
      else if(que.eq.3)then
      rc3=rc3+1
      beta=1/rlink(48)
      call expo(beta,bx,ix)
      a(in)=bx
       scut=0
      do 26 j=1,in
      scut=scut+a(j)
26    continue
      driv(in)=0
      cut(in)=scut
      end if
      rang1=rang1+a(in)
      rang=rang+a(in)
      if(i1-o1.eq.1)go to 735
      drive=0
      delay=0
750   w1(k1)=sumw
      do 36 k1=1,in
      if(cut(k1).eq.0)then
      sumwa(k1)=0
      else if(cut(k1).gt.0)then
      sumwa(k1)=rang1-cut(k1)
      end if
```

```
      delay=delay+sumwa(k1)
      drive=drive+driv(k1)
36    continue
      drive=drive+add*a(o1+1)
755   redn=rc2+rc3
      delay1=delay/redn
      g=4*gn/rang1
      red=6*redn/rang1
      wait=rang1
      drive1=cdrive+drive
      delay1=cdelay+delay
      redn1=credn+redn
      gn1=cgn+gn
      cost=(delay1*gn1*(redn+gn))/(100*redn1*gn*drive1)
      if(wait.gt.20)go to 760
      go to 735
760   ch1(e1)=wait
      e1=e1+1
      if(rang.ge.1800) go to 789
      f1=1
      z1=1
      gn=rc2
      add=rc2
      cdrive=drive1
      cdelay=delay1
      credn=redn1
      cgn=gn1
      rc1=0
      rang2=0
      j1=i1
      in=0
764   j1=j1+1
      rj1=j1
      in=in+1
      p1=rinten(3)/(rinten(3)+rinten(31)+rinten(48))
```

```
      p2=rinten(31)/(rinten(3)+rinten(31)+rinten(48))
      p3=rinten(48)/(rinten(3)+rinten(31)+rinten(48))
      p4=0
      call allocate(p1,p2,p3,p4,q,ix)
      que=q
      if(que.eq.2) then
      gn=gn+1
      beta=1/rlink(31)
      call expo(beta,bx,ix)
      a(in)=bx
      sumg=0
      do 22 j=1,in
      sumg=sumg+a(j)
22    continue
      driv(in)=sumg
      cut(in)=0
      else if(que.eq.1) then
      rc1=rc1+1
      beta=1/rlink(3)
      call expo(beta,bx,ix)
      a(in)=bx
       scut=0
      do 27 j=1,in
      scut=scut+a(j)
27    continue
      driv(in)=0
      cut(in)=scut
      else if(que.eq.3)then
      rc3=rc3+1
      beta=1/rlink(48)
      call expo(beta,bx,ix)
      a(in)=bx
      scut=0
      do 127 j=1,in
      scut=scut+a(j)
```

```
127    continue
       driv(in)=0
       cut(in)=scut
       end if
       rang=rang+a(in)
       rang2=rang2+a(in)
       if(j1-i1.eq.1)go to 764
       drive=0
       delay=0
       do 46 k1=1,in
       if(cut(k1).eq.0)then
       sumwa(k1)=0
       else if(cut(k1).gt.0)then
       sumwa(k1)=rang1-cut(k1)
       end if
       delay=delay+sumwa(k1)
       drive=drive+driv(k1)
46     continue
       drive=drive+add*a(i1+1)
780    redn=rc1+rc3
       delay2=delay/redn
       g=4*gn/rang2
       red=6*redn/rang2
       wait=rang2
       drive1=cdrive+drive
       delay1=cdelay+delay
       redn1=credn+redn
       gn1=cgn+gn
       cost=(delay1*gn1*(redn+gn))/(100*redn1*gn*drive1)
       if(wait.gt.25) go to 785
       go to 764
785    ch2(f1)=wait
       f1=f1+1
       if(rang.gt.1800) go to 789
       gn=rc3
```

```
      add=rc3
      cdrive=drive1
      cdelay=delay1
      credn=redn1
      cgn=gn1
      rc2=0
      rang3=0
      o1=j1
      in=0
790   o1=o1+1
      in=in+1
      p1=rinten(3)/(rinten(3)+rinten(31)+rinten(48))
      p2=rinten(31)/(rinten(3)+rinten(31)+rinten(48))
      p3=rinten(48)/(rinten(3)+rinten(31)+rinten(48))
      p4=0
      call allocate(p1,p2,p3,p4,q,ix)
      que=q
      if(que.eq.3) then
      gn=gn+1
      beta=rlink(48)
      call expo(beta,bx,ix)
      a(in)=bx
      sumg=0
      do 123 j=1,in
      sumg=sumg+a(j)
123   continue
      driv(in)=sumg
      cut(in)=0
      else if(que.eq.1) then
      rc1=rc1+1
      beta=rlink(3)
      call expo(beta,bx,ix)
      a(in)=bx
      scut=0
      do 28 j=1,in
```

```
      scut=scut+a(j)
28    continue
      driv(in)=0
      cut(in)=scut
      else if(que.eq.2)then
      rc2=rc2+1
      beta=rlink(31)
      call expo(beta,bx,ix)
      a(in)=bx
       scut=0
      do 29 j=1,in
      scut=scut+a(j)
29    continue
      driv(in)=0
      cut(in)=scut
      end if
      rang=rang+a(in)
      rang3=rang3+a(in)
      if(o1-j1.eq.1)go to 790
      drive=0
      delay=0
      do 56 k1=1,in
      if(cut(k1).eq.0)then
      sumwa(k1)=0
      else if(cut(k1).gt.0)then
      sumwa(k1)=rang1-cut(k1)
      end if
      delay=delay+sumwa(k1)
      drive=drive+driv(k1)
56     continue
      drive=drive+add*a(j1+1)
809    redn=rc1+rc2
      delay3=delay/redn
      g=4*gn/rang3
      red=6*redn/rang3
```

```
      wait=rang3
      drive1=cdrive+drive
      delay1=cdelay+delay
      redn1=credn+redn
      gn1=cgn+gn
      cost=(delay1*gn1*(redn+gn))/(100*redn1*gn*drive1)
      if(wait.gt.25)go to 784
      go to 790
784   ch3(z1)=wait
      z1=z1+1
      if(rang.ge.1800)go to 789
      i1=o1
      gn=rc1
      add=rc1
      cdrive=drive1
      cdelay=delay1
      credn=redn1
      cgn=gn1
      rc3=0
      in=0
      rang1=0
      go to 735
789   h1=1
      y1=1
      ai=1
      i1=0
      o1=0
      rc1=0
      rc3=0
      rang=0
      rang1=0
      gn=0
      in=0
794   i1=i1+1
      ri1=i1
```

```
      in=in+1
      p1=rinten(42)/(rinten(42)+rinten(17)+rinten(37))
      p2=rinten(17)/(rinten(42)+rinten(17)+rinten(37))
      p3=rinten(37)/(rinten(42)+rinten(17)+rinten(37))
      p4=0
      call allocate(p1,p2,p3,p4,q,ix)
      que=q
      if(que.eq.2) then
      gn=gn+1
      beta=1/rlink(17)
      call expo(beta,bx,ix)
      a(in)=bx
      sumg=0
      do 61 j=1,in
      sumg=sumg+a(j)
61    continue
      driv(in)=sumg
      cut(in)=0
      else if(que.eq.1) then
      rc1=rc1+1
      beta=1/rlink(42)
      call expo(beta,bx,ix)
      a(in)=bx
       scut=0
      do 64 j=1,in
      scut=scut+a(j)
64    continue
      driv(in)=0
      cut(in)=scut
      else if(que.eq.3)then
      rc3=rc3+1
      beta=1/rlink(37)
      call expo(beta,bx,ix)
      a(in)=bx
       scut=0
```

```
      do 66 j=1,in
      scut=scut+a(j)
66    continue
      driv(in)=0
      cut(in)=scut
      end if
      rang=rang+a(in)
      rang1=rang1+a(in)
      if(i1-o1.eq.1)go to 794
      drive=0
      delay=0
      do 73 k1=1,in
      if(cut(k1).eq.0)then
      sumwa(k1)=0
      else if(cut(k1).gt.0)then
      sumwa(k1)=rang1-cut(k1)
      end if
      delay=delay+sumwa(k1)
      drive=drive+driv(k1)
73    continue
      drive=drive+add*a(o1+1)
815   redn=rc3+rc1
      delay1=delay/redn
      g=2*gn/rang1
      red=4*redn/rang1
      wait=rang1
      drive1=cdrive+drive
      delay1=cdelay+delay
      redn1=credn+redn
      gn1=cgn+gn
      cost=(delay1*gn1*(redn+gn))/(100*redn1*gn*drive1)
      call fuzzy(g,red,wait,mu)
      degree=mu
      if(degree.eq.1) go to 820
      go to 794
```

```
820   cp1(h1)=wait
      h1=h1+1
      if(rang.gt.1800)go to 925
      y1=1
      gn=rc3
      add=rc2
      cdrive=drive1
      cdelay=delay1
      credn=redn1
      cgn=gn1
      rc2=0
      rang2=0
      j1=i1
      in=0
825   j1=j1+1
      rj1=j1
      in=in+1
      p1=rinten(42)/(rinten(42)+rinten(17)+rinten(37))
      p2=rinten(17)/(rinten(42)+rinten(17)+rinten(37))
      p3=rinten(37)/(rinten(42)+rinten(17)+rinten(37))
      p4=0
      call allocate(p1,p2,p3,p4,q,ix)
      que=q
      if(que.eq.3) then
      gn=gn+1
      beta=1/rlink(37)
      call expo(beta,bx,ix)
      a(in)=bx
      sumg=0
      do 62 j=1,in
      sumg=sumg+a(j)
62    continue
      driv(in)=sumg
      cut(in)=0
      else if(que.eq.1) then
```

```
      rc1=rc1+1
      beta=1/rlink(42)
      call expo(beta,bx,ix)
      a(in)=bx
       scut=0
      do 67 j=1,in
      scut=scut+a(j)
67    continue
      driv(in)=0
      cut(in)=scut
      else if(que.eq.2)then
      rc2=rc2+1
      beta=1/rlink(17)
      call expo(beta,bx,ix)
      a(in)=bx
      scut=0
      do 68 j=1,in
      scut=scut+a(j)
68    continue
      driv(in)=0
      cut(in)=scut
      end if
      rang=rang+a(in)
      rang2=rang2+a(in)
      if(j1-i1.eq.1)go to 825
      delay=0
      drive=0
      do 74 k1=1,in
      if(cut(k1).eq.0)then
      sumwa(k1)=0
      else if(cut(k1).gt.0)then
      sumwa(k1)=rang1-cut(k1)
      end if
      delay=delay+sumwa(k1)
      drive=drive+driv(k1)
```

```
74    continue
      drive=drive+add*a(i1+1)
845   redn=rc1+rc2
      delay2=delay/redn
      g=2*gn/rang2
      red=4*redn/rang2
      wait=rang2
      drive1=cdrive+drive
      delay1=cdelay+delay
      redn1=credn+redn
      gn1=cgn+gn
      cost=(delay1*gn1*(redn+gn))/(100*redn1*gn*drive1)
c     write(6,23)rj1,rang,rang2,gn,redn,cost,degree
c23   format(7f10.3)
      call  fuzzy(g,red,wait,mu)
      degree=mu
      if(degree.eq.1) go to 850
      go to 825
850   cp2(y1)=wait
      y1=y1+1
      if(rang.gt.1800)go to 925
      gn=rc1
      add=rc1
      cdrive=drive1
      cdelay=delay1
      credn=redn1
      cgn=gn1
      rc3=0
      rang3=0
      in=0
      o1=j1
860   o1=o1+1
      in=in+1
      p1=rinten(42)/(rinten(42)+rinten(17)+rinten(37))
      p2=rinten(17)/(rinten(42)+rinten(17)+rinten(37))
```

```
      p3=rinten(37)/(rinten(42)+rinten(17)+rinten(37))
      p4=0
      call allocate(p1,p2,p3,p4,q,ix)
      que=q
      if(que.eq.1) then
      gn=gn+1
      beta=1/rlink(42)
      call expo(beta,bx,ix)
      a(in)=bx
      sumg=0
      do 63 j=1,in
      sumg=sumg+a(j)
63     continue
      driv(in)=sumg
      cut(in)=0
      wase(in)=0
      else if(que.eq.2) then
      rc2=rc2+1
      beta=1/rlink(17)
      call expo(beta,bx,ix)
      a(in)=bx
       scut=0
      do 69 j=1,in
      scut=scut+a(j)
69     continue
      driv(in)=0
      cut(in)=scut
      else if (que.eq.3)then
      rc3=rc3+1
      beta=1/rlink(37)
      call expo(beta,bx,ix)
      a(in)=bx
       scut=0
      do 171 j=1,in
      scut=scut+a(j)
```

```
171    continue
      driv(in)=0
      cut(in)=scut
      end if
      rang=rang+a(in)
      rang3=rang3+a(in)
      if(o1-j1.eq.1)go to 860
      delay=0
      drive=0
      do 276 k1=1,in
      if(cut(k1).eq.0)then
      sumwa(k1)=0
      else if(cut(k1).gt.0)then
      sumwa(k1)=rang1-cut(k1)
      end if
      delay=delay+sumwa(k1)
      drive=drive+driv(k1)
276   continue
      drive=drive+add*a(j1+1)
880   redn=rc2+rc3
      delay3=delay/redn
      g=2*gn/rang3
      red=4*redn/rang3
      wait=rang3
      drive1=cdrive+drive
      delay1=cdelay+delay
      redn1=credn+redn
      gn1=cgn+gn
      cost=(delay1*gn1*(redn+gn))/(100*redn1*gn*drive1)
      call  fuzzy(g,red,wait,mu)
      degree=mu
      if(degree.eq.1) go to 885
      go to 860
885   cp3(ai)=wait
      ai=ai+1
```

```
      if(rang.gt.1800)go to 925
      gn=rc2
      add=rc2
      cdrive=drive1
      cdelay=delay1
      credn=redn1
      cgn=gn1
      rc1=0
      in=0
      rang1=0
      i1=o1
      go to 794
925   b1=1
      c1=1
      v1=1
      xo=1
      gn=0
      s1=0
      i1=0
      s1=0
      in=0
      rc1=0
      rc2=0
      rc3=0
      rang=0
      rang1=0
890   i1=i1+1
      ri1=i1
      in=in+1
      p1=rinten(28)/(rinten(28)+rinten(29)+rinten(24)+rinten(1))
      p2=rinten(29)/(rinten(28)+rinten(29)+rinten(24)+rinten(1))
      p3=rinten(24)/(rinten(28)+rinten(29)+rinten(24)+rinten(1))
      p4=rinten(1)/(rinten(28)+rinten(29)+rinten(24)+rinten(1))
      call allocate(p1,p2,p3,p4,q,ix)
      que=q
```

```
      if(que.eq.4) then
      gn=gn+1
      beta=1/rlink(1)
      call expo(beta,bx,ix)
      a(in)=bx
      sumg=0
      do 71 j=1,in
      sumg=sumg+a(j)
71    continue
      driv(in)=sumg
      cut(in)=0
      else if(que.eq.2) then
      rc2=rc2+1
      beta=1/rlink(29)
      call expo(beta,bx,ix)
      a(in)=bx
      scut=0
      do 76 j=1,in
      scut=scut+a(j)
76    continue
      driv(in)=0
      cut(in)=scut
      else if(que.eq.3) then
      rc3=rc3+1
      beta=1/rlink(24)
      call expo(beta,bx,ix)
      a(in)=bx
      scut=0
      do 77 j=1,in
      scut=scut+a(j)
77    continue
      driv(in)=0
      cut(in)=scut
      else if(que.eq.1)then
      rc1=rc1+1
```

```
      beta=1/rlink(28)
      call expo(beta,bx,ix)
      a(in)=bx
      scut=0
      do 78 j=1,in
      scut=scut+a(j)
78    continue
      driv(in)=0
      cut(in)=scut
      end if
      rang=rang+a(in)
      rang1=rang1+a(in)
      if(i1-s1.eq.1)go to 890
      delay=0
      drive=0
      do 91 k1=1,in
      if(cut(k1).eq.0)then
      sumwa(k1)=0
      else if(cut(k1).gt.0)then
      sumwa(k1)=rang1-cut(k1)
      end if
      delay=delay+sumwa(k1)
      drive=drive+driv(k1)
91    continue
      drive=drive+add*a(s1+1)
915   redn=rc1+rc2+rc3
      delay1=delay/redn
      g=2*gn/rang1
      red=3*redn/rang1
      wait=rang1
      drive1=cdrive+drive
      delay1=cdelay+delay
      redn1=credn+redn
      gn1=cgn+gn
      cost=(delay1*gn1*(redn+gn))/(100*redn1*gn*drive1)
```

```
      call fuzzy(g,red,wait,mu)
      degree=mu
      if(degree.eq.1) go to 920
      go to 890
920   ap1(b1)=wait
      b1=b1+1
      if(rang.gt.1800)go to 1020
      gn=rc1
      add=rc1
      cdrive=drive1
      cdelay=delay1
      credn=redn1
      cgn=gn1
      rc4=0
      rang2=0
      j1=i1
      in=0
924   j1=j1+1
      rj1=j1
      in=in+1
      p1=rinten(28)/(rinten(28)+rinten(29)+rinten(24)+rinten(1))
      p2=rinten(29)/(rinten(28)+rinten(29)+rinten(24)+rinten(1))
      p3=rinten(24)/(rinten(28)+rinten(29)+rinten(24)+rinten(1))
      p4=rinten(1)/(rinten(28)+rinten(29)+rinten(24)+rinten(1))
      call allocate(p1,p2,p3,p4,q,ix)
      que=q
      if(que.eq.1) then
      gn=gn+1
      beta=1/rlink(28)
      call expo(beta,bx,ix)
      a(in)=bx
      sumg=0
      do 72 j=1,in
      sumg=sumg+a(j)
72    continue
```

```
      driv(in)=sumg
      cut(in)=0
      else if(que.eq.2) then
      rc2=rc2+1
      beta=1/rlink(29)
      call expo(beta,bx,ix)
      a(in)=bx
      scut=0
      do 79 j=1,in
      scut=scut+a(j)
79    continue
      driv(in)=0
      cut(in)=scut
      else if(que.eq.3) then
      rc3=rc3+1
      beta=1/rlink(24)
      call expo(beta,bx,ix)
      a(in)=bx
      scut=0
      do 81 j=1,in
      scut=scut+a(j)
81    continue
      driv(in)=0
      cut(in)=scut
      else if(que.eq.4)then
      rc4=rc4+1
      beta=1/rlink(1)
      call expo(beta,bx,ix)
      a(in)=bx
      scut=0
      do 82 j=1,in
      scut=scut+a(j)
82    continue
      driv(in)=0
      cut(in)=scut
```

```
      wase(in)=bx
      end if
      rang=rang+a(in)
      rang2=rang2+a(in)
      if(j1-i1.eq.1)go to 924
      delay=0
      drive=0
      do 92 k1=1,in
      if(cut(k1).eq.0)then
      sumwa(k1)=0
      else if(cut(k1).gt.0)then
      sumwa(k1)=rang1-cut(k1)
      end if
      delay=delay+sumwa(k1)
      drive=drive+driv(k1)
92    continue
      drive=drive+add*a(i1+1)
945   redn=rc2+rc3+rc4
      delay2=delay/redn
      g=3*gn/rang2
      red=3*redn/rang2
      wait=rang2
      drive1=cdrive+drive
      delay1=cdelay+delay
      redn1=credn+redn
      gn1=cgn+gn
      cost=(delay1*gn1*(redn+gn))/(100*redn1*gn*drive1)
      call fuzzy(g,red,wait,mu)
      degree=mu
      if(degree.eq.1) go to 950
      go to 924
950   ap2(c1)=wait
      c1=c1+1
      if(rang2.ge.1800)go to 1020
      v1=1
```

```
      gn=rc3
      add=rc3
      cdrive=drive1
      cdelay=delay1
      credn=redn1
      cgn=gn1
      rc1=0
      o1=j1
      rang3=0
      in=0
955   o1=o1+1
      in=in+1
      p1=rinten(28)/(rinten(28)+rinten(29)+rinten(24)+rinten(1))
      p2=rinten(29)/(rinten(28)+rinten(29)+rinten(24)+rinten(1))
      p3=rinten(24)/(rinten(28)+rinten(29)+rinten(24)+rinten(1))
      p4=rinten(1)/(rinten(28)+rinten(29)+rinten(24)+rinten(1))
      call allocate(p1,p2,p3,p4,q,ix)
      que=q
      if(que.eq.3) then
      gn=gn+1
      beta=1/rlink(24)
      call expo(beta,bx,ix)
      a(in)=bx
      sumg=0
      do 273 j=1,in
      sumg=sumg+a(j)
273   continue
      driv(in)=sumg
      cut(in)=0
      wase(in)=0
      else if(que.eq.1) then
      rc1=rc1+1
      beta=1/rlink(28)
      call expo(beta,bx,ix)
      a(in)=bx
```

```
      scut=0
      do 83 j=1,in
      scut=scut+a(j)
83    continue
      driv(in)=0
      cut(in)=scut
      else if(que.eq.2) then
      rc2=rc2+1
      beta=1/rlink(29)
      call expo(beta,bx,ix)
      a(in)=bx
      scut=0
      do 84 j=1,in
      scut=scut+a(j)
84    continue
      driv(in)=0
      cut(in)=scut
      else if(que.eq.4)then
      rc4=rc4+1
      beta=1/rlink(1)
      call expo(beta,bx,ix)
      a(in)=bx
      scut=0
      do 86 j=1,in
      scut=scut+a(j)
86    continue
      driv(in)=0
      cut(in)=scut
      end if
      rang=rang+a(in)
      rang3=rang3+a(in)
      if(o1-j1.eq.1)go to 955
      delay=0
      drive=0
      do 93 k1=1,in
```

```
      if(cut(k1).eq.0)then
      sumwa(k1)=0
      else if(cut(k1).gt.0)then
      sumwa(k1)=rang1-cut(k1)
      end if
      delay=delay+sumwa(k1)
      drive=drive+driv(k1)
93    continue
      drive=drive+add*a(j1+1)
975   redn=rc1+rc2+rc4
      delay3=delay/redn
      g=2*gn/rang3
      red=3*redn/rang3
      wait=rang3
      drive1=cdrive+drive
      delay1=cdelay+delay
      redn1=credn+redn
      gn1=cgn+gn
      cost=(delay1*gn1*(redn+gn))/(100*redn1*gn*drive1)
      call fuzzy(g,red,wait,mu)
      degree=mu
      if(degree.eq.1) go to 980
      go to 955
980   ap3(v1)=wait
      v1=v1+1
      if(rang.ge.1800)go to 1020
      gn=rc2
      add=rc2
      cdrive=drive1
      cdelay=delay1
      credn=redn1
      cgn=gn1
      rc3=0
      rang4=0
      s1=o1
```

```
      in=0
985   s1=s1+1
      in=in+1
      p1=rinten(28)/(rinten(28)+rinten(29)+rinten(24)+rinten(1))
      p2=rinten(29)/(rinten(28)+rinten(29)+rinten(24)+rinten(1))
      p3=rinten(24)/(rinten(28)+rinten(29)+rinten(24)+rinten(1))
      p4=rinten(1)/(rinten(28)+rinten(29)+rinten(24)+rinten(1))
      call allocate(p1,p2,p3,p4,q,ix)
      que=q
      if(que.eq.2) then
      gn=gn+1
      beta=1/rlink(29)
      call expo(beta,bx,ix)
      a(in)=bx
      sumg=0
      do 174 j=1,in
      sumg=sumg+a(j)
174   continue
      driv(in)=sumg
      cut(in)=0
      else if(que.eq.3) then
      rc3=rc3+1
      beta=1/rlink(24)
      call expo(beta,bx,ix)
      a(in)=bx
      scut=0
      do 87 j=1,in
      scut=scut+a(j)
87    continue
      driv(in)=0
      cut(in)=scut
      else if(que.eq.1)then
      rc1=rc1+1
      beta=1/rlink(28)
      call expo(beta,bx,ix)
```

```
      a(in)=bx
      scut=0
      do 88 j=1,in
      scut=scut+a(j)
88    continue
      driv(in)=0
      cut(in)=scut
      wase(in)=bx
      else if(que.eq.4)then
      rc4=rc4+1
      beta=1/rlink(1)
      call expo(beta,bx,ix)
      a(in)=bx
      scut=0
      do 89 j=1,in
      scut=scut+a(j)
89    continue
      driv(in)=0
      cut(in)=scut
      end if
      rang=rang+a(in)
      rang4=rang4+a(in)
      if(s1-o1.eq.1)go to 985
      delay=0
      drive=0
      do 94 k1=1,in
      if(cut(k1).eq.0)then
      sumwa(k1)=0
      else if(cut(k1).gt.0)then
      sumwa(k1)=rang1-cut(k1)
      end if
      delay=delay+sumwa(k1)
      drive=drive+driv(k1)
94    continue
      drive=drive+add*a(o1+1)
```

```
1005  redn=rc1+rc3+rc4
      delay4=delay/redn
      g=2*gn/rang4
      red=3*redn/rang4
      wait=rang4
      drive1=cdrive+drive
      delay1=cdelay+delay
      redn1=credn+redn
      gn1=cgn+gn
      cost=(delay1*gn1*(redn+gn))/(100*redn1*gn*drive1)
c      write(6,43)s1,rang,rang4,gn,redn,cost,degree
c43    format(7f10.5)
      call fuzzy(g,red,wait,mu)
      degree=mu
      if(degree.eq.1) go to 1010
      go to 985
1010  ap4(xo)=wait
      xo=xo+1
      if(rang.ge.1800)go to 1020
      i1=s1
      gn=rc4
      add=rc4
      cdrive=drive1
      cdelay=delay1
      credn=redn1
      cgn=gn1
      rc2=0
      in=0
      rang1=0
      go to 890
1020  STOP
      end
```

```
c*************************************************
c                        2. Sub-Program
c*************************************************
c*************************************************
c          2.1   Subroutine for allocate car to each  branch
c*************************************************
      subroutine allocate(p1,p2,p3,p4,q,ix)
      rn=unif(ix)
      if(rn.lt.p1) then
      q=1
      else if(rn.ge.p1.and.rn.lt.p1+p2) then
      q=2
      else if(rn.ge.p1+p2.and.rn.lt.p1+p2+p3) then
      q=3
      else if(rn.ge.p1+p2+p3) then
      q=4
      end if
      return
      end
c********************************************************
c      2.2  Subroutine for generate exponential random variable
c********************************************************
      subroutine expo(beta,bx,ix)
      rn=unif(ix)
      bx=-beta*alog(rn)
      return
      end
C*************************************************
C        2.3 Subroutine for generate gamma random variable
C*************************************************
      Subroutine gamma
      common ix,al,be,x,xp,rmean,min,xmax,xu,g,re,w1,l1,p
555   rn=unif(ix)
      v=-be*alog(rn)
      x=x+v
```

```
      if(al.eq.1)go to 520
      al=al-1
      go to 555
520   x=x
      return
      end
C*****************************************************
C         2.4 Subroutine for generate poisson random variable
C*****************************************************
      subroutine poiss
      common ix,ial,be,x,xp,rmean,min,xmax,xu,g,re,w1,l1,p
      xp=0.0
      a=2.718**(-rmean)
      s=1.0
4     rn=unif(ix)
      s=s*rn
      if(s-a)9,7,7
7     xp=xp+1.0
      go to 4
9     return
      end
c**********************************************
c       2.5 Subroutine for generate bernoulie random variable
c**********************************************
      subroutine ber(p,x,ix)
      rn=unif(ix)
      rr=1-p
      if(rn.le.rr)go to 525
      X=1.0
      go to 530
525   x=0.0
530   return
      end
```

```
c*************************************************
c        2.6 Function for generate random number
c*************************************************
      FUNCTION UNIF(IX)
      K1=IX/127773
      IX=16807*(IX-K1*127773)-K1*2836
      IF(IX.LT.0)IX=IX+2147483647
      UNIF=IX*4.656612875E-10
      IX=IX
      RETURN
c*****************************************************
c            2.8 Subroutine for fuzzy logic system
c*****************************************************
      SUBROUTINE FUZZY(G,RED,WAIT,MU)
      IX=1234567
      IF(G.GT.1) THEN
      GZ=0
      ELSE IF(G.LE.1.AND.G.GE.0) THEN
      GZ=1-G
      END IF
      IF(G.LE.1.AND.G.GE.0) THEN
      GL=G
      ELSE IF(G.LE.2.AND.G.GT.1) THEN
      GL=1
      ELSE IF(G.LE.3.AND.G.GT.2) THEN
      GL=3-G
      ELSE IF(G.GT.3) THEN
      GL=0
      END IF
      IF(G.LE.2) THEN
      GM=0
      ELSE IF(G.LE.3.AND.G.GT.2) THEN
      GM=G-2
      ELSE IF(G.LE.4.AND.G.GT.3) THEN
      GM=4-G
```

```
ELSE IF(G.GT.4) THEN
GM=0
END IF
IF(G.LT.3) THEN
GH=0
ELSE IF(G.LT.4.AND.G.GE.3) THEN
GH=G-3
ELSE IF(G.GE.4) THEN
GH=1
END IF
IF(RED.LT.1.AND.RED.GE.0) THEN
RZ=1-RED
ELSE IF(RED.GE.1)THEN
RZ=0
END IF
IF(RED.LT.1.AND.RED.GE.0) THEN
RL=RED
ELSE IF(RED.LT.3.AND.RED.GE.1) THEN
RL=1
ELSE IF(RED.LT.6.AND.RED.GE.3) THEN
RL=2-RED/3
ELSE IF(RED.GE.6) THEN
RL=0
END IF
IF(RED.LT.3) THEN
RM=0
ELSE IF(RED.LT.6.AND.RED.GE.3) THEN
RM=RED/3-1
ELSE IF(RED.LT.9.AND.RED.GE.6) THEN
RM=3-RED/3
ELSE IF(RED.GE.9) THEN
RM=0
END IF
IF(RED.LT.6) THEN
RH=0
```

```
ELSE IF(RED.LT.9.AND.RED.GE.6) THEN
RH=RED/3-2
ELSE IF(RED.GE.9) THEN
RH=1
END IF
IF(WAIT.LT.30.AND.WAIT.GE.0) THEN
WS=1
ELSE IF(WAIT.LT.60.AND.WAIT.GE.30) THEN
WS=2-WAIT/30
ELSE IF(WAIT.GE.60) THEN
WS=0
END IF
IF(WAIT.LT.30) THEN
WM=0
ELSE IF(WAIT.LT.60.AND.WAIT.GE.30) THEN
WM=WAIT/30-1
ELSE IF(WAIT.LT.90.AND.WAIT.GE.60) THEN
WM=3-WAIT/30
ELSE IF(WAIT.GE.90) THEN
WM=0
END IF
IF(WAIT.LT.60) THEN
WL=0
ELSE IF(WAIT.LT.90.AND.WAIT.GE.60) THEN
WL=WAIT/30-2
ELSE IF(WAIT.GE.90) THEN
WL=1
END IF
A1=0.05
A2=0.2
A3=0.2
A4=0.2
A5=0.15
C1=0.033
C2=0.3
```

```
      C3=0.5
      C4=0.7
      C5=0.85
      S1=GZ*RZ*A1
      S2=GZ*RL*A5
      S3=GZ*RM*A5
      S4=GZ*RH*A5
      S5=RZ*A1
      S6=GL*RL*A1
      S7=GM*RM*A1
      S8=GH*RH*A1
      S9=GL*RM*WS*A3
      S10=GL*RM*WM*A4
      S11=GL*RM*WL*A5
      S12=GL*RH*WS*A2
      S13=GL*RH*WM*A3
      S14=GL*RH*WL*A4
      S15=GM*RL*WS*A2
      S16=GM*RL*WM*A2
      S17=GM*RL*WL*A3
      S18=GM*RH*WS*A3
      S19=GM*RH*WM*A4
      S20=GM*RH*WL*A5
      S21=GH*RL*WS*A3
      S22=GH*RL*WM*A4
      S23=GH*RL*WL*A5
      S24=GH*RM*WS*A2
      S25=GH*RM*WM*A2
   S26=GH*RM*WL*A3
UPER=S1*C1+S2*C5+S3*C5+S4*C5+S5*C1+S6*C1+S7*C1+S8*C1&
+S9*C3+S10*C4+S11*C5+S12*C2+S13*C3+S14*C4+S15*C2
&+S16*C2+S17*C3+S18*C3+S19*C4+S20*C5+S21*C3+S22*C4
&+S23*C5+S24*C2+S25*C2+S26*C3
ROWER=S1+S2+S3+S4+S5+S6+S7+S8+S9+S10+S11+S12+S13+S14
&+S15+S16+S17+S18+S19+S20+S21+S22+S23+S24+S25+S26
```

```
RUL=UPER/ROWER
P=RUL
CALL BER(P,X,ix)
MU=X
RETURN
END
```