CHAPTER 3 TPS CONTROL LOGIC AND PROPOSED MODELS

This chapter first introduces the concept of the FLC-based conditional TPS model. Then the design of the FLC is presented. In the last two sections of this chapter, the models integrating GA and ACO into the FLC (named GFLC and AGFLC) are developed and elaborated in detail.

3.1 TPS Control Logic

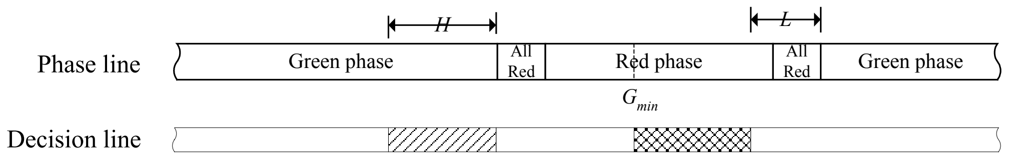
This study employs green extension and red truncation as priority strategies for a transit vehicle. Green extension is activated only in the green phases to conclude an extension green time as a transit vehicle approaching the intersection, while red truncation is activated only in the red phases to shorten the present red time as a transit vehicle approaching the intersection. The control logic of implementing green extension and red truncation under unconditional and FLC-based conditional TPS is detailed as follows.

3.1.1 Unconditional TPS

If unconditional TPS model is employed, the control logic of green extension strategy and red truncation strategy is described by following two rules (depicted in Figure 3-1):

Rule 1 (green extension strategy): In the green phase, **IF** $GR < H$, **THEN** implement green extension strategy and let $G_{ext} = H - GR$, where GR represents the remaining green time at the moment when a transit vehicle actuates the detector. H represents the time needed for a transit vehicle traveling from the detector through the far-side stop line of the intersection. G_{ext} represents the green extension time.

Rule 2 (red truncation strategy): In the red phase, **IF** $(RR + AR) > L$, **THEN** implement red truncation strategy and let $R_{tru} = RR + AR - L$, where RR represents the remaining red time when a transit actuates the detector; and AR represents the all-red time. L represents the time needed for a transit vehicle traveling from the detector to the near-side stop line of the intersection. R_{tru} represents the red truncation time.



Notes: ▨: Implement green extension strategy while transit actuates detector during this period.
▦: Implement red truncation strategy while transit actuates detector during this period.
□: Do not implement any transit-preemption strategy while transit actuates detector during this period.

Figure 3-1 Control logic of unconditional TPS model.

The definitions of H and L are illustrated in Figure 3-2. The green time can be extended by the continuously arriving transit vehicles as long as the green time do not exceed the maximal green time. The red time is truncated after the minimal green time once the transit vehicle requests for the priority. Therefore, to avoid the serious distortion of original signal timing plans, these two strategies are implemented under the following conditions: (1) If the phase comes to a transition period, such as all-red, it will not activate any strategy. (2) The total green extension time should not exceed the maximal green time (G_{max}). (3) The red time after truncation should not be less than the minimal green time (G_{min}). (4) No compensation mechanism is provided. All parameters including H, L, AR, G_{max} and G_{min}, are given.

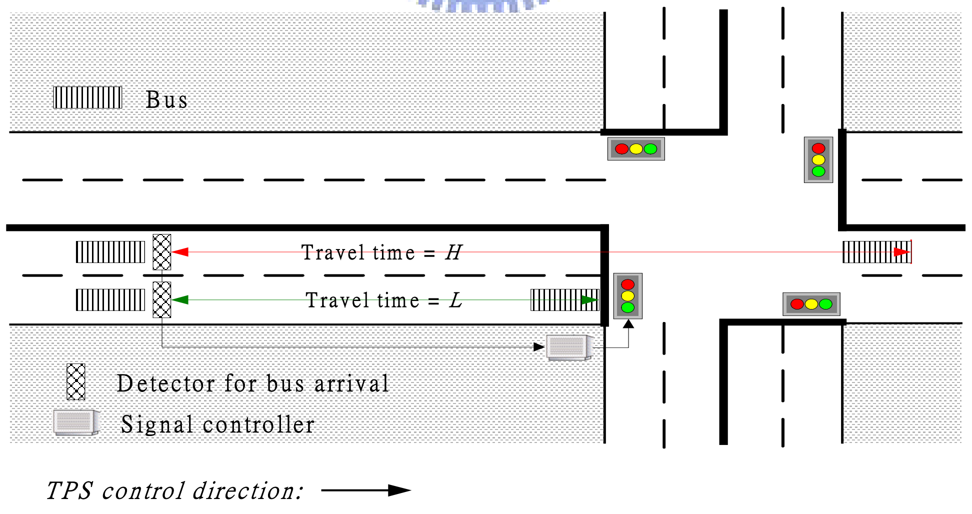


Figure 3-2 Illustrated definitions of H and L.

3.1.2 Conditional TPS

If conditional TPS is implemented, giving of priority should be determined before communicating to the traffic signal controller. This study concludes a decision for providing priority to the approaching transit vehicles by the GFLC or AGFLC with considerations of traffic situations in all approaches so as to minimize the total person delay of the intersection. The control logic of green extension strategy and red truncation strategy is described as rules 3 and 4 (depicted in Figure 3-3).

Rule 3 (green extension strategy): In the green phase, **IF** $GR < H$ **AND** $NE \geq N_t$, **THEN** implement the green extension strategy and let $G_{ext} = H - GR$, where NE represents the degree of necessity to implement TPS, which is concluded by the AGFLC with a value ranging from 0 to 1. N_t represents the threshold value preset to determine whether the priority is provided or not.

Rule 4 (red truncation strategy): In the red phase, **IF** $(RR + AR) > L$ **AND** $NE \geq N_t$, **THEN** implement the red truncation strategy and let $R_{tru} = RR + AR - L$.

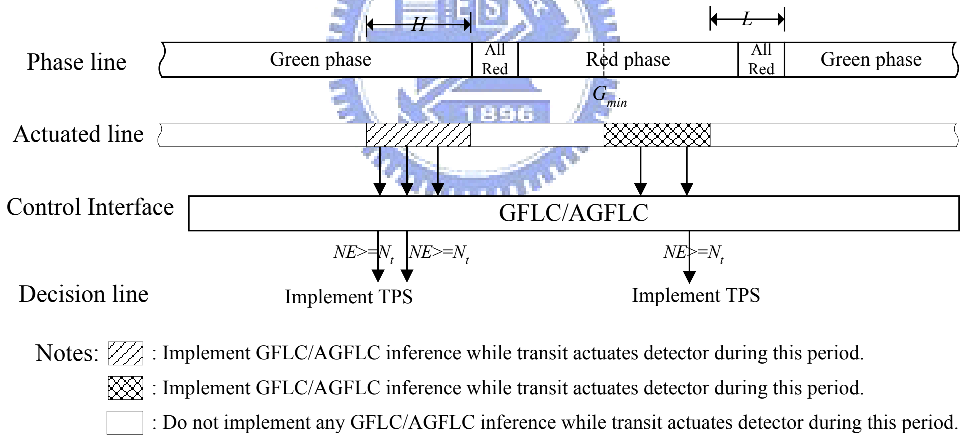

H
L
Phase line
Green phase
All Red
Red phase
All Red
Green phase
G_{min}
Actuated line
Control Interface
GFLC/AGFLC
$NE>=N_t$
$NE>=N_t$
$NE>=N_t$
Decision line
Implement TPS
Implement TPS
Notes: ▨ : Implement GFLC/AGFLC inference while transit actuates detector during this period.
▩ : Implement GFLC/AGFLC inference while transit actuates detector during this period.
□ : Do not implement any GFLC/AGFLC inference while transit actuates detector during this period.

Figure 3-3 Control logic of conditional TPS model.

Similarly, the abovementioned two rules are implemented under the same conditions of unconditional TPS model.

3.2 Design of the FLC

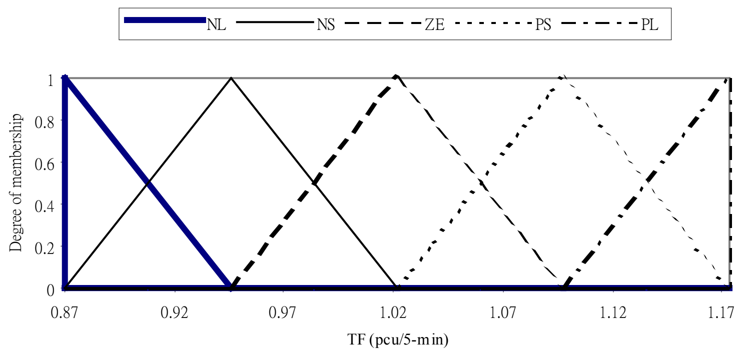
As abovementioned, an FLC is constituted by fuzzy logic rules that form a control mechanism to approximate expert perception and judgment under the given information. The rule base is composed of finite IF-THEN rules with state and control variables. With the considerations of expert intuition and the convenience of traffic data acquirement, this study uses total traffic flows (*TF*) at all approaches in the green phase and total queue length (*QL*) at all approaches in the red phase as the state variables and the degree of necessity for implementing TPS (*NE*) as the control variable to form the proposed FLC. These variables are assumed with five linguistic degrees (*NL*: negative large, *NS*: negative small, *ZE*: zero, *PS*: positive small, *PL*: positive large) represented by triangular membership functions. This makes a total of 25 combinations in the antecedent part of the fuzzy rule base. Moreover, the fuzzy rules use AND as the connecting operators between the state variables. The rule base is illustrated in Figure 3-4 and the triangular membership functions are illustrated in Figure 3-5.

Rule 1 ： IF $TF = NL$ AND $QL = NL$ THEN $NE = B_1$
Rule 2 ： IF $TF = NL$ AND $QL = NS$ THEN $NE = B_2$
Rule 3 ： IF $TF = NL$ AND $QL = ZE$ THEN $NE = B_3$
.
.
.
Rule 25 ： IF $TF = PL$ AND $QL = PL$ THEN $NE = B_{25}$

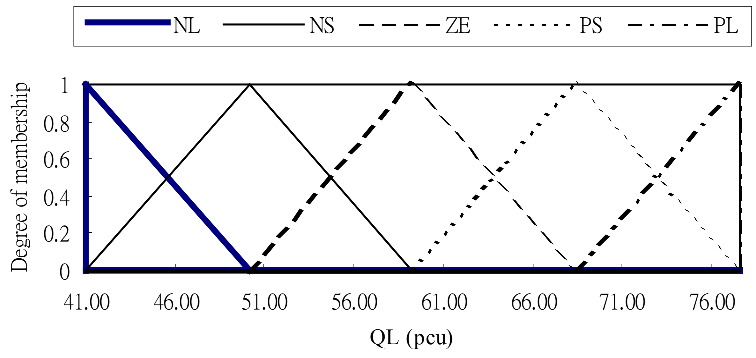
Where, $B_i \in \{NL, NS, ZE, PS, PL\}$, $i = 1 \sim 25$

Figure 3-4 Designed rule base of the FLC.

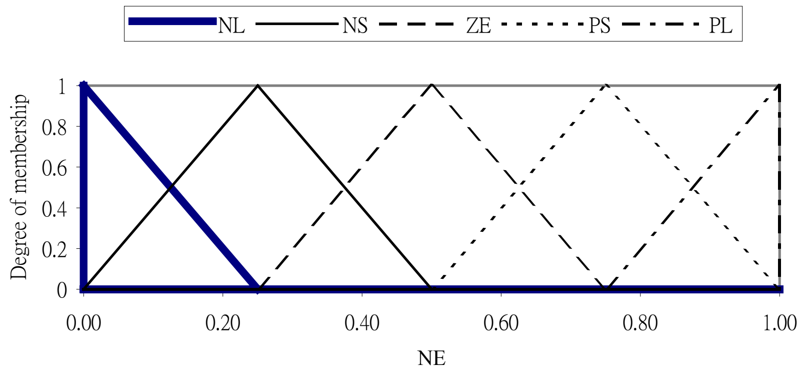
The implementation of the proposed FLC requires some real-time traffic information: arrival of transit, traffic flow in the green-phase direction, and queue length in the red-phase direction. Thus, the transit vehicles should be equipped with such devices as global positioning system (GPS) to provide the information on arrival. On the other hand, in order to collect the information on traffic flow and queue length of other vehicles, two sets of detectors acting as check-in and check-out points are also required on all lanes in all approaches. The former detector set can be located near the stop line of the intersection to count the number of departing vehicles; the later detector set can be at a certain distant point from the stop line to count the number of arriving traffic. The queue length is thus determined by the difference between these two counting results. In practice, the distance between these two sets of detectors requires a proper design to accommodate the possible maximum queue length. Moreover, with the emerging detection and communication technologies, numerous advanced detectors are introduced that would facilitate the implementation of more sophisticated TPS systems.



(a)



(b)



(c)

Figure 3-5 Designed membership functions of the FLC: (a) traffic flow (*TF*), (b) queue length (*QL*), (c) necessity for implementing TPS (*NE*).

3.3 Genetic Fuzzy Logic Controller (GFLC)

The two models proposed by this study select fuzzy rules either by GA or ACO method and then tune membership functions by GA. They are named as GFLC and AGFLC, respectively. This section presents GFLC model first. The GFLC selects fuzzy rules and tunes membership functions by GA in sequence. The encoding methods for fuzzy rules and membership functions, genetic operators, and iterative GFLC evolution algorithm are described below.

3.3.1 Encoding method for fuzzy rules

The encoding method proposed by Thrift (1991) is adopted to effectively shorten the length of chromosome. Each antecedent in the rule base is represented by one gene and its linguistic degree of control variable is indicated by the value of the corresponding gene. Taking two state variables (x_1, x_2) and one control variable (y) as an example, if each variable has five linguistic degrees (*NL*: negative large, *NS*: negative small, *ZE*: zero, *PS*: positive small, *PL*: positive large), then the chromosome length is 25. Genes take the integers from 0 to 5, where 0 represents the exclusion of the rules; other numbers indicate the inclusion of the rules and the linguistic degrees of control variable. This encoding method is depicted in Figure 3-6. A chromosome with gene sequence of 0002040010000001000030000, for example, will represent the following five fuzzy rules being selected.

Rule 1: IF $x_1 = NL$ AND $x_2 = PS$ THEN $y = NS$

Rule 2: IF $x_1 = NS$ AND $x_2 = NL$ THEN $y = PS$

Rule 3: IF $x_1 = NS$ AND $x_2 = PS$ THEN $y = NL$

Rule 4: IF $x_1 = PS$ AND $x_2 = NL$ THEN $y = NL$

Rule 5: IF $x_1 = PL$ AND $x_2 = NL$ THEN $y = ZE$

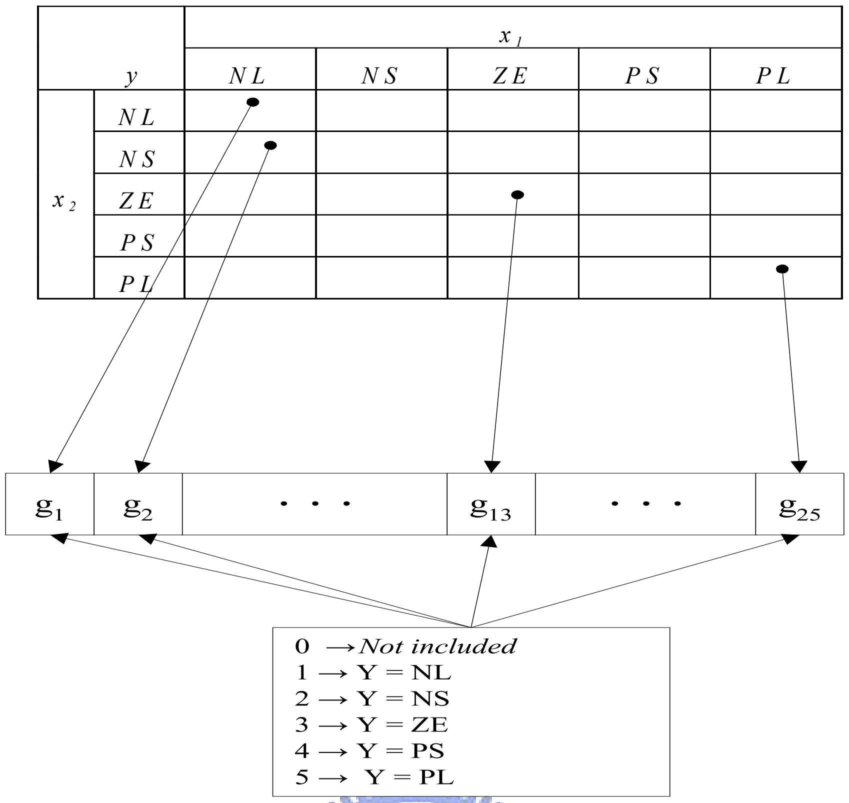


Figure 3-6 Encoding method for fuzzy rules.

3.3.2 Encoding method for membership function

Consider a triangle fuzzy number and let parameters c_k^r, c_k^c and c_k^l respectively represent the coordinates of right anchor, cortex and left anchor of k^{th} linguistic degree. Then 15 parameters need to be calibrated for a variable with five linguistic degrees. Furthermore, it is assumed the first and last degrees of fuzzy numbers are left- and right-skewed triangles, respectively and that the others are isosceles triangles as shown in Figure 3-7. Therefore, a variable with five linguistic degrees has eight parameters to be calibrated and their orders are:

$$c_{\max} = c_5^c = c_5^r \geq c_4^r \geq \genfrac{}{}{0pt}{}{c_5^l}{c_3^r} \geq \genfrac{}{}{0pt}{}{c_4^l}{c_2^r} \geq \genfrac{}{}{0pt}{}{c_3^l}{c_1^r} \geq c_2^l \geq c_1^c = c_1^l = c_{\min} \tag{3.1}$$

$$c_k^c = \frac{(c_k^r + c_k^l)}{2}, \; k\text{=2, 3, 4} \tag{3.2}$$

where $c_{\max}$ and $c_{\min}$ are the maximum and minimum values of the variable, respectively. The orders between c_5^l and c_3^r, c_4^l and c_2^r, c_3^l and c_1^r are indeterminate. In order to tune these eight parameters, nine position variables $r_1 \sim r_9$ are designed as follows:

$$c_2^l = c_{\min} + r_1 \times \omega \tag{3.3}$$

$$c_1^r = c_2^l + r_2 \times \omega \tag{3.4}$$

$$c_3^l = c_2^l + r_3 \times \omega \tag{3.5}$$

$$c_2^r = \max\{c_1^r, c_3^l\} + r_4 \times \omega \tag{3.6}$$

$$c_4^l = \max\{c_1^r, c_3^l\} + r_5 \times \omega \tag{3.7}$$

$$c_3^r = \max\{c_2^r, c_4^l\} + r_6 \times \omega \tag{3.8}$$

$$c_5^l = \max\{c_2^r, c_4^l\} + r_7 \times \omega \tag{3.9}$$

$$c_4^r = \max\{c_3^r, c_5^l\} + r_8 \times \omega \tag{3.10}$$

where $\omega = \dfrac{(c_{\max} - c_{\min})}{\sum_{i=1}^{9} r_i}$.

Each position variable is represented by four real-coding genes which are also depicted in Figure 3-7. The maximum value of the position variables is 99.99 and the minimum value is 0. Thus, in the example of two state variables and one control variable (each with five linguistic degrees), the chromosome is composed of 108 genes.

3.3.3 Genetic Operators

For the crossover, the max-min-arithmetical crossover proposed by Herrera *et al.* (1995) is employed in this study. Let $G_w^t = \{ g_{w1}^t, \ldots, g_{wk}^t, \ldots, g_{wK}^t \}$ and $G_v^t = \{ g_{v1}^t, \ldots, g_{vk}^t, \ldots, g_{vK}^t \}$ be two chromosomes selected for crossover, and the following four descendants will be generated:

$$G_1^{t+1} = aG_w^t + (1-a)G_v^t \tag{3.11}$$

$$G_2^{t+1} = aG_v^t + (1-a)G_w^t \tag{3.12}$$

$$G_3^{t+1} \text{ with } g_{3k}^{t+1} = \min\{g_{wk}^t, g_{vk}^t\} \tag{3.13}$$

G_4^{t+1} with $g_{4k}^{t+1}=max\{g_{wk}^{t}, g_{vk}^{t}\}$ (3.14)

where a is a parameter ($0 < a < 1$), and t is the number of generations.

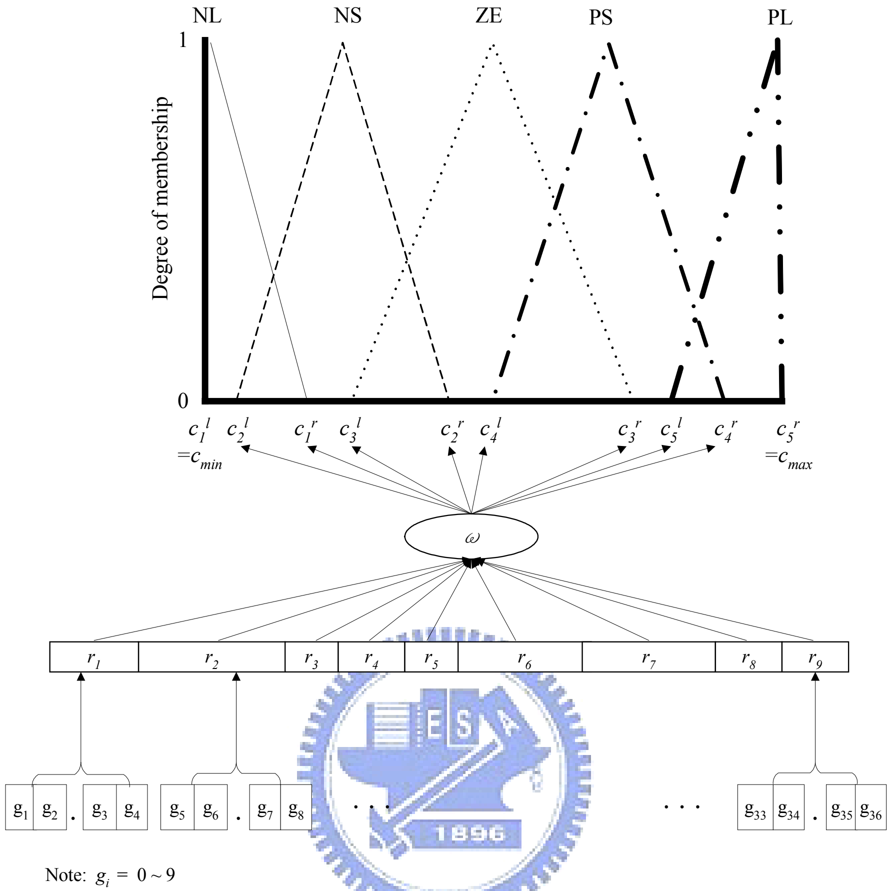


Figure 3-7 Encoding method for membership functions.

For the mutation, the non-uniform mutation proposed by Michalewicz (1992) is employed in this study. Let $G^t = \{ g_1^t ,..., g_k^t ,..., g_K^t \}$ be a chromosome and the gene g_k^t be selected for mutation (the domain of g_k^t is $[g_k^l, g_k^u]$), the value of g_k^{t+1} after mutation can be computed as follows:

$$g_k^{t+1} = \begin{cases} g_k^{t} + \Delta(t, g_k^u - g_k^{t}) & if \quad b=0 \\ g_k^{t} - \Delta(t, g_k^{t} - g_k^l) & if \quad b=1 \end{cases} \tag{3.15}$$

where b randomly takes a binary value of 0 or 1. The function $\Delta(t,z)$ returns a value in the range of $[0, z]$ such that the probability of $\Delta(t,z)$ approaches to 0 as t increases:

$$\Delta(t,z) = z(1 - r^{(1-t/T)^h}) \tag{3.16}$$

where r is a random number in the interval [0,1], T is the maximum number of generations, and h is a given constant. In Equation (3.15), the value returned by $\Delta(t,z)$ will gradually decrease as the evolution progresses.

3.3.4 Iterative GFLC evolution algorithm

The iterative evolution algorithm for selecting the fuzzy rules and tuning the membership functions is a two-step evolution process. The first step is to solve the composition of fuzzy rules using the membership functions tuned by the second step. The second step is to determine the shape of membership functions using the fuzzy rules learned from the first step. Consider an FLC with N state variables $x_1, x_2, \ldots, x_N$ and one control variable y, each with $d_1, d_2, \ldots, d_N$ and d_{N+1} linguistic degrees. Assume that the membership functions of all linguistic degrees to be triangle-shaped. The iterative evolution algorithm is structured as follows:

Step 0: Initialization. Let v=1 where v represents the number of evolution.

Step 1: Selecting fuzzy rules by GA.

Step 1-1: Encoding the fuzzy rules.

Step 1-2: Generating initial population. Randomly generate an initial population with p chromosomes. Each chromosome would have $\prod_{i=1}^{N} d_i$ genes and each gene randomly takes one integer from the interval [0, d_{N+1}].

Step 1-3: Calculating fitness values. The fitness value is set as the reciprocal of objective function of the problem to be minimized. For a rule learning problem with an input-output training dataset, the objective function could be to minimize the error between the observed output and the output concluded by the GFLC. For a rule learning problem without the training dataset, the objective function could be defined as the performance index of the control plant. The fitness value of each chromosome is calculated for the evaluation of the next step.

Step 1-4: Selection. Select the chromosomes for crossover and mutation by evaluating their fitness values with the Monte Carlo wheel method.

Step 1-5: Crossover.

Step 1-6: Mutation.

Step 1-7: Testing the stop condition of Step 1. The stop condition is set based on whether the mature rate (the proportion of same chromosome in a population) has reached a given constant δ. If so, proceed to **Step 2**; otherwise, go back to **Step 1-4**.

Step 2: Tuning membership functions by GA.

Step 2-1: Encoding the membership functions.

Step 2-2: Generating initial population. Randomly generate an initial population with p chromosomes. Each chromosome has $36(N+1)$ genes and each gene randomly takes one integer from 0, 1, 2, …, 9.

Step 2-3: Calculating fitness values. The fitness value is set as the reciprocal of objective function of the problem to be minimized. The fitness value of each chromosome is calculated for the evaluation of the next step.

Step 2-4: Selection. Select the chromosomes for crossover and mutation by evaluating their fitness values with the Monte Carlo wheel method.

Step 2-5: Crossover.

Step 2-6: Mutation.

Step 2-7: Testing the stop condition of Step 2. The stop condition is set based on whether the mature rate has reached a given constant δ. If so, proceed to **Step 3**; otherwise, go back to **Step 2-4**.

Step 3: Testing of the stop condition. If $(f_v - f_{v-1}) \le \varepsilon$, then stop, where f_v and f_{v-1} are the best objective value for the v^{th} and v-1^{th} evolution epoch respectively and ε is an arbitrary small number. The incumbent fuzzy rules and membership functions are the optimal learning results. Otherwise, let $v=v+1$ and go to **Step 1**.

The iterative GFLC evolution algorithm can be portrayed as Figure 3-8.

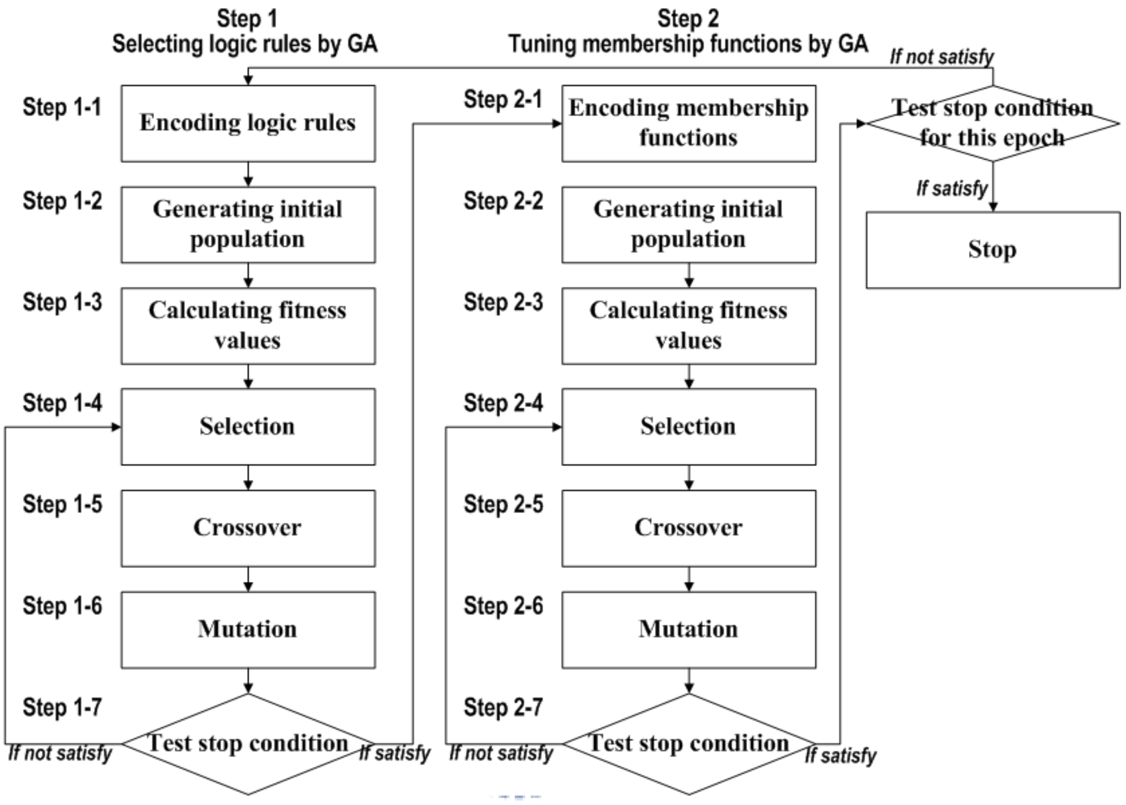


Figure 3-8 Iterative evolution flow chart of GFLC.

3.4 Ant-Genetic Based Fuzzy Logic Controller (AGFLC)

The AGFLC selects fuzzy rules by ACO and then tunes membership functions by GA. The details of applying ACO to select fuzzy rules and iterative evolution AGFLC algorithm are elaborated as follows. The employment of applying GA to tune membership functions presented in the previous section would not be repeated here.

3.4.1 Selecting fuzzy rules by ACO

While ACO is applied to a specific problem, the following steps have to be performed (Casillas, *et al.*, 2000):

- Obtain a problem representation as a graph or a similar structure easily covered by ants.
- Define the way of assigning a heuristic preference to each choice that the ant has to take in each step to generate the solution.
- Establish an appropriate way of initializing the pheromone.
- Define the updating rules of pheromone.

These steps are performed to apply to the rules selection problem and described as follows.

(1) Problem description

For the adaptability of ACO to rules selection problem, we reformulate the problem into a clustering problem which divides an antecedent of fuzzy rules into a corresponding consequent to form a complete fuzzy rule. The network formulation is similar to the one done in Casillas *et al.* (2005) but the heuristic information on the arcs is different. The steps for this reformulation are elaborated as follows:

(a) Determine the clustering objects. The i^{th} object to be clustered is defined as the i^{th} antecedent of fuzzy rules, represented by AR_i, $i=1,\ldots,M$. M represents the total number of antecedents. Each antecedent can be composed of the conjunction of several state variables. With a RB has N state variables and M fuzzy rules, the antecedent part can be expressed by the following:

AR_1 : IF $x_1 = A_{11}$ AND $x_2 = A_{12}$ AND … AND $x_N = A_{1N}$
AR_2 : IF $x_1 = A_{21}$ AND $x_2 = A_{22}$ AND … AND $x_N = A_{2N}$
.
.
.
AR_i : IF $x_1 = A_{i1}$ AND $x_2 = A_{i2}$ AND … AND $x_N = A_{iN}$
.
.
.
AR_M : IF $x_1 = A_{M1}$ AND $x_2 = A_{M2}$ AND … AND $x_N = A_{MN}$

Taking two state variables as an example, if each linguistic variable is assumed to be five linguistic degrees, there are 25 potential antecedents in a total, as depicted in Table 3.1.

Table 3.1 Antecedents of two state variables with five linguistic degrees

x_2	x_1				
	NL	***NS***	***ZE***	***PS***	***PL***
NL	AR_1	AR_6	AR_{11}	AR_{16}	AR_{21}
NS	AR_2	AR_7	AR_{12}	AR_{17}	AR_{22}
ZE	AR_3	AR_8	AR_{13}	AR_{18}	AR_{23}
PS	AR_4	AR_9	AR_{14}	AR_{19}	AR_{24}
PL	AR_5	AR_{10}	AR_{15}	AR_{20}	AR_{25}

Note: *NL* represents negative large. *NS* represents negative small. *ZE* represents zero. *PS* represents positive small. *PL* represents positive large.

(b) Link antecedents to consequents. The potential antecedent AR_i will be linked to any one of the possible consequents, denoted C_j, j=1, 2,…, J. J is the number of linguistic degree of control variable. To exclude badly defined or conflicted rules, the antecedent AR_i could be possibly assigned to a cluster set, called exclusion set (C_{J+1}). Taking two state variables and one control variable as an example, if each variable has five linguistic degrees, a total of 25 objects (potential antecedents) can be grouped into 6 clusters, where C_j, j=1, 2,…, 5, stand for the consequents of y=NL, y=NS, y=ZE, y=PS, y=PL, respectively, and C_6 represents the exclusion set. All objects are fully connected to these 6 clusters as depicted in Figure 3-9.

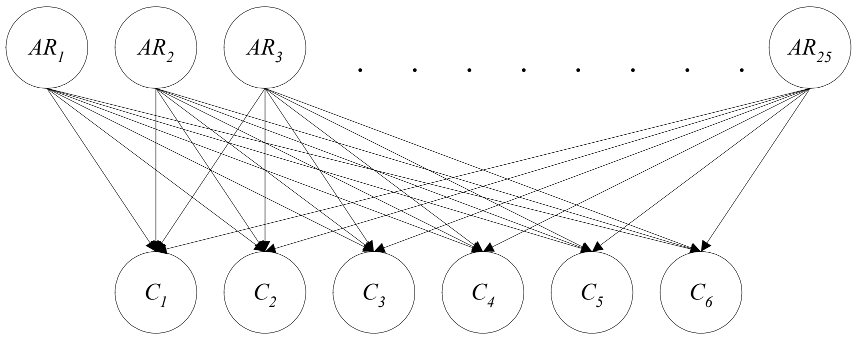


Figure 3-9 Clustering network of two state variables and one control variable (each with five linguistic degrees).

To construct a complete solution, each ant successively visits each potential antecedent and chooses a consequent with transition probability depending on the heuristic information and pheromone level. Initially, each ant is put on one antecedent. That is, the number of ants is equal to the number of antecedents. After assigning an antecedent to a consequent for each ant, the ants move to the next antecedent in parallel. Thus, the choice of each ant would be affected by the previous tour construction step because of the local pheromone update rule.

(2) Heuristic information

The heuristic information on the arc connecting the potential antecedent to consequent represents the preference of selecting a complete fuzzy rule. In this study, the reasonability of linking one antecedent to a consequent is taken as the heuristic information for tour construction. The reasonability information (θ_{ij}) is defined as the degree of similarity between the assigning result of a specific antecedent to a consequent and a predetermined assigning result. Thus, to obtain the reasonability information, a predetermined rule table must be established in advance. A higher information value shows that the selection result is more similar to the predetermined rules. For instance, assuming that AR_{21} (*i.e.* IF $x_1=PL$ and $x_2=NL$) is connected to cluster C_5 ($y=PL$) in the predetermined rule table, the reasonability information on this arc would have the highest value, followed by the arc connecting to C_4 ($y=PS$), C_3 ($y=ZE$), C_2 ($y=NS$), while the arc connecting to cluster C_1 ($y=NL$) would have the least value. In this study, the value of reasonability information on the arc connecting to cluster C_6 is preset, and this value will serve as the threshold to exclude the rules. Without loss of generality, we assume that the maximum value of reasonability information equals to 1 and that the value is decreased by $1/J$ (J stands for the number of linguistic degrees of control variable) for each additional linguistic degree gap. If an antecedent AR_i is assigned to consequent C_j, but AR_i is connected to consequent C_p in a predetermined rule table, then the reasonability information can be expressed as:

$$\theta_{ij} = 1 - \frac{|j-p|}{J} \tag{3.17}$$

where θ_{ij} represents reasonability information value on the arc connecting AR_i and C_j. Take J=5, C_j= C_2 ($y=NS$) and C_p= C_4 ($y=PS$) for instance, $\theta_{ij} = 1 - \frac{|2-4|}{5} = 0.6$.

After defining the reasonability information, ant k in antecedent r choosing consequent s can be determined by the following equations:

$$s = \arg\max_{j=1}^{J+1}\{[\theta_{rj}]^{\alpha}[\tau_{rj}]^{\beta}\}, \text{ if } q \le q_0 \text{ (exploitation),} \tag{3.18}$$

or visit s with P_{rs}^{k}, if $q > q_0$ (exploration), where

$$P_{rs}^{k} = \frac{[\theta_{rs}]^{\alpha}[\tau_{rs}]^{\beta}}{\sum_{j=1}^{J+1}[\theta_{rj}]^{\alpha}[\tau_{rj}]^{\beta}}. \tag{3.19}$$

τ_{rj} is the amount of pheromone on the arc connecting AR_r and C_j. The symbols α and β are parameters that determine the relative importance of reasonability and pheromone. q is a random number chosen randomly with uniform probability in [0,1] and q_0 $(0 \le q_0 \le 1)$ is a parameter representing the threshold to implement exploitation or exploration. P_{rs}^{k} is the probability of ant k assigning the antecedent r to consequent s when implementing exploration.

(3) Pheromone initialization

For a minimization problem, the initial pheromone value (τ^0) can be set as the reciprocal of an objective function (E) of any initial solution (namely the predetermined rule base). For a rule learning problem with an input-output training dataset, the objective function could be to minimize the error between the observed output and the output concluded by the AGFLC. For a rule learning problem without the training dataset, the objective function could be defined as the performance index of the control system.

(4) Pheromone updates

In this proposed AGFLC model, the pheromone levels on arcs are updated both locally and globally. The local pheromone update rule is applied immediately after one ant has crossed an arc (i, j) during the tour construction. It can be represented by:

$$\tau_{ij} \leftarrow (1-\xi)\tau_{ij} + \xi\tau^{0} \tag{3.20}$$

where τ^0 is the value of initial pheromone. $\xi \in (0,1)$ is a pheromone decay parameter of local update rule making the pheromone not going too far beyond τ^0.

After all ants have completed their tours, the global updating rule is to deposit a certain amount of pheromone ($\Delta\tau_{ij}$) on the arcs belonging to the best-so-far tour ($T^*(t)$) constructed by the best-so-far performed ant. The pheromone level of the t^{th} iteration is updated by:

$$\tau_{ij}(t+1) \leftarrow (1-\rho)\tau_{ij}(t) + \Delta\tau_{ij}(t) \quad \textit{if } arc(i,j) \in T^*(t) \tag{3.21}$$

where $\tau_{ij}(t)$ and $\tau_{ij}(t+1)$ are the pheromone levels of the incumbent iteration and next iteration on arc (i, j) respectively. $T^*(t)$ is the best-so-far tour constructed by the best-so-far ant till the t^{th} iteration. $\Delta\tau_{ij}(t) = 1/E^*(t)$ where $E^*(t)$ is the objective function of $T^*(t)$. Finally, $\rho \in (0,1]$ is a pheromone decay parameter of global update rule governing the evaporation of the pheromone trail.

3.4.2 Iterative AGFLC evolution algorithm

Consider an FLC with N state variables $x_1, x_2, \ldots, x_N$ and one control variable y, each with $d_1, d_2, \ldots, d_N$ and d_{N+1} linguistic degrees. Assume that the membership functions of all linguistic degrees to be triangle-shaped. The iterative AGFLC evolution algorithm is structured as follows:

Step 0: Initialization. Let v=1 where v represents the number of evolution.

Step 1: Selecting fuzzy rules by ACO.

Step 1-1: Network formulation. There are $\prod_{i=1}^{N} d_i$ potential antecedents to be divided into d_{N+1} clusters.

Step 1-2: Pheromone initialization.

Step 1-3: Tour construction. There are a total of $\prod_{i=1}^{N} d_i$ ants and each ant is put on one antecedent. Each ant chooses a consequent with transition probability depending on the heuristic information and pheromone level. After assigning an antecedent to a consequent for each ant, the ants move to the next antecedent in parallel.

Step 1-4: Pheromone updates.

Step 1-5: Incumbent tour updating. After an iteration (global updating) has been completed, the incumbent solution is tested and updated as follows: If $\min_k\{E_k(t)\} = E^+(t) < E^*(t)$, then let $E^*(t)=E^+(t)$ and $T^*(t)=T^+(t)$; otherwise $E^*(t)$ and $T^*(t)$ remain unchanged, where $E_k(t)$ is the value of the objective function of ant k of iteration t; $E^+(t)$ is the value of objective function of the best tour $T^+(t)$ of iteration t.

Step 1-6: Testing the stop condition of Step 1. If the maximal iterations $t_{\max}$ has been reached, then proceed to **Step 2**. Otherwise, go back to **Step 1-3**.

Step 2: Tuning membership functions by GA.

Step 2-1: Encoding the membership functions.

Step 2-2: Generating initial population. Randomly generate an initial population with p chromosomes. Each chromosome has $36(N+1)$ genes and each gene randomly takes one integer from the interval [0, 9].

Step 2-3: Calculating fitness values. The fitness value is set as the reciprocal of objective function of the problem to be minimized. The fitness value of each chromosome is calculated for the evaluation of the next step.

Step 2-4: Selection. Select the chromosomes for crossover and mutation by evaluating their fitness values with the Monte Carlo wheel method.

Step 2-5: Crossover.

Step 2-6: Mutation.

Step 2-7: Testing the stop condition of Step 2. The stop condition is set based on whether the mature rate has reached a given constant δ. If so, proceed to **Step 3**; otherwise, go back to **Step 2-4**.

Step 3: Testing of the stop condition. If $(f_v - f_{v-1}) \le \varepsilon$, then stop, where f_v and f_{v-1} are the best objective value for the v^{th} and v-1^{th} evolution epoch respectively and ε is an arbitrary small number. The incumbent fuzzy rules and membership functions are the compromising learning results. Otherwise, let v=v+1 and go to **Step 1**.

The iterative AGFLC evolution algorithm is portrayed as Figure 3-10.

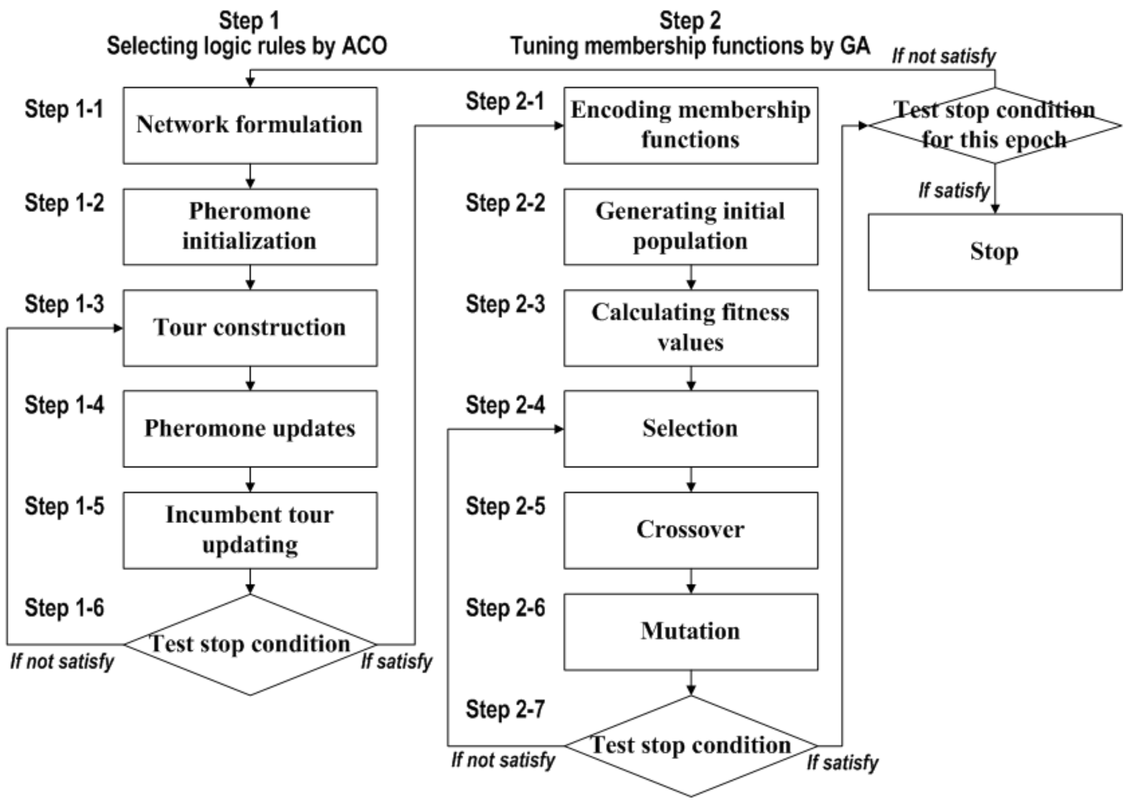


Figure 3-10 Iterative evolution flow chart of AGFLC.