เอกสารอ้างอิง

1. George R. Strakosch, Vertical Transportation: Elevators and Escalators, John Wiley & Sons, Inc., New York, 2nd ed., 1983.
2. Benjamin Stein, John S. Reynolds and William J. McGuinness, " Vertical Transportation: Passenger Elevators," Mechanical Equipment for Buildings, Vol. 2, pp. 1141-1161, Wiley, New York, 7th ed., 1986.
3. กฤษดา วิศวธีรานนท์, " การออกแบบระบบควบคุมกลุ่มลิฟท์โดยสาร," การประชุมทางวิชาการและนิทรรศการ การประยุกต์ใช้คอมพิวเตอร์ในทางวิศวกรรม, หน้า 75-90, คณะวิศวกรรมศาสตร์ มหาวิทยาลัยสงขลานครินทร์, 2531.
4. กฤษดา วิศวธีรานนท์, "การออกแบบระบบควบคุมกลุ่มลิฟท์โดยสาร," วิศวกรรมสาร, ปีที่ 41 เล่มที่ 4, หน้า 55-60, 2531.
5. A. J. Miller and A. J. Wareing, " A Microprocessor Network for Use in Control of Lifts," GEC Journal of Science & Technology, Vol. 46, No. 1, pp. 11-18, 1980.
6. วรวุฒิ จิตขจรวานิช, " อัลกอริธึม การควบคุมกลุ่มลิฟท์โดยสาร," วิศวกรรมศาสตร์ (ฉบับนิทรรศการวิชาการทางวิศวกรรม ครั้งที่ 8), พฤศจิกายน 2530.
7. นรังสรรค์ วิไลสกุลยง, " การพัฒนาเครื่องควบคุมลิฟท์โดยสารเดี่ยว โดยใช้ไมโครคอมพิวเตอร์," วิทยานิพนธ์ปริญญามหาบัณฑิต ภาควิชาวิศวกรรมศาสตร์ บัณฑิตวิทยาลัย จุฬาลงกรณ์มหาวิทยาลัย, 2530.
8. นรังสรรค์ วิไลสกุลยง และ กฤษดา วิศวธีรานนท์, " เครื่องควบคุมลิฟท์โดยสารเดี่ยว โดยใช้ไมโครคอมพิวเตอร์," การประชุมวิชาการทางวิศวกรรมไฟฟ้า 8 สถาบันอุดมศึกษาแห่งประเทศไทย ครั้งที่ 9, เล่มที่ 2, หน้า 4.4.1-4.4.16, คณะวิศวกรรมศาสตร์ มหาวิทยาลัยขอนแก่น, 2529.

9. James A. Payne, Introduction to Simulation: Programming Techniques and Methods of Analysis, McGraw-Hill Book Co., New York, 1988.

10. George T. Hummet, Thomas D. Moser and Bruce A. Powell, "Real Time Simulation of Elevators," Simulation Conference, Vol. 2, pp. 393-402, 1978.

11. Watanabe, Urahara, " Speed Control and Group Control of High Speed Elevator," System and Control, Vol. 22, No. 6, pp. 316-323, 1978.

12. K. Hirasawa, S. Kuzunuki, T. Iwasaka and T. Kaneko, " Optimal Hall Call Assignment Method of Elevator Group Supervisory Control System," Proceeding Automation Control Conference, Vol. 1, pp. 305-313, 1978.

13. วรวุฒิ จิตชจรวานิช และ กฤษดา วิศวธีรานนท์, " วิธีการเลือกส่งลิฟท์ไปรับการเรียกในระบบควบคุมกลุ่มลิฟท์," การประชุมวิชาการทางวิศวกรรมไฟฟ้า 9 สถาบันอุดมศึกษาแห่งประเทศไทย ครั้งที่ 11, เล่มที่ 2, หน้า 4.4.1-4.4.13, คณะวิศวกรรมเทคโนโลยี สถาบันเทคโนโลยีราชมงคล, 2531.

14. G. C. Barney, Elevator Technology, Ellis Horwood Limited, Chichester, England, 1986.

15. Mitsubishi Electric Corporation, Computer Controller Passenger Elevators, 1982.

16. Hitachi Co. Ltd., Elevator Control by Microcomputer, 1981.

17. Toshiba Corporation, Group Supervisory System Command, 1982.

18. KONE Corporation, TMS-900 Training Manual, 1984.

19. พฤกษ์ พูลเกษม, " Lift Controller (Hard Ware)," เอกสารประกอบโครงงานวิศวกรรมไฟฟ้า ภาควิชาวิศวกรรมไฟฟ้า คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์ มหาวิทยาลัย, 2531.

20. ประสงค์ ฉัตรศิริสกุล, " การสื่อสารข้อมูล Current Loop," เอกสารประกอบโครงงานวิศวกรรมไฟฟ้า ภาควิชาวิศวกรรมไฟฟ้า คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย, 2531.

21. Athanasios Papoulis, Probability, Random Variables, and Stochastic Processes, McGraw-Hill Book Co., New York, 1965.

22. R. E. Shannon, " System Simulation: The Art and Science," Prentice-Hall, Englewood Cliffs, N.J., 1975, quited in James A. Payne, Introduction to Simulation: Programming Techniques and Methods of Analysis, pp. 2, McGraw-Hill Book Co., New York, 1988.

ภาคผนวก

ภาคผนวก ก

วิธีการเลือกส่งลิฟต์และโครงสร้างระบบกลุ่มลิฟต์ ของบริษัทผู้ผลิตต่าง ๆ

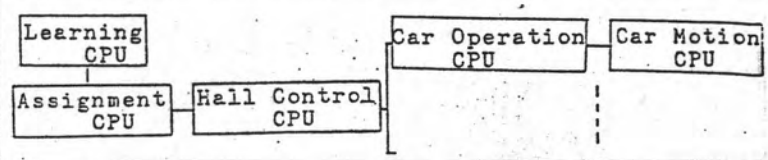
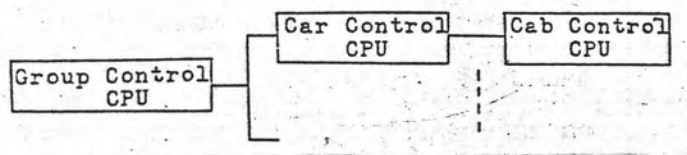
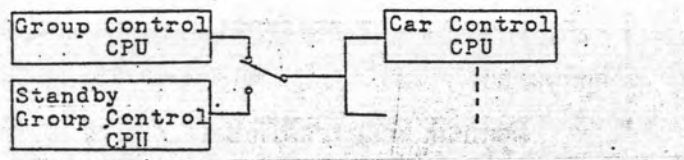
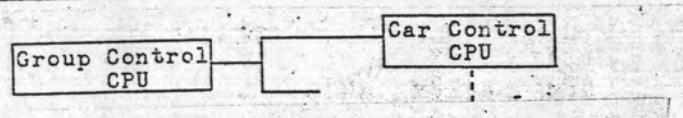
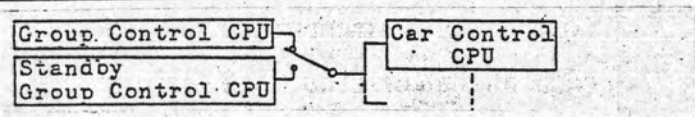
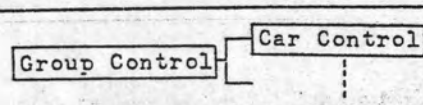
ตาราง ก.1 วิธีการเลือกส่งลิฟต์แบบต่างๆ

ผู้ผลิต	ชื่อระบบควบคุม	วิธีการเลือกส่งลิฟต์
MISUBISHI	OS-2100C	ใช้การคำนวนตามหลักจิตวิทยาการรอลิฟต์ (Psychological Waiting Time) โดยลิฟต์ที่ถูกเลือกจะไม่เป็นตัวเพิ่มเวลารอลิฟต์ของชั้นอื่นด้วย การเลือกลิฟต์จะเลือกลิฟต์ที่ให้ค่า ผลรวมของเวลารอลิฟต์ยกกำลังสองน้อยที่สุด
OTIS	ELEVONIC-401	ใช้การคำนวนผลต่างเวลาของการตอบสนองระบบ (Relative System Response Time) โดยลิฟต์ที่เลือกจะให้เวลารอลิฟต์น้อย สามารถบริการชั้นที่กดเรียกติดต่อกันได้ดี และให้ความสำคัญแก่ลิฟต์ที่จำเป็นต้องจอดที่ชั้นที่มีการกดเรียกอยู่แล้ว
HITACHI	CIP-3800	MIN-MAX Call Assignment คำนวนเวลารอลิฟต์ของ Assigned Hall Call ของลิฟต์ทุกตัว และเลือกลิฟต์ที่ให้ค่าเวลานี้ น้อยที่สุด

ตาราง ก.1 (ต่อ)

ผู้ผลิต	ชื่อระบบควบคุม	วิธีการเลือกส่งลิฟต์
TOSHIBA	COMMAND-6000	ใช้การกระจายเวลารอลิฟต์ โดยเลือกลิฟต์ที่ไม่ทำให้เกิดการรอลิฟต์นานมาก หรือน้อยเกินไป ซึ่งจะทำให้เวลารอลิฟต์อยู่ระหว่าง 10-20 นาที
FUJITEC	FLEX-11	ใช้คำนวนเวลารอลิฟต์ของการกดเรียก Hall Call ปัจจุบัน, Hall Call ที่เลือกลิฟต์ไปแล้ว และทำนายการเกิด Hall Call ใหม่ แล้วใช้ค่าคำนวนได้ เป็นการตัดสินใจเลือกลิฟต์
SCHINDER	VARIOMATIC 2	ใช้การเปลี่ยนแปลงเขตบริการของลิฟต์ (Service Zone) โดยลิฟต์ที่ถูกเลือกจะต้องเป็นลิฟต์ที่ถูกกำหนดให้อยู่ในเขตบริการของตนเอง และการเปลี่ยนแปลงเขตบริการจะเกิดขึ้นอยู่ตลอดเวลา เมื่อลิฟต์มีการเปลี่ยนชั้นหรือเปลี่ยนทิศทางการเคลื่อนที่
หมายเหตุ: วิธีการเลือกลิฟต์ที่รวบรวมมานี้ได้มาจากเอกสารโฆษณาซึ่งให้รายละเอียดได้จำกัด จึงเป็นการยากที่จะเปรียบเทียบวิธีการต่างๆ ว่ามีข้อแตกต่างหรือคล้ายคลึงกันอย่างไร		

ตาราง ก.2 โครงสร้างระบบควบคุมลิฟต์แบบต่างๆ

ผู้ผลิต	ชื่อระบบควบคุม	โครงสร้างระบบควบคุม
MISUBISHI	OS-2100C	
OTIS	ELEVONIC-401	
HITACHI	CIP-3800	
TOSHIBA	COMMAND-6000	
FUJITEC	FLEX-11	
SCHINDER	VARIOMATIC 2	 The system doesn't use the micro-processor

ภาคผนวก ข

โปรแกรมคำนวณเพื่อเลือกส่งลิฟต์เมื่อมีการกดเรียก

ข.1 โปรแกรมคำนวณเพื่อเลือกส่งลิฟต์แบบ Mean Waiting Time Minimization Method

```
/*****************************************************************************/
/*                                                                           */
/*                Calculating Waiting Time Parameter                         */
/*                                                                           */
/*             Mean Waiting Time Minimization Method                         */
/*                                                                           */
/*****************************************************************************/


/* Grobal Variable */

int ext_hall_call[MAX_2FLOOR],ext_car_call[MAX_2FLOOR];
/* extended hall call array and car call array */
/* The first half represents the calls' status in UP direction
   and the second half represents the calls' status in DOWN direction.
   So, the total elements equal (2 * total floor) -2 */


/* Program */

/* Calculate parameter 'Q(i,f)' of lift i
   for a new hall call up at floor 'f'  */
long gup_time(i,f)
int i,f;                    /* Lift i and floor f */
{  int point1,point2;       /* Starting point of the 1st and 2nd path to
                               calculate. */
   long para;               /* The result parameter */
   long mean_wait();        /* Function to calculate the parameter
                               with Mean Waiting Time Minimization Method. */

/* Define ext_hall_call[] and ext_car_call[] of lift i. */
   extend_call(i);
```

```
/* Define starting point of the 1st and 2nd path. */
   if (drt[i]==0) {
      point1=psn[i];              point2=f;
   } else {
      point1=amt_route-psn[i];    point2=f;
   }

/* Calculate parameter Q(i,f) with Mean Waiting Time Minimization Method */
   para=mean_wait(point1,point2);

/* Return the result */
   return(para);
}



 /* Calculate parameter 'Q(i,f)' of lift i
    for a new hall call down at floor 'f'  */
 long gdw_time(i,f)
 int i,f;
 {  int point1,point2;
    long para,mean_wait();
    extend_call(i);
    if (drt[i]==0) {
       point1=psn[i];
       point2=amt_route-f;
    } else {
       point1=amt_route-psn[i];
       point2=amt_route-f;
    }
    para=mean_wait(point1,point2);
    return(para);
 }


 /* Define ext_hall_call[] and ext_car_call[] of lift i. */
 extend_call(i)
 int i;
 { int fl,ffl;

 /* for 1st half of ext_hall_call[] and ext_car_call[] */
   ffl=0;
   for (fl=0; fl<amt_endfl; ++fl,++ffl) {
       ext_hall_call[ffl]=assnup[i][fl];
       if (drt[i]==0)
          ext_car_call[ffl]=car_call[i][fl];
       else
          ext_car_call[ffl]=0;
   }
```

```
/* for 2nd half of ext_hall_call[] and ext_car_call[] */
  for ( ; fl>0; --fl, ++ffl) {
      ext_hall_call[ffl]=assndw[i][fl];
      if (drt[i]==1)
         ext_car_call[ffl]=car_call[i][fl];
      else
         ext_car_call[ffl]=0;
  }
}
```

```
/* Caculate with Mean Waiting Time Minimization Method */
long mean_wait(p1,p2)
int p1,p2;                  /* The starting point of 2 path */
{  int point;               /* Working variable */
   long para1,para2;        /* Parameter caculated in path 1 and 2 */
   if (p1==p2)              /* If p1 and p2 is the same point, */
      return(0);            /* return zero. */
   para1=para2=0;           /* Initial para1 and para2 */

/* Calculate para1. */
   point=p1;                /* Set the working point to be p1 */
   while (point!=p2) {      /* Do until point equal p2 */
      para1+=amt_mclk;      /* Add the running time between successive floor
                               to para1 */
      if (ext_hall_call[point]==1 || ext_car_call[point]==1)
         para1+=amt_stop;   /* If found car call or hall call in the 1st path,
                               add the stopping time to para1 */
      ++point;              /* Increment point */
      if (point>amt_route)  /* If point exceed total distance,  */
         point=0;           /* set point to be zero. */
      }

/* Calculate para2. */
   point=p2+1;              /* Set the working point to be p2+1 */
   while (point!=p1) {      /* Do until point equal p1 */
      if (ext_hall_call[point]==1)
         para2+=amt_stop;   /* If found hall call in the 2st path,
                               add the stopping time to para2 */
      ++point;              /* Increment point */
      if (point>amt_route)  /* If point exceed total distance,  */
         point=0;           /* set point to be zero. */
   }

/* Return the result */
   return(para1+para2);
}
```

ข.2 โปรแกรมคำนวณเพื่อเลือกส่งลิฟต์แบบ Long Waiting Time Minimization Method

```
/*****************************************************************************/
/*                                                                           */
/*              Calculating Waiting Time Parameter                           */
/*                                                                           */
/*            Long Waiting Time Minimization Method                          */
/*                                                                           */
/*****************************************************************************/


/* Grobal Variable */

int ext_hall_call[MAX_2FLOOR],ext_car_call[MAX_2FLOOR];
/* extended hall call array and car call array */
int ext_wait[MAX_2FLOOR];  /* Last waiting time */
/* The first half represents the calls' status in UP direction
   and the second half represents the calls' status in DOWN direction.
   So, the total elements equal (2 * total floor) -2 */


/* Program */

/* Calculate parameter 'Q(i,f)' of lift i
   for a new hall call up at floor 'f'  */
long gup_time(i,f)
int i,f;
{  int point1,point2;       /* Starting point of the 1st and 2nd path to
                               calculate. */
   long para;               /* The result parameter */
   long long_wait();        /* Function to calculate the parameter
                               with Long Waiting Time Minimization Method. */
/* Define ext_hall_call[] and ext_car_call[] of lift i. */
   extend_call(i);
/* Define starting point of the 1st and 2nd path. */
   if (drt[i]==0) {
      point1=psn[i];              point2=f;
   } else {
      point1=amt_route-psn[i];    point2=f;
   }
/* Calculate parameter Q(i,f) with Long Waiting Time Minimization Method */
   para=long_wait(point1,point2);
/* Return the result */
   return(para);
}
```

```
/* Calculate parameter 'Q(i,f)' of lift i
   for a new hall call down at floor f  */
long gdw_time(i,f)
int i,f;                    /* lift i and floor f */
{  int point1,point2;
   long para,mean_wait();
   extend_call(i);
   if (drt[i]==0) {
      point1=psn[i];
      point2=amt_route-f;
   } else {
      point1=amt_route-psn[i];
      point2=amt_route-f;
   }
   para=long_wait(point1,point2);
   return(para);
}


/* Define ext_hall_call[] and ext_car_call[] of lift i. */
extend_call(i)
int i;
{ int fl,ffl;

/* for 1st half of ext_hall_call[] and ext_car_call[] */
  ffl=0;
  for (fl=0; fl<amt_endfl; ++fl,++ffl) {
      ext_hall_call[ffl]=assnup[i][fl];
      if (drt[i]==0)
         ext_car_call[ffl]=car_call[i][fl];
      else
         ext_car_call[ffl]=0;
  }

/* for 2nd half of ext_hall_call[] and ext_car_call[] */
  for ( ; fl>0; --fl, ++ffl) {
      ext_hall_call[ffl]=assndw[i][fl];
      if (drt[i]==1)
         ext_car_call[ffl]=car_call[i][fl];
      else
         ext_car_call[ffl]=0;
  }
}
```

```
/* Caculate with Long Waiting Time Minimization Method */
long long_wait(p1,p2)
int p1,p2;                  /* The starting point of 2 paths */
{  int point;
   long count_time,max_time,total_wait;  /* working variables */
   if (p1==p2)              /* If p1 and p2 is the same point, */
      return(0);            /* return zero. */
   count_time=0;            /* Set time counter to zero */

/* 1st loop */
   point=p1;                /* Set the working point to be p1 */
   while (point!=p2) {      /* Do until point equal p2 */
      count_time+=amt_mclk; /* Add the running time between successive floor
                               to count_time */
      if (ext_hall_call[point]==1 || ext_car_call[point]==1)
         count_time+=amt_stop; /* If found car call or hall call in the 1st
                                  path, add the stopping time to count_time */
      ++point;              /* Increment point */
      if (point>amt_route)  /* If point exceed total distance,  */
         point=0;           /*    set point to be zero. */
   }
   max_time=count_time;     /* Set max_time to be count_time */

/* 2nd loop */
   point=end;               /* Set point to be end */
   while (point!=start) {   /* Do until point equal start */
      if (ext_hall_call[point]==1) {     /* If found hall call in the 2nd
                                            path, calculate total_wait. */
         /* total_wait is the sum of count_time and the past waiting time */
         total_wait=count_time+ext_wait[point];
         if (total_wait>max_time) {      /* If total_wait is greater than */
            max_time=total_wait;            max_time, max_time is total_wait */
         }
      }
      count_time+=amt_mclk;   /* Add the running time between successive floor
                                 to count_time */
      if (ext_hall_call[point]==1 || ext_car_call[point]==1)
         count_time+=amt_stop; /* If found car call or hall call in the 1st
                                  path, add the stopping time to count_time */
      ++point;              /* Increment point */.
      if (point>amt_route)  /* If point exceed total distance,  */
         point=0;           /*    set point to be zero. */
   }

/* return result */
   return(max_time);        /* return max_time */
}
```

ข.3 โปรแกรมเลือกส่งลิฟต์ที่เพิ่มส่วนการคำนวณค่าประเมินการจอด และค่าประเมินการผ่านชั้นต่างๆ

```
/*****************************************************************************/
/*                                                                           */
/*                Calculating Waiting Time Parameter                         */
/*                                                                           */
/*   Mean Waiting Time Minimization Method & Predicted STOPS & PATHS         */
/*                                                                           */
/*****************************************************************************/


/* Grobal Variable */

......
......
float p_stop[MAX_2FLOOR],p_path[MAX_2FLOOR];

/* predicted stop & predicted path array */
/* The first half represents the calls' status in UP direction
   and the second half represents the calls' status in DOWN direction.
   So, the total elements equal (2 * total floor) -2 */


/* Program */

/* Calculate parameter 'Q(i,f)' of lift i
   for a new hall call up at floor 'f'  */
long gup_time(i,f)
int i,f;
{
   ........
   ........
   extend_call(i);
   ........
   ........
   para=mean_wait(point1,point2);
   ........
   ........
}

/* Calculate parameter 'Q(i,f)' of lift i
   for a new hall call down at floor 'f'  */
long gdw_time(i,f)
int i,f;
{  .......
   extend_call(i);
   .......
   para=mean_wait(point1,point2);
   .......
}
```

```
/* Define ext_hall_call[] and ext_car_call[] of lift i. */
extend_call(i)
int i;
{ int fl,ffl,m,n,far_hall;
  float g,r,q,p[MAX_FLOOR];
/* find highest hall call */
  fl=amt_endfl;
  while (assnup[i][fl]==0 && fl>=0)
     --j;
  far_hall=j;

/* for the up direction */
  m=ffl=0;     g=q=1.0;
  for (fl=0; fl<amt_endfl; ++fl,++ffl) {
      ext_hall_call[ffl]=assnup[i][fl];

      /* Find predicted stops. */
      if (assnup[i][fl] || (car_call[i][fl] && drt[i]==0))
         p_stop[ffl]=1.0;
      else
         p_stop[ffl]=1.0-g;
      if (assnup[i][fl]) {
         r=amt_endfl-fl;
         g*=(r-1.0)/r;
      }

      /* Find predicted paths. */
      if (fl<=far_hall) {
         p_path[ffl]=1.0;
         if (aasnup[fl]) {
            q*=(amt_endfl-fl);
            p[m]=far_hall-fl;
            ++m;
         }
      } else {
         p_parh[ffl]=1.0;
         for (n=0; n<m; ++n) {
            ++p[n];
            p_path[ffl]*=p[n];
         }
         p_path[ffl]=1.0-p_path[ffl]/q;
      }
   }
```

```
/* find lowest hall call */
  fl=0;
  while (assnup[i][fl]==0 && fl<=amt_endfl)
     ++j;
  far_hall=j;

/* for the down direction */
  m=0;     g=q=1.0;
  for (fl=amt_endfl; fl>0; --fl,++ffl) {
      ext_hall_call[ffl]=assndw[i][fl];

      /* Find predicted stops. */
      if (assndw[i][fl] || (car_call[i][fl] && drt[i]==1))
         p_stop[ffl]=1.0;
      else
         p_stop[ffl]=1.0-g;
      if (assndw[i][fl]) {
         r=fl;
         g*=(r-1.0)/r;
      }

      /* Find predicted paths. */
      if (fl>=far_hall) {
         p_path[ffl]=1.0;
         if (aasndw[fl]) {
            q*=fl;
            p[m]=fl-far_hall;
            ++m;
         }
      } else {
         p_parh[ffl]=1.0;
         for (n=0; n<m; ++n) {
            ++p[n];
            p_path[ffl]*=p[n];
         }
         p_path[ffl]=1.0-p_path[ffl]/q;
      }
   }
}
```

```
/* Caculate with Mean Waiting Time Minimization Method */
long mean_wait(p1,p2)
int p1,p2;                  /* The starting point of 2 path */
{  int point;               /* Working variable */
   long para1,para2,        /* Parameter caculated in path 1 and 2 */
        add_stop;           /* Stop time to add in path 2 */
   if (p1==p2)              /* If p1 and p2 is the same point, */
     return(0);             /* return zero. */
   para1=para2=0;           /* Initial para1 and para2 */

/* Calculate para1. */
   point=p1;                /* Set the working point to be p1 */
   while (point!=p2) {      /* Do until point equal p2 */

      /* Add the running time between successive floor to para1 */
      para1+=(amt_mclk*p_path[point]);   /* Weighting with p_path[] */
      para1+=(amt_stop*p_stop[point]);   /* add the stopping time to para1 */
                                         /* weighting with p_stop[] */
      ++point;              /* Increment point */
      if (point>amt_route)  /* If point exceed total distance,  */
        point=0;            /* set point to be zero. */
      }

/* Calculate para2. */
   add_stop=amt_stop*p_stop[p2];
   point=p2+1;              /* Set the working point to be p2+1 */
   while (point!=p1) {      /* Do until point equal p1 */
     if (ext_hall_call[point]==1)
       para2+=add_stop;     /* If found hall call in the 2st path,
                               add the stopping time to para2 */
     ++point;               /* Increment point */
     if (point>amt_route)   /* If point exceed total distance,  */
       point=0;             /* set point to be zero. */
   }

/* Return the result */
   return(para1+para2);
}
```

ประวัติผู้เขียน

นาย วรวุฒิ จิตขจรวานิช เกิดเมื่อวันที่ 22 สิงหาคม พ.ศ. 2503 ที่กรุงเทพมหานคร สำเร็จการศึกษาขั้นปริญญาบัณฑิตในสาขาวิชาเชิงเลข ภาควิชาวิศวกรรมไฟฟ้า คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย เมื่อปี พ.ศ. 2527 เข้าศึกษาต่อขั้นปริญญามหาบัณฑิตในสาขาวิชาวิศวกรรมไฟฟ้า ภาควิชาวิศวกรรมศาสตร์ บัณฑิตวิทยาลัย จุฬาลงกรณ์มหาวิทยาลัย ในปี พ.ศ. 2528 จนถึงปัจจุบัน

ในระหว่างการศึกษาในขั้นปริญญามหาบัณฑิต ได้ส่งผลงานการวิจัยเรื่อง "วิธีการเลือกส่งลิฟท์ไปรับการเรียกในระบบควบคุมกลุ่มลิฟท์" เข้าร่วมในการประชุมวิชาการทางวิศวกรรมไฟฟ้า 9 สถาบันอุดมศึกษาแห่งประเทศไทย ครั้งที่ 11 เมื่อปี พ.ศ. 2531 และผลงานการวิจัยบางส่วนเกี่ยวกับงานวิจัยวิทยานิพนธ์นี้ ได้เคยลงพิมพ์ในวารสารทางวิชาการดังที่ปรากฏในเอกสารอ้างอิงของรายงานวิทยานิพนธ์ฉบับนี้