Collecting Paired-Associate Learning Data Using Computer

Dr. Teara Archwamety
University of Nebraska at Kearney
Dr. Charnchai Archwamety
King Mongkut's Institute of Technology, Thonburi

Abstract

The purpose of the present paper was to present a computer program written in C language for collecting "paired-associate learning" data. First, the importance of paired-associate learning was described. Next, a brief history of data collection for paired-associate learning was given. Then, the complete program itself was presented and explained. Finally, suggestions for using and modifying this program in various possible research projects were made.

การเก็บข้อมูลการเรียนคู่สัมพันธ์โดยคอมพิวเตอร์

ศาสตราจารย์ ดร. ธีระ อาชวเมธี
ผู้ช่วยศาสตราจารย์ ดร. ชาญชัย อาชวเมธี

ความสำคัญของการเรียนคู่สัมพันธ์ (Paired-Associate Learning)

ทฤษฎีการเรียนรู้นั้นแบ่งได้เป็นสองกลุ่มใหญ่ คือ ทฤษฎีพฤติกรรม(Behavioral Theories) และทฤษฎีมโนทัศน์ (Cognitive Theories) (Bigge, 1982, p.48; Lefrancois, 1994, p.86) ทฤษฎีพฤติกรรมนั้นมีความเห็นว่าการเรียนรู้อะไรก็ตามสามารถวิเคราะห์เป็นการสร้างความสัมพันธ์ระหว่างสิ่งเร้า (stimulus) และการตอบสนองที่ถูกต้อง (correct response) ตัวอย่างเช่น ทีแรกเด็กทารกเห็นหนังสือเล่มหนึ่ง (สิ่งเร้า) ก็อาจออกเสียงอ้อแอ้อ้อแอ้ (การตอบสนอง) แต่ทว่าการตอบสนองนี้ไม่ใช่การตอบสนองที่ถูกต้อง ต่อมาทารกเห็นเล่มหนังสือ (สิ่งเร้า) แล้วออกเสียง "หนังสือ" (การตอบสนอง) เพราะมารดาสอน เราก็กล่าวได้ว่าความสัมพันธ์ระหว่างสิ่งเร้าและการตอบสนองได้เกิดขึ้น ทารกได้เรียนรู้ การเรียนคู่สัมพันธ์เกิดขึ้นตลอดชีวิต เช่น เมื่อทารกโตขึ้นเข้าโรงเรียนก็เรียนรู้ 1+1 (สิ่งเร้า) เป็น 2 (การตอบสนองที่ถูกต้อง) เมื่อทารกโตเป็นหนุ่มเข้ามหาวิทยาลัยก็เรียนรู้ว่า คำถาม "ใครเป็นบิดาจิตวิทยาการศึกษา" (สิ่งเร้า) นั้น คำตอบ ที่ถูกคือ "อี แอล ธอร์นไดค์" (การตอบสนอง) ตามทฤษฎีพฤติกรรมการเรียนรู้ทุกชนิดสามารถวิเคราะห์เป็นการสร้างความสัมพันธ์ระหว่างสิ่งเร้าและการตอบสนองที่ถูกต้อง สำหรับทฤษฎีมโนทัศน์นั้นจะไม่นำมากล่าวในบทความฉบับนี้ ขอแนะให้ท่านผู้อ่านที่สนใจทฤษฎีมโนทัศน์อ่านจากบิกก์และเลอฟรังซัว(Bigge, 1982; Lefrancois, 1994)

การเก็บข้อมูลการเรียนคู่สัมพันธ์ (อดีตถึงปัจจุบัน)

การเก็บข้อมูลการเรียนคู่สัมพันธ์โดยทั่วไปใช้วิธี "คาดล่วงหน้า" (Anticipation Method) ในวิธีดังกล่าว ผู้สอนเสนอสิ่งเร้า ผู้เรียนพยายามตอบ ตามด้วยผู้สอนเสนอสิ่งเร้าคู่กับการ

ตอบสนองที่ถูกต้อง ครั้งแล้วครั้งเล่า จนผู้เรียนตอบได้ถูกหมดสองครั้งติดต่อกัน (Jung, 1968, p. 43)

ก่อนที่เทคโนโลยีเจริญการเก็บข้อมูลการเรียนคู่สัมพันธ์ในการทำวิจัยทำกันในรูปบัตรคำ เช่น ในการเรียนศัพท์ภาษาอังกฤษ ศัพท์ภาษาอังกฤษ (สิ่งเร้า) อาจพิมพ์อยู่ด้านหนึ่งของบัตร และศัพท์ภาษาอังกฤษพร้อมคำแปลไทย (การตอบสนองที่ถูกต้อง)พิมพ์อยู่อีกด้านหนึ่งของบัตร ผู้เรียนดูคำภาษาอังกฤษ (สิ่งเร้า) แล้วพยายามพูดคำแปลที่ถูกต้อง (การตอบสนอง) ส่วนผู้สอนก็พลิกบัตรให้ดูคำตอบที่ถูกต้องการใช้บัตรคำมีข้อดีคือทำง่าย แต่ข้อเสียคือผู้สอนไม่สามารถควบคุมความเร็วในการเสนอและพลิกบัตรคำได้แม่นตรง เช่นผู้สอนอยากจะให้ผู้เรียนดูศัพท์ภาษาอังกฤษเพียง 3 วินาที และคำแปลไทย 2 วินาที ทำด้วยมือจะเที่ยงตรงได้อย่างไร

เมื่อเทคโนโลยีก้าวหน้าขึ้นข้อเสียดังกล่าวก็แก้ไขโดยการใช้เมมโมรี่ดรัม (memory drum) แทนบัตรคำ เมมโมรี่ดรัมเป็นเครื่องกลที่มีล้อขนาดหนาเป็นส่วนประกอบสำคัญ ผู้วิจัยพิมพ์สิ่งเร้าและสิ่งเร้ากับการตอบสนองที่ถูกต้องเป็นคู่ๆไปบนกระดาษซึ่งพันไปบนล้อ เครื่องควบคุมเวลาของเครื่องกลนี้จะหมุนล้อให้สิ่งเร้าหรือสิ่งเร้ากับการตอบสนองไปปรากฏที่ช่องหน้าต่างเล็ก ๆ ตามเวลาที่ตั้งไว้ เช่น สิ่งเร้า 3 วินาทีตามด้วยสิ่งเร้ากับการตอบสนองที่ถูกต้องอีก 2 วินาทีก่อนเสนอคู่ต่อไป หลังจากผู้เรียนดูการเสนอโดยไม่ปริปากไปหนึ่งรอบ จากรอบที่สองเป็นต้นไปผู้เรียนจะต้องพยายามเอ่ยคำตอบสนองที่ถูกต้องก่อนคำตอบจะปรากฏ ผู้วิจัยเป็นผู้เก็บคะแนน ท่านผู้อ่านดูตัวอย่างรูปเมมโมรี่ดรัมได้จากฮิลการ์ด (Hilgard, Atkinson & Atkinson 1975, p. 213) และเฮเบอร์ (Haber & Fried 1975, p. 10) ส่วนตัวอย่างการใช้เมมโมรี่ดรัมในการวิจัยจริง ๆ อ่านได้จากคอปเปนนาล (Koppennaal, 1963)

แม้เมมโมรี่ดรัมจะควบคุมเวลาการเสนอสิ่งเร้าและการตอบสนองได้แม่นตรงแต่ก็ประสบปัญหาที่ว่าผู้วิจัยต้องคอยสังเกตคำตอบของผู้เรียนและบันทึกคะแนนด้วยมือ นอกจากนั้นผู้วิจัยประสบปัญหาไม่สามารถบันทึกความเร็วของการตอบได้แม่นตรง ข้อเสียเหล่านี้แก้ไขได้โดยการใช้คอมพิวเตอร์ในสมัยปัจจุบัน ต่อไปนี้เป็นโปรแกรมคอมพิวเตอร์ที่ผู้เขียนเขียนขึ้นเพื่อใช้เก็บข้อมูลการเรียนคู่สัมพันธ์

ตัวคอมพิวเตอร์โปรแกรม

```
#include <stdio.h>
#include <dos.h>
#include <sys\types.h>
#include <sys\timeb.h>

/**** 1. define parameters *****/

#define PAIRS        8    /* set number of pairs to learn */
#define SINTV      300    /* set stimulus interval to 3 seconds */
#define SRINTV     200    /* set stimulus-response interval to 2 sec's */
#define MAXTRIALS   30    /* set maximum number of trials */

int seqvec[PAIRS];        /* sequence vector created in function
                             will be used by main( ) also */

main( )
{
    FILE *fp;                          /* results file */
    register int i, j;                 /* fast counter */
    int dex;                           /* index of which pair to present */
    char ans, ascnum[20], idcode[10];  /* ans for answer 0-9,
                                          ascn for stopper */
    int *seqsto, *hitsto, *timsto, *score;  /* declaring *elem[#R's][#C's]
                                               will cause stack overflow */
    long time1, time 2, seconds;
```

```
/**** 2. stimulus and correct response setting *****/
    static char *stim[PAIRS] =
                    { "0 0 0",
                      "0 0 1",
                      "0 1 0",
                      "0 1 1",
                      "1 0 0",
                      "1 0 1",
                      "1 1 0",
                      "1 1 1" };
    static char *resp[PAIRS] =
                    { "0",
                      "1",
                      "2",
                      "3",
                      "4",
                      "5",
                      "6",
                      "7" };
    int khflag=0;                   /* key-hit flag */
    int trial=0;                    /* init trial number */
    seqsto = (int*) malloc(MAXTRIALS*PAIRS*sizeof(int));
    hitsto = (int*) malloc(MAXTRIALS*PAIRS*sizeof(int));
    timsto = (int*) malloc(MAXTRIALS*PAIRS*sizeof(int));
    score  = (int*) malloc(MAXTRIALS*sizeof(int));
    ans = ' ';                      /* init answer to blank */
    clrscrn( );
```

```
/**** 3. start with learner's ID code *****/
    printf("Please enter your ID code: ");
        gets(idcode);
    printf("\nLearning task will start in about 5 seconds!");
    srand(time(&seconds));        /* seed rand gen */
    while (trial < MAXTRIALS) {
        gettime(&time1);
/**** 3.1 call random sequencer function *****/
    seqer(trial, seqsto);
    gettime(&time2);
    while (time2-time1 < SINTV+SRINTV)
        gettime(&time2);        /* delay 5 secs while resequencing */
    for (dex=0; dex<PAIRS; dex++) {
        clrscrn( );
        clrkh( );
        setcur(10, 30, 0);
        printf("%s = ", stim[seqvec[dex]]);
        gettime(&time1);
        gettime(&time2);
        while (time2-time1 < SINTV) {
            if(kbhit( )) {
                ans = getch( );
                printf("%c", ans);
                if (ans==resp[seqvec[dex]][0]) {
                    printf("   right!");
                    hitsto[trial*PAIRS + dex] = 1;
                    timsto[trial*PAIRS + dex] = time2-time1;
                    khflag = 1;          /* key-hit flag */
```

```
            }
            else {
                hitsto[trial*PAIRS + dex] = 0;
                timsto[trial*PAIRS + dex] = time2-time1;
                khflag = 1;
            }
            break;
        }
        else
            gettime(&time2);
    }
    while (time2-time1 < SINTV)
        gettime(&time2);
    if (khflag==1)
        khflag = 0;              /* reset flag */
    else {
        hitsto[trial*PAIRS + dex] = 0;
        timsto[trial*PAIRS + dex] = SINTV;
    }
    clrkh( );
    setcur(12, 30, 0);
    printf('%s = %s', stim[seqvec[dex]], resp[seqvec[dex]]);
    while (time2-time1 < SINTV+SRINTV) {
        gettime(&time2);
    }
}                    /* this is for loop closing bracket */
clrscrn( );
/* compute trial score */
score[trial] = 0;
```

```
for (i=0; i<PAIRS; i++) {
    if (hitsto[trial*PAIRS+i] == 1)
        score[trial]++;
}

/* determine trials to crit then advance trial # */
setcur(20,1,0);
printf("\nEnd of trial %d, score = %d out of %d .",trial+1,score[trial],PAIRS);
printf("\nGet ready for next trial.");
if (trial>0 && score[trial]==PAIRS && score[trial-1]==PAIRS) {
    trial++;
    break;
    }
    else
        trial++;
    }                    /* this is while loop closing bracket */

    clrscrn( );
    printf("\nCongratulation! learning is completed after %d trials",
        trial);
    /* print results to file */
    fp = fopen(idcode, "w");
    for (i=0; i<(trial); i++) {
        fprintf(fp, "\n\nTrial #%d score = %d", i+1, score[i]);
        fprintf(fp, "\nSequence: ");
    for (j=0; j<PAIRS; j++)
        fprintf(fp, "%3d,", seqsto[i*PAIRS+j]);
    fprintf(fp, "\nHit&miss: ");
    for (j=0; j<PAIRS; j++)
        fprintf(fp, "%3d,", hitsto[i*PAIRS+j]);
```

```
fprintf(fp, "\nReactime: ");
for (j=0; j<PAIRS; j++)
    fprintf(fp, "%3d,", timsto[i*PAIRS+j]);
    }
    fclose(fp);
}

/ * function clear screen using bios call. #include <dos.h> needed */
clrscrn( )
{
    union REGS regs;    /* union REGS type defined in dos.h */
    regs.h.ah = 6;      /* scroll up function */
    regs.h.al = 0;      /* blank window */
    regs.h.ch = 0;      /* top row */
    regs.h.cl = 0;      /* left col */
    regs.h.dh = 24;     /* bot row */
    regs.h.dl = 79;     /* right col */
    regs.h.bh = 7;      /* blank line attribute */
    int86(0x10, &regs, &regs);
    regs.h.ah = 2;      /* cursor pos function */
    regs.h.dh = 0;      /* row */
    regs.h.dl = 0;      /* col */
    regs.h.bh = 0;      /* page */
    int86(0x10, &regs, &regs);
} /* end function */
int setcur(row, col, page)
int row, col, page;
{
```

```
    union REGS regs;
    regs.h.ah = 2;          /* interrupt 16 function #2 */
    regs.h.dh = row;
    regs.h.dl = col;
    regs.h.bh = page;
    int86(0x10, &regs, &regs);
}
/ sequencer funtion */
int seqer(t, seqsto)
int t, *seqsto;     /* t is trial #, seqsto stores sequence #'s */
{
    int tmpvec[PAIRS];              /* to receive 10 random #'s */
    int tmpval;                     /* temp storage for swap */
    register int i, j;              /* fast counters */
    for (i=0; i<PAIRS; i++)
        seqvec[i] = i;
    for (i=0; i<PAIRS; i++)
        tmpvec[i] = rand( );        /* do srand( ) in main( ) */
/* Now bubble sort tmpvec[] tag swap to seqvec[] */
for (i=0; i<(PAIRS-1); i++) {
    for (j=i+1; j<PAIRS; j++) {
        if (tmpvec[i] < tmpvec[j])
            ;                       /* do nothing */
        else {
            tmpval = tmpvec[j];     /* start tmpvec swap */
            tmpvec[j] = tmpvec[i];
            tmpvec[i] = tmpval;
            tmpval = seqvec[j];     /* start seqvec swap */
            seqvec[j] = seqvec[i];
```

```
            seqvec[i] = tmpval;
        }
    }
}
/* Sto content of seqvec[ ] in seqsto[ ], t is trial # */
for (j=0; j<PAIRS; j++)
    seqsto[t*PAIRS+j] = seqvec[j];
}

/**** 4. get time in hundredths of a second *****/

it gettime(ptr)
lng *ptr;
{
    struct timeb xtime;
    ftime(&xtime);
    *ptr = (long)xtime.millitm/10+xtime.time*100;
}

int clrkh()                 /* function clear kbhit. kbhit funtion has a long
                               memory, waiting in buffer. move them out */
{
    char dummy;
    while (kbhit( ))
        dummy = getch( );
}
```

โปรแกรมนี้เขียนโดยใช้ภาษาซี (C Language) ซึ่งเป็นภาษาคอมพิวเตอร์ที่ได้รับความนิยมกันมาก เมื่อไม่นานมานี้ (สำหรับผู้อ่านที่สนใจภาษาซี ผู้เขียนขอแนะนำหนังสือของ Lafore, 1987) ตัวโปรแกรม (source program) ที่นำเสนอในบทความนี้ได้รับการทดลองคอมไพล์ (compile) และลิ้งค์ (link) โดยไมโครซอฟท์ซีรุ่นที่ 4 (Microsoft C Version 4) สำหรับเอมเอสดอส (MSDOS) ได้ผลดี เพราะฉะนั้นถ้าท่านผู้อ่านมีไมโครซอฟท์ซีสำหรับเอมเอสดอสตั้งแต่รุ่นที่ 4 ขึ้นไปก็ไม่น่าจะมีปัญหาในการคอมไพล์และลิ้งค์ ถ้าหากผู้อ่านต้องการโปรแกรมสำเร็จรูปที่ใช้ได้เลย (executable program) กรุณาติดต่อ ดร. ชาญชัย อาชวเมธี (ผู้ร่วมเขียนบทความนี้) ต่อไปนี้เป็นการอธิบายตัวโปรแกรมโดยย่อ

คำอธิบายตัวคอมพิวเตอร์โปรแกรมโดยย่อ

ตอนที่ 1 (โปรดสังเกตบรรทัดสี่บรรทัดใต้ /***** 1. define parameters *****/ ในตัวโปรแกรม) ผู้วิจัยกำหนดเอาเองว่า

1.1 จะศึกษาคู่สัมพันธ์กี่คู่ ในตัวอย่างที่ให้ไว้ แปดคู่ (PAIRS = 8)

1.2 สิ่งเร้าจะปรากฏก่อนการเฉลยคำตอบ (การตอบสนอง) นานกี่เซ็นติวินาที (หนึ่งเซ็นติวินาที = เศษหนึ่งส่วนร้อยวินาที) ตัวอย่างในโปรแกรมกำหนดสามวินาที (SINTV = 300)

1.3 สิ่งเร้าพร้อมคำเฉลย (การตอบสนอง) จะปรากฏนานกี่เซ็นติวินาที ตัวอย่างในโปรแกรมกำหนดสองวินาที (SRINTV = 200)

1.4 ต้องการให้ผู้เรียนใช้เวลามากที่สุดกี่รอบ ตัวอย่างในโปรแกรมกำหนดไว้ 30 รอบ (MAXTRIALS = 30)

ตอนที่ 2 (โปรดสังเกตส่วนที่อยู่ใต้ /***** 2. stimulus and correct response stting *****/ ในตัวโปรแกรม) ส่วนนี้เป็นส่วนที่ผู้วิจัยกำหนดเอาเองว่าจะให้ผู้เรียนเรียนคู่สัมพันธ์อะไร ตัวอย่างในโปรแกรมเป็นการเรียนเลขฐานแปด (octal numbers) ในรูปฐานสอง (binary)

ตอนที่ 3 (โปรดสังเกตส่วนที่อยู่ใต้ /***** 3. start with learner's ID code *****/ ในตัวโปรแกรม) คอมพิวเตอร์เริ่มโดยให้ผู้เรียนพิมพ์หมายเลขประจำตัวลงในคอมพิวเตอร์แล้วเริ่มเสนอคู่สัมพันธ์ให้ผู้เรียนเรียนทีละรอบจนถูกหมดสองรอบติดต่อกัน หรือถ้าผู้เรียนไม่สามารถตอบได้ถูกหมดสองรอบติดต่อกันภายใน 30 รอบ คอมพิวเตอร์ก็จะหยุดเมื่อครบ 30 รอบ ในแต่ละรอบคู่สัมพันธ์ทั้งแปดคู่จะได้รับการเรียงสุ่ม (random sequence) ใหม่โดยฟังก์ชั่น seqer

(โปรดสังเกตบรรทัดใต้ /***** 3.1 call random sequencer function *****/ ในตัวโปรแกรม) ในแต่ละคู่ สิ่งเร้าจะปรากฏตัวขึ้นก่อนเป็นเวลาสามวินาทีเพื่อให้ผู้เรียนกดคำตอบ ถ้าผู้เรียนกดคำตอบถูก คอมพิวเตอร์ก็จะแจ้งว่าตอบถูก แล้วบันทึกว่าถูก แล้วบันทึกเวลาในการกดด้วย (เป็นเซ็นติวินาที) ถ้าผู้เรียนกดคำตอบผิด คอมพิวเตอร์จะไม่แจ้งว่าตอบถูก แต่จะบันทึกในสมองอิเล็กทรอนิกส์ว่าผิดและบันทึกเวลาในการกดด้วย ถ้าผู้เรียนไม่กดคำตอบภายในสามวินาทีคอมพิวเตอร์ถือว่าตอบผิดและบันทึกเวลาการตอบเป็นสามวินาที หลังจากสิ่งเร้าปรากฏตัวสามวินาที (ไม่ว่าผู้เรียนจะตอบหรือไม่) สิ่งเร้าพร้อมคำตอบที่ถูกจะปรากฏตัวเป็นเวลาสองวินาที หลังจากคอมพิวเตอร์เสนอคู่สัมพันธ์ครบแปดคู่ในแต่ละรอบก็จะแจ้งให้ผู้เรียนทราบว่าในจำนวนแปดคู่ตอบถูกกี่คู่

ตอนที่ 4 (โปรดสังเกตส่วนที่อยู่ใต้ /***** 4. get time in hundredths of a second ****/ ในตัวโปรแกรม) ฟังก์ชั่น time.millitm ของไมโครซอฟท์ซี (Microsoft C) แม้จะให้ผลเป็นหน่วย "มิลลิวินาที" (เศษหนึ่งส่วนพันของวินาที) แต่จะไม่แม่นตรงในระดับมิลลิวินาที ทั้งนี้เป็นเพราะว่านาฬิกาคอมพิวเตอร์ภายในติ๊ก (tick) ทุกๆ 55 มิลลิวินาที แทนที่จะเป็นทุกหนึ่งมิลลิวินาที (Lafore, 1997, p.510) ตัวอย่างเช่นถ้าเวลาที่เก็บไว้ในคอมพิวเตอร์ขณะนี้เป็น 220 มิลลิวินาที เวลาที่ปรับใหม่ ครั้งต่อไปจะเป็น 275 มิลลิวินาทีโดยไม่ผ่าน 221, 222, ... ดังนั้น ผู้เขียนบทความเรื่องนี้จึงเขียนโปรแกรมให้บันทึกความเร็วของการกดคำตอบเป็นเซ็นติวินาทีแทนที่จะเป็นมิลลิวินาที (โปรดสงเกตบรรทัด:

```
*ptr = (long)xtime.millitm/10 + xtime.time*100;
```

ในตัวโปรแกรม) แม้กระนั้น เวลาอ่านผลการวิจัยจากโปรแกรมนี้ผู้วิจัยควรปัดเศษขึ้นไปอีกหนึ่งหน่วย เช่น อ่าน 242 เซ็นติวินาที เป็น 2.4 วินาที แทนที่จะเป็น 2.42 วินาที

ตัวอย่างผลการเรียนของผู้เรียนที่คอมพิวเตอร์พิมพ์ออกมา

หลังจากผู้เรียนเรียนสำเร็จ (ถูกหมดสองรอบติดต่อกันหรือเรียนไปครบ 30 รอบ) คอมพิวเตอร์จะพิมพ์ข้อมูลการเรียนไว้ในแฟ้มข้อมูลชื่อเดียวกับหมายเลขประจำตัวที่ผู้เรียนพิมพ์ลงไปในคอมพิวเตอร์แฟ้มข้อมูลนี้เก็บไว้ในดิสค์ซึ่งผู้วิจัยสามารถสั่งให้คอมพิวเตอร์พิมพ์ลงกระดาษในภายหลังได้ ต่อไปนี้เป็นตัวอย่างข้อมูลการเรียนคู่สัมพันธ์ที่คอมพิวเตอร์พิมพ์เก็บไว้ในดิสค์

Trial #1 score = 0

Sequence: 4, 7, 3, 0, 5, 6, 2, 1,

Hit&Miss: 0, 0, 0, 0, 0, 0, 0, 0,

Reactime: 300,300,300,300,300,300,300,300,

Trial #2 score = 6

Sequence: 6, 3, 7, 0, 1, 4, 2, 5,

Hit&Miss: 1, 1, 1, 1, 0, 1, 0, 1,

Reactime: 242,192,231,148,148,154,137, 93,

Trial #3 score = 5

Sequence: 0, 5, 3, 7, 2, 6, 4, 1,

Hit&Miss: 1, 0, 0, 1, 1, 0, 1, 1,

Reactime: 121,176,187,241,181,126, 82,115,

.

.

.

ตัวอย่างข้างต้นแสดงข้อมูลที่คอมพิวเตอร์บันทึกไว้ในสามรอบแรก โปรดสังเกตว่าในรอบที่หนึ่ง (Trial#1) บรรทัดที่หนึ่งเนื่องจากผู้เรียนเพียงแต่สังเกตว่าสิ่งเร้าใดคู่กับคำตอบสนองใด คะแนน (score) จึงเป็นศูนย์ ในบรรทัดที่สอง (Sequence) คอมพิวเตอร์บันทึกลำดับการเสนอคู่สัมพันธ์ทั้งแปดคู่ โปรดสังเกตว่าเนื่องจากคอมพิวเตอร์สุ่มลำดับ (randomize) ทุกรอบ ลำดับการเสนอจึงไม่เรียงเป็นระเบียบ 1, 2, 3, ... ในบรรทัดที่สาม (Hit&miss) ถ้าผู้เรียนไม่ตอบหรือตอบผิดในคู่ใดคะแนนสำหรับคู่นั้นจะเป็นศูนย์ ถ้าตอบถูกก็จะเป็นหนึ่ง ในบรรทัดที่สี่(Reactime ซึ่งย่อมาจาก Reaction time)คอมพิวเตอร์บันทึกเวลาการตอบสนองนับตั้งแต่สิ่งเร้าปรากฏตัว คอมพิวเตอร์บันทึกเป็นเศษหนึ่งส่วนร้อยของวินาที (เซ็นติวินาที) ถ้าผู้เรียนไม่ตอบคอมพิวเตอร์จะบันทึกเป็น 300 เซ็นติวินาที(เท่ากับ 3 วินาที)

โครงการวิจัย

คอมพิวเตอร์โปรแกรมที่เสนอในบทความเรื่องนี้สามารถรับการดัดแปลงเพื่อใช้ในโครงการวิจัยต่าง ๆ ได้ ตัวอย่างที่เสนอนี้เป็นการเรียนเลขฐานแปด (octal numbers) ในรูป

binary ท่านผู้อ่านอาจจะดัดแปลงไปศึกษาเลขฐานสิบหก (hexadecimal numbers) ในรูป binary หรือการเรียนคำภาษาต่างประเทศ หรือการเรียนคู่สัมพันธ์อื่นๆ คำตอบสนองใดที่ไม่ปรากฏในคีย์บอร์ดก็พิมพ์ใส่กระดาษติดไปบนคีย์บอร์ดใช้ชั่วคราวได้ ในการเรียนคู่สัมพันธ์แต่ละอย่าง เช่น เลขฐานแปด ผู้วิจัยสามารถทดลองวิธีสอนหรือวิธีแนะแบบต่าง ๆ ก่อนเริ่มโปรแกรมคอม-พิวเตอร์ที่เสนอในบทความนี้ เพื่อดูว่าวิธีไหนมีประสิทธิภาพสูงสุด นอกจากนี้ ผู้วิจัยอาจจะลองดูกับผู้เรียนวัยต่าง ๆ ได้ จากบันทึกผลการเรียนที่โปรแกรมคอมพิวเตอร์พิมพ์ออกมาผู้วิจัยสามารถวิเคราะห์ได้ด้วยว่าคู่สัมพันธ์ใดเรียนง่ายที่สุดและคู่สัมพันธ์ใดเรียนยากที่สุด

โครงการวิจัยบางอย่างสามารถใช้โปรแกรมคอมพิวเตอร์ที่เสนอในบทความนี้ได้เลยโดยไม่ต้องดัดแปลงแต่อย่างใด เช่น วิจัยการสอนเลขฐานแปดแบบต่าง ๆ โครงการบางโครงการอาจต้องดัดแปลง โปรแกรมคอมพิวเตอร์ที่เสนอนี้เล็กน้อยแต่ทำได้โดยง่าย เช่น โครงการวิจัยการเรียนเลขฐานสิบหกแทนที่จะเป็นฐานแปด โครงการวิจัยระยะเวลาการเสนอสิ่งเร้าก่อนเฉลยคำตอบ(เช่น 6 วินาที แทนที่จะเป็น 3 วินาที) โครงการบางโครงการก็อาจจะต้องดัดแปลงโปรแกรมคอมพิวเตอร์ที่เสนอนี้มากหน่อย เช่น เปรียบเทียบการเรียนคู่สัมพันธ์แบบสุ่มคู่ทุกรอบ (random sequence every trial) กับการเรียนคู่สัมพันธ์แบบที่เสนอคู่ที่ตอบผิดก่อนคู่ที่ตอบถูกในรอบที่แล้ว ท่านผู้อ่านที่สนใจจะดัดแปลงโปรแกรมคอมพิวเตอร์ที่เสนอในบทความนี้เพื่อใช้ในงานวิจัยแต่ไม่สามารถดัดแปลงด้วยตนเองได้อาจติดต่อนักโปรแกรมในมหาวิทยาลัยที่อยู่ใกล้เคียง หรือ ดร. ชาญชัย ซึ่งเป็นผู้เขียนผู้หนึ่งของบทความฉบับนี้

บรรณานุกรม

Bigge, Morris L. (1982). **Learning theories for teachers** (4th ed.). New York: Harper and Row.

Haber, R. N., & Fried, A. H. (1975). **An introduction to psychology** New York: Holt, Rinehart and Winston.

Hilgard, E. R., Atkinson, R. T., & Atkinson, R. L. (1975). **Introduction to psychology** (6th ed.). New York: Harcourt Brace Jovanovich.

Jung, John (1968). **Verbal learning**. New York: Holt, Rinehart and Winston.

Koppenaal, R. J. (1963). Time changes in the strengths of A-B, A-C lists; spontaneous recovery? **Journal of Verbal Learning and Verbal Behavior**, 2, 310-319.

Lafore, Robert (1987). **Microsoft C programming for the IBM**. Indianapolis, IN: Howard W. Sams & Company.

Lefrancois, Guy R. (1994). **Psychology for teaching** (8th ed.). Belmont, CA: Wadsworth Publishing Company.