人工知能 ii　A*アルゴリズム 課題

5CS 29 坂田大地

1.　アルゴリズムの説明

A*アルゴリズムは最良優先探索法の一つで，コストの総和が最小となる経路を探索するアルゴリズムである．

以下に A*アルゴリズムのフローチャートを示す．

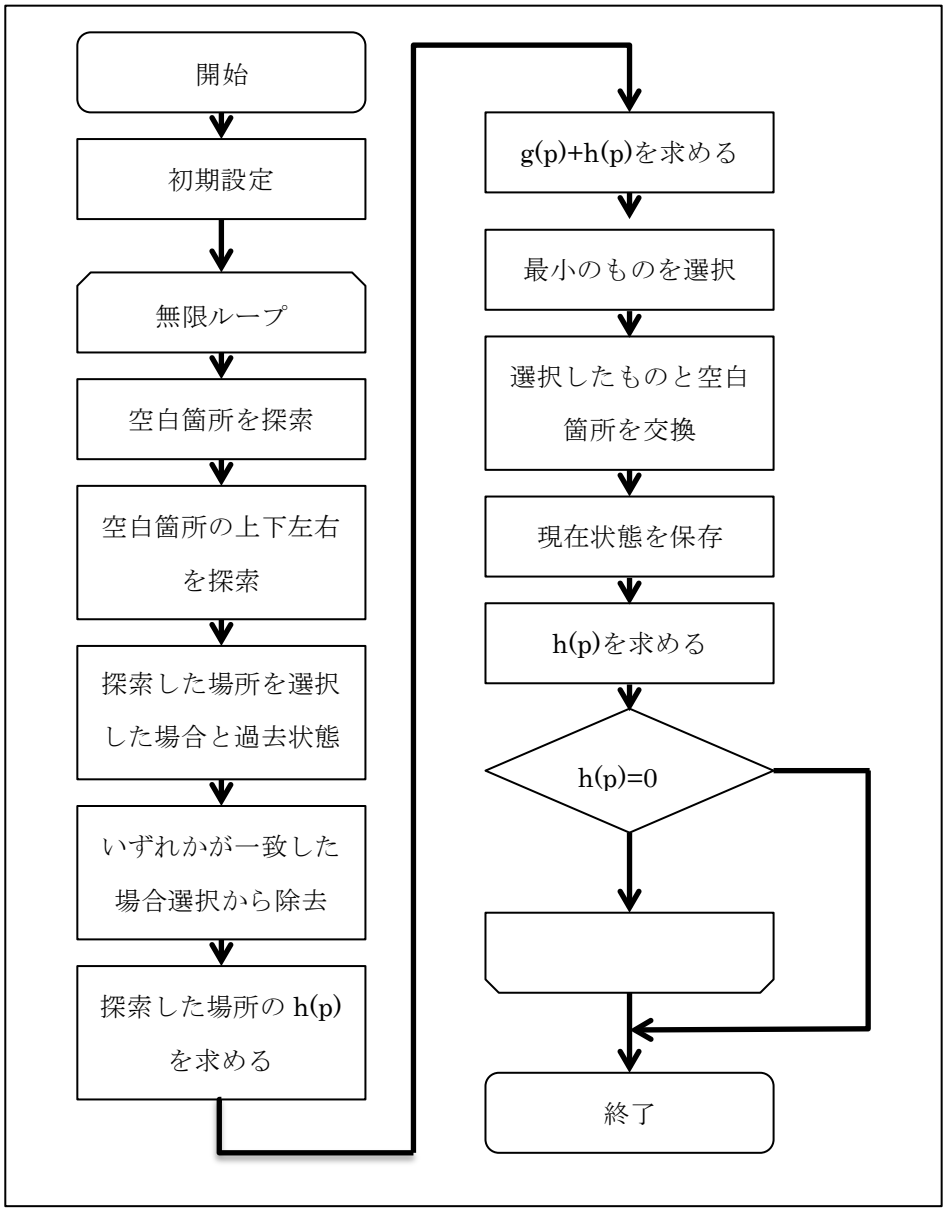

図 1 フローチャート

2.　ソースコード

ソースコードを以下に記す.

ただし，掲載するソースコードは，作成途中のものである.

```
#include <stdio.h>
#include <stdlib.h>

void swap(int *x, int *y);
int h(int now_p[3][3]);
int past_comparison(int nowcount, int x[3][3], int y[255][3][3]);
int pn_zahyo_x(int num);
int pn_zahyo_y(int num);

int main(){
    int count = 1;
    int pn[3][3] = {{1,2,3},
                    {4,5,6},
                    {7,8,0}};
    int  p[3][3] = {{8,1,5},
                    {2,0,4},
                    {6,3,7}};
    int past[500][3][3]  = {{0},{8,1,5},
                            {2,0,4},
                            {6,3,7}};

    int fp[4];
    int hp[4];
    int gp = 0;
    int pn_x, pn_y, nowzero_x, nowzero_y, min, choice, befor_x, befor_y;
    FILE *filep;
    filep = fopen("result.txt", "w");
```

```
count = 1;
//無限ループ
while(1){
//printf("¥n%d回目**********¥n", count);

pn_x = pn_y = 0;

//現在状態表示
for(int cnt_x = 0; cnt_x < 3; cnt_x++){
	for(int cnt_y = 0; cnt_y < 3; cnt_y++){
		printf("%d ", p[cnt_x][cnt_y]);
		fprintf(filep, "%d ", p[cnt_x][cnt_y]);
	}
	printf("¥n");
	fprintf(filep, "¥n");
}
printf("¥n");
fprintf(filep, "¥n");

//STEP1
//次の状態の探索
int nextmove_cnt = 0;
int nextmove[4];
int nextmove_zahyo[4][2];

//0探査⇨0周囲を探索
for(int cnt_x = 0; cnt_x < 3; cnt_x++){
	for(int cnt_y = 0; cnt_y < 3; cnt_y++){
		if(p[cnt_x][cnt_y] == 0){
			//ゼロ保持
			nowzero_x = cnt_x;
			nowzero_y = cnt_y;
```

```
//初回のみ
if(count == 1){
    //ゼロ左
    if(0 <= cnt_y - 1 && cnt_y - 1 < 3){
        nextmove[nextmove_cnt] = p[cnt_x][cnt_y - 1];
        nextmove_zahyo[nextmove_cnt][0] = cnt_x;
        nextmove_zahyo[nextmove_cnt][1] = cnt_y - 1;
        nextmove_cnt++;
    }
    //ゼロ右
    if(0 <= cnt_y + 1 && cnt_y + 1 < 3){
        nextmove[nextmove_cnt] = p[cnt_x][cnt_y + 1];
        nextmove_zahyo[nextmove_cnt][0] = cnt_x;
        nextmove_zahyo[nextmove_cnt][1] = cnt_y + 1;
        nextmove_cnt++;
    }
    //ゼロ上
    if(0 <= cnt_x - 1 && cnt_x - 1 < 3){
        nextmove[nextmove_cnt] = p[cnt_x - 1][cnt_y];
        nextmove_zahyo[nextmove_cnt][0] = cnt_x - 1;
        nextmove_zahyo[nextmove_cnt][1] = cnt_y;
        nextmove_cnt++;
    }
    //ゼロ下
    if(0 <= cnt_x + 1 && cnt_x + 1 < 3){
        nextmove[nextmove_cnt] = p[cnt_x + 1][cnt_y];
        nextmove_zahyo[nextmove_cnt][0] = cnt_x + 1;
        nextmove_zahyo[nextmove_cnt][1] = cnt_y;
        nextmove_cnt++;
    }
    break;
}
```

```
                    //2回目以降
                    else{
                        if(0 <= cnt_y - 1 && cnt_y - 1 < 3 && p[cnt_x][cnt_y -
1] != p[befor_x][befor_y]){
                            swap(&p[cnt_x][cnt_y], &p[cnt_x][cnt_y - 1]);
                            if(0 == past_comparison(count, p, past)){
                                nextmove[nextmove_cnt] = p[cnt_x][cnt_y - 1];
                                nextmove_zahyo[nextmove_cnt][0] = cnt_x;
                                nextmove_zahyo[nextmove_cnt][1] = cnt_y - 1;
                                nextmove_cnt++;
                            }
                            swap(&p[cnt_x][cnt_y], &p[cnt_x][cnt_y - 1]);
                        }
                        //ゼロ右
                        if(0 <= cnt_y + 1 && cnt_y + 1 < 3 && p[cnt_x][cnt_y +
1] != p[befor_x][befor_y]){
                            swap(&p[cnt_x][cnt_y], &p[cnt_x][cnt_y + 1]);
                            if(0 == past_comparison(count, p, past)){
                                nextmove[nextmove_cnt] = p[cnt_x][cnt_y + 1];
                                nextmove_zahyo[nextmove_cnt][0] = cnt_x;
                                nextmove_zahyo[nextmove_cnt][1] = cnt_y + 1;
                                nextmove_cnt++;
                            }
                            swap(&p[cnt_x][cnt_y], &p[cnt_x][cnt_y + 1]);
                        }
                        //ゼロ上
                        if(0 <= cnt_x - 1 && cnt_x - 1 < 3 && p[cnt_x -
1][cnt_y] != p[befor_x][befor_y]){
                            swap(&p[cnt_x][cnt_y], &p[cnt_x - 1][cnt_y]);
                            if(0 == past_comparison(count, p, past)){
                                nextmove[nextmove_cnt] = p[cnt_x - 1][cnt_y];
                                nextmove_zahyo[nextmove_cnt][0] = cnt_x - 1;
```

```
                                nextmove_zahyo[nextmove_cnt][1] = cnt_y;
                                nextmove_cnt++;
                            }
                            swap(&p[cnt_x][cnt_y], &p[cnt_x - 1][cnt_y]);
                        }
                        //ゼロ下
                        if(0 <= cnt_x + 1 && cnt_x + 1 < 3 && p[cnt_x +
1][cnt_y] != p[befor_x][befor_y]){
                            swap(&p[cnt_x][cnt_y], &p[cnt_x + 1][cnt_y]);
                            if(0 == past_comparison(count, p, past)){
                                nextmove[nextmove_cnt] = p[cnt_x + 1][cnt_y];
                                nextmove_zahyo[nextmove_cnt][0] = cnt_x + 1;
                                nextmove_zahyo[nextmove_cnt][1] = cnt_y;
                                nextmove_cnt++;
                            }
                            swap(&p[cnt_x][cnt_y], &p[cnt_x + 1][cnt_y]);
                        }
                        break;
                    }
                }
            }
        }

        //printf("Now zero = (%d, %d)¥n", nowzero_x, nowzero_y);
        for(int cnt = 0; cnt < nextmove_cnt; cnt++)
            //printf("nextmove = %d (%d, %d)¥n", nextmove[cnt],
nextmove_zahyo[cnt][0], nextmove_zahyo[cnt][1]);



        //STEP2
        //Hpを求める
```

```
	for(int cnt = 0; cnt < nextmove_cnt; cnt ++){
		//printf("swap %d¥n",
p[nextmove_zahyo[cnt][0]][nextmove_zahyo[cnt][1]]);
		//要素入れ替え
		swap(&p[nowzero_x][nowzero_y],
&p[nextmove_zahyo[cnt][0]][nextmove_zahyo[cnt][1]]);
		//hp求める
		hp[cnt] = h(p);
		//元に戻す
		swap(&p[nowzero_x][nowzero_y],
&p[nextmove_zahyo[cnt][0]][nextmove_zahyo[cnt][1]]);
		//printf("¥nhp = %d¥n¥n", hp[cnt]);
	}


	//STEP3
	//Gp + Hpを求めて経路を選択

	for(int cnt = 0; cnt < nextmove_cnt; cnt ++){
		fp[cnt] = gp + hp[cnt];
		if(cnt == 0){
			min = fp[cnt];
			choice = cnt;
		}
		else{
			if(min > fp[cnt]){
				min = fp[cnt];
				choice = cnt;
			}
		}
	}
```

```
        printf("Choose the : %d¥nCost : %d¥n",
p[nextmove_zahyo[choice][0]][nextmove_zahyo[choice][1]], hp[choice]);
        fprintf(filep, "f(%d) = %d¥n", count, fp[choice]);
        swap(&p[nowzero_x][nowzero_y],
&p[nextmove_zahyo[choice][0]][nextmove_zahyo[choice][1]]);
                befor_x = nowzero_x;
                befor_y = nowzero_y;
                for(int cnt_x = 0; cnt_x < 3; cnt_x++){
                        for(int cnt_y = 0; cnt_y < 3; cnt_y++){
                                past[count][cnt_x][cnt_y] = p[cnt_x][cnt_y];
                        }
                }
                for(int cnt_x = 0; cnt_x < 3; cnt_x++){
                        for(int cnt_y = 0; cnt_y < 3; cnt_y++){
                                printf("%d ", past[count][cnt_x][cnt_y]);
                        }
                        printf("¥n");
                }
                printf("¥n");

                gp = fp[choice];
                //printf("Gp : %d¥n", gp);
                if(h(p) == 0){
                        break;
                }
                else{
                        count++;
                        if(count > 500)                 //一応
                                break;
                }
```

```
	}



	printf("¥n==end==¥n");
	printf("count¥t¥t: %d¥nTotal Cost¥t: %d¥n", count, gp);
	for(int cnt_x = 0; cnt_x < 3; cnt_x++){
		for(int cnt_y = 0; cnt_y < 3; cnt_y++){
			printf("%d ",p[cnt_x][cnt_y]);
		}
		printf("¥n");
	}
	printf("¥n");
	fclose(filep);
	 return 0;
}

int h(int now_p[3][3]){
	int pn_x, pn_y, hp = 0;
	int M[3][3];
	//マンハッタン距離計算しつつhp計算
	//printf("Manhattan¥n");
	for(int cnt_x = 0; cnt_x < 3; cnt_x++){
		for(int cnt_y = 0; cnt_y < 3; cnt_y++){
			pn_x = pn_zahyo_x(now_p[cnt_x][cnt_y]);
			pn_y = pn_zahyo_y(now_p[cnt_x][cnt_y]);
			M[cnt_x][cnt_y] = abs(cnt_x - pn_x) + abs(cnt_y - pn_y);
			//printf("%d ", M[cnt_x][cnt_y]);
			hp += M[cnt_x][cnt_y];
		}
		//printf("¥n");
	}
```

```
	return hp;
}

void swap(int *x, int *y){
	int tmp;
	tmp = *x;
	*x = *y;
	*y = tmp;
}

int past_comparison(int nowcount, int x[3][3], int y[255][3][3]){
	int matchnum;
	for(int cnt = 0; cnt < nowcount; cnt++){
		matchnum = 0;
		for(int cnt_x = 0; cnt_x < 3; cnt_x++){
			for(int cnt_y = 0; cnt_y < 3; cnt_y++){
				if(x[cnt_x][cnt_y] == y[cnt][cnt_x][cnt_y]){
					matchnum ++;
				}
			}
		}
		if(matchnum == 9){
			printf("Count %d is match¥n", cnt);
			return 1;
			break;
		}
	}
	return 0;
}

int pn_zahyo_x(int num){
	switch(num){
```

```
            case 1:
            case 2:
            case 3:
                return 0;
                break;
            case 4:
            case 5:
            case 6:
                return 1;
                break;
            default:
                return 2;
                break;
        }
}

int pn_zahyo_y(int num){
        switch(num){
            case 1:
            case 4:
            case 7:
                return 0;
                break;
            case 2:
            case 5:
            case 8:
                return 1;
                break;
            default:
                return 2;
                break;
        }
```

```
}
```

図 2 ソースコード

3. プログラムの実行結果

プログラムのコンパイル実行環境を以下に記す.

表 1 実行環境

PC	MacBook Pro(Retina, 13-inch, Mid 2014)
OS	macOS High Sierra(ver 10.13)
CPU	2.6 GHz Intel Core i5
Ram	8 GB 1600 MHz DDR3
コンパイラ	GCC 4.2.1

プログラムの実行結果を以下に記す.

ただし，掲載する実行結果はファイルに出力された内容である．また，プログラムは作成途中のため，施工中に無限ループになる．そのため，実行回数は 20 回としている．

```
8 1 5
2 0 4
6 3 7

f(1) = 18
8 1 5
2 4 0
6 3 7

f(2) = 36
```

8 1 0

2 4 5

6 3 7

f(3) = 56

8 0 1

2 4 5

6 3 7

f(4) = 76

0 8 1

2 4 5

6 3 7

f(5) = 94

2 8 1

0 4 5

6 3 7

f(6) = 110

2 8 1

4 0 5

6 3 7

f(7) = 124

2 8 1

4 5 0

6 3 7

f(8) = 138

2 8 1

4 5 7

6 3 0

f(9) = 152

2 8 1

4 5 7

6 0 3

f(10) = 166

2 8 1

4 5 7

0 6 3

f(11) = 182

2 8 1

0 5 7

4 6 3

f(12) = 198

2 8 1

5 0 7

4 6 3

f(13) = 212

2 8 1

5 7 0

4 6 3

f(14) = 224

2 8 1

5 7 3

4 6 0

```
f(15) = 236
2 8 1
5 7 3
4 0 6

f(16) = 248
2 8 1
5 0 3
4 7 6

f(17) = 260
2 8 1
0 5 3
4 7 6

f(18) = 270
2 8 1
4 5 3
0 7 6

f(19) = 278
2 8 1
4 5 3
7 0 6

f(20) = 286
```

図 3 実行結果

発見的関数の値の推移のグラフを以下に記す.

ただし，プログラムが作成途中のため，施工中に無限ループになる．そのため，実行回数は 20 回としている.

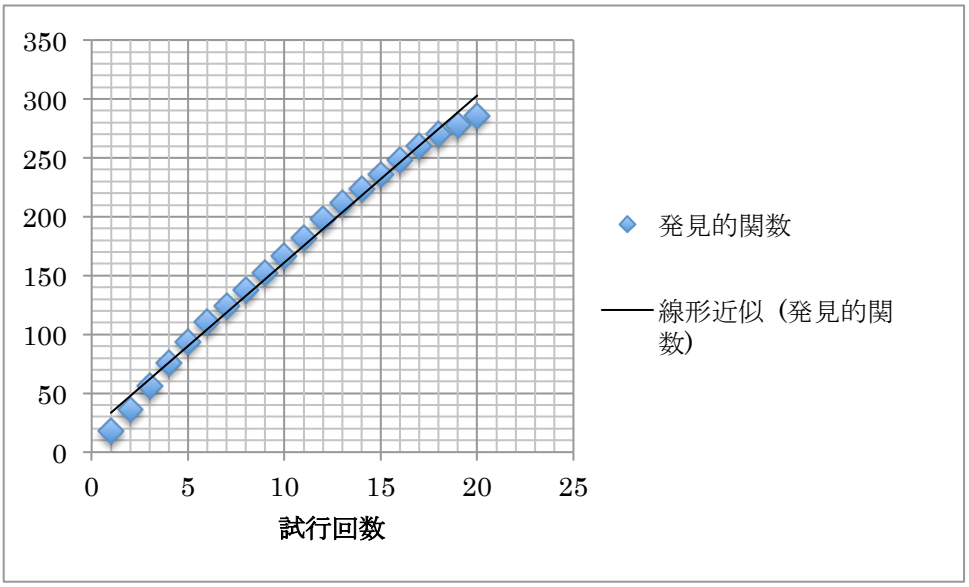

図 4 発見的関数の値の推移