using System;

using System.Collections.Generic;

using System.Text;

namespace AntSystem

{

public class Ant

{

/\*\*//// <summary>

/// 对信息量的重视程度

/// </summary>

private int alpha;

/\*\*//// <summary>

/// 启发式信息的受重视程度

/// </summary>

private int beta;

/\*\*//// <summary>

/// 信息素的挥发速度

/// </summary>

private double lo;

/\*\*//// <summary>

/// 城市距离矩阵

/// </summary>

private double[,,] City\_Cost;

/\*\*//// <summary>

/// 信息素矩阵

/// </summary>

private double[,,] Message;

/\*\*//// <summary>

/// opneList用于存放下一步可行城市

/// </summary>

private Queue<int> openList=new Queue<int> ();

/\*\*//// <summary>

/// closedList用于存放已经访问过的城市

/// </summary>

private Queue<int> closedList=new Queue<int> ();

/\*\*//// <summary>

/// 储存较好的路径

/// </summary>

private Queue <int> BestList=new Queue<int> ();

/\*\*//// <summary>

/// 储存较好的路径使用的运输方式

/// </summary>

private Queue<int> TraficList = new Queue<int>();

private int N;

private int Pro\_time = 0;

/\*\*//////////////////////////////////////////////////////////

/// <summary>

/// 构造函数：形成城市距离和信息素矩阵

/// </summary>

/// <param name="City\_Cost">城市距离矩阵</param>

/// <param name="Lo"> 信息素的挥发速度</param>

public Ant(double[ , , ] City\_Cost,double Lo,int Alpha,int Beta,int traficNum)

{

alpha = Alpha;

beta = Beta;

lo=Lo;

int temp = Convert.ToInt32(Math.Sqrt(city\_Cost.Length / TraficNum));

City\_Cost=new double [temp,temp,traficNum];

Message=new double [temp,temp,traficNum];

for (int i = 0; i < temp; i++)

{

for (int j = 0; j < temp; j++)

{

for (int k = 0; k < traficNum; k++)

{

City\_Cost[i, j, k] = city\_Cost[i, j, k];

}

}

}

//初始化信息素矩阵

for (int i = 0; i < temp; i++)

{

for (int j = 0; j < temp; j++)

{

if (i != j)

{

for (int k = 0; k < traficNum; k++)

{

Message[i, j, k] = (double)1 / (temp \* temp - temp);

}

}

}

}

Order = getorder();

}

/\*\*/////////////////////////////////////////////////////////////

/// <summary>

/// 改变信息素矩阵，closed\_list为较好的路径

/// </summary>

/// <param name="closed\_list"></param>

private void Change\_Message(Queue<int> closed\_list,int n)

{

lock (this)

{

int[] temp\_Array = new int[closed\_list.Count];

temp\_Array = closed\_list.ToArray();

for (int i = 0; i < closed\_list.Count - 1; i++)

{

for (int k = 0; k < TraficNum; k++)

{

Message[temp\_Array[i], temp\_Array[i + 1], k] = Message[temp\_Array[i], temp\_Array[i + 1], k] + lo / ((1 - lo) \* Convert.ToInt32(Get\_Weight(closed\_list) + 1));

}

}

for (int k = 0; k < TraficNum; k++)

{

Message[temp\_Array[temp\_Array.Length - 1], temp\_Array[0], k] = Message[temp\_Array[temp\_Array.Length - 1], temp\_Array[0], k] + lo / ((1 - lo) \* Convert.ToInt32(Get\_Weight(closed\_list)));

}

for (int i = 0; i < closed\_list.Count; i++)

{

for (int j = 0; j < closed\_list.Count; j++)

{

for (int k = 0; k < TraficNum; k++)

{

Message[i, j, k] = (1 - lo) \* Message[i, j, k];

}

}

}

}

}

/\*\*////////////////////////////////////////////////////////////////

/// <summary>

/// 输入一个链表，计算出其对应的总路径

/// </summary>

/// <param name="closed\_list"></param>

/// <returns></returns>

public double Get\_Weight(Queue <int> closed\_list)

{

lock (this)

{

double sum = 0;

int[] temp\_Array = new int[closed\_list.Count];

temp\_Array = closed\_list.ToArray();

for (int i = 0; i < Convert.ToInt32(temp\_Array.Length) - 1; i++)

{

for (int k = 0; k < TraficNum; k++)

{

sum = sum + Total\_Cost(City\_Cost);

}

}

for (int k = 0; k < TraficNum; k++)

{

sum = sum + Total\_Cost(City\_Cost);

}

return sum;

}

}

/\*\*///////////////////////////////////////////////////////////////

/// <summary>

/// 计算总的运输成本，返回到Get\_Weight中。

/// </summary>

/// <param name="i"></param>

private double Total\_Cost()

{

}

/\*\*///////////////////////////////////////////////////////////////

/// <summary>

/// 产生到i城市后，下一个可走城市的集合。并将城市编号加入到openList中。

/// 产生的城市不可以已经存在closedList中

/// </summary>

/// <param name="i"></param>

private void NextCity()

{

openList.Clear();

int temp\_int=Convert.ToInt32(Math.Sqrt(City\_Cost.Length/TraficNum));

for (int i = 0; i < temp\_int; i++)

{

if (closedList.Contains(i) ==false)

{

openList.Enqueue(i);

}

}

}

/\*\*///////////////////////////////////////////////////////////////

/// <summary>

/// 选择应该走那条路，选择完路A后，清空openList，再把A加入到openList中

/// </summary>

/// <returns></returns>

private int choiceRoute()

{

int index = 0;//记录选择的城市

Random random = new Random();

double random\_value =(double) random.NextDouble();//随机选择的概率

int[] temp\_Array=new int [openList.Count];

temp\_Array=openList.ToArray();

double sum\_Message = 0;//openList所有节点的总信息量

for (int i = 0; i < openList.Count; i++)

{

for (int k = 0; k < TraficNum; k++)

{

double eta = 1 / City\_Cost[Pro\_time, temp\_Array[i], k];

sum\_Message = sum\_Message + Math.Pow(Message[Pro\_time, temp\_Array[i], k], alpha) \*Math.Pow(eta, beta);

}

}

double temp=0;

for (int j = 0; j < openList.Count; j++)

{

for (int k = 0; k < TraficNum; k++)

{

double eta = 1 / City\_Cost[Pro\_time, temp\_Array[j], k];

temp = temp + Math.Pow(Message[Pro\_time, temp\_Array[j], k], alpha) \* Math.Pow(eta, beta) / sum\_Message;

if (temp > random\_value)

{

index = temp\_Array[j];

TraficList.Enqueue(k);

break;

}

}

}

openList.Clear();

openList.Enqueue(index);

return index;

}

/\*\*///////////////////////////////////////////////////////////////

/// <summary>

/// 返回每条路所选的运输方式，记录在TraficList中

/// </summary>

/// <returns></returns>

public Queue<int> ReturnTrafic()

{

return TraficList;

}

/\*\*//////////////////////////////////////////////////////////////

public Queue<int> Main\_DW(int l,int n)

{

N= n;

TraficList.Clear();

BestList.Clear();

/\*\*////共循环20次

for (int i = 0; i < 200; i++)

{

/\*\*/

///共有n只蚂蚁n=City\_Cost'number Convert.ToInt32(Math.Sqrt(City\_Cost

.Length/TraficNum))

for (int j = 0; j < Convert.ToInt32(Math.Sqrt(City\_Cost.Length / TraficNum)); j++)

{

openList.Enqueue(l);

closedList.Clear();

int m = 10000;

while (openList.Count != 0 && closedList.Count != Convert.ToInt32(Math

.Sqrt(City\_Cost.Length / TraficNum)))

{

int temp = openList.Dequeue();

Pro\_time = temp;

closedList.Enqueue(temp);

if (n==m)

{

if (BestList.Count == 0)

{

int[] temp\_Array = new int[closedList.Count];

temp\_Array = closedList.ToArray();

for (int k = 0; k < closedList.Count; k++)

{

BestList.Enqueue(temp\_Array[k]);

}

}

if (Get\_Weight(BestList) < Get\_Weight(closedList))

{

BestList.Clear();

int[] temp\_Array = new int[Convert.ToInt32(Math.Sqrt(

City\_Cost.Length / TraficNum))];

temp\_Array = closedList.ToArray();

for (int k = 0; k < Convert.ToInt32(Math.Sqrt(City\_Cost.Length / TraficNum)); k++)

{

BestList.Enqueue(temp\_Array[k]);

}

}

break;

}

NextCity();

m=choiceRoute();

}

}

Change\_Message(BestList,n);//修改信息量

}

return BestList;

}

}

}