目 录

第1章　状态值函数的符号定义

1.1　State Value symbol definition

S: 状态空间大小为n，即$S = \{S_1, S_2, \cdots, S_k, \cdots, S_n\}$

A: 动作空间大小为吗，即$A = \{A_1, A_2, \cdots, A_k, \cdots, A_m\}$

M: 轨迹数目

N: 轨迹长度，也就是当前最大的step 数

即，通过随机策略探索M条轨迹，每条轨迹长度为N

得到的轨迹集合为T

第i 条轨迹 $T_i = \{(s_{i1}, a_{i1}), (s_{i2}, a_{i2}), \cdots, (s_{ij}, a_{ij}), \cdots, (s_{iN}, a_{iN})\}$,

其中$s_{ij} \in S, a_{ij} \in A, i \in \{1, 2, \cdots, M\}$

对每条轨迹进行统计转换：

第i条轨迹中共有状态W_i个，第t个出现的记为$u_{iw_{it}}$,其中,w_{it}为首次出现状态的次序，$w_{it} \in R, 0 < w_{it} < W_i$ 。则有，

首次出现的状态排序：

$U_i = \{u_{i1}, u_{i2}, \cdots, u_{iw_{it}}, \cdots, u_{iW_i}\}$, 其中，$u_{iw_{it}} \in S$

各个状态在轨迹中出现的次数：

$C_i = \{c_{i1}, c_{i2}, \cdots, c_{iw_{it}}, \cdots, c_{iW_i}\}$,其中，$c_{iw_{it}} \in R$

状态S_k在轨迹T_i出现的次序是w_{ik}

在M条长度为N的轨迹中，某个状态S_k出现的总次数P_k:

$P_k = \sum_{i=1}^{M} \sum_{w_{it}=1}^{W_i} c_{iw_{it}} \cdot I(u_{iw_{it}} = S_k)$

或

$P_k = \sum_{i=1}^{M} c_{w_{ik}}$

每条轨迹中状态S_k之前出现的状态统计数据之和为BE_k(BEfore k):

$BE_k = \sum_{i=1}^{M} \sum_{j=1}^{w_{ik}-1} c_{ij}$

之后出现的状态统计之和 AF_k (AFter k):

$AF_k = \sum_{i=1}^{M} \sum_{j=w_{ik}+1}^{W_i} c_{ij}$

根据P_k 对各个状态进行从高到低排序，最高的状态我们认为是最容易到达的状态，而且是熵最高的状态，得到的结果记为O(order of state):

$O = \{o_1, o_2, \cdots, o_i, \cdots, o_n\}$

对应的BE,AF 为 OBE,OAF(因为之前三个特征都是以状态为索引的，所以是可以同时排序的)

那么我们现在要做的两种比较方式是：

1.$\frac{\partial OBE}{\partial O} = 0$

2.p_k很小，但是BE很大的角落也有可能是我们需要的点

接下来要讨论如何定义Im(s)

另外几种不同的情况都可以试试，比如：

1. BE_k

2.$\frac{BE_k}{AF_k}$

3 $\frac{BE_k}{\sum_{i=1}^{M}(W_i - w_{ik})}$

4.$\frac{\sum_{i=1}^{M} w_{ik}}{\sum_{i=1}^{M}(W_i - w_{ik})} BE_k$

5.$\frac{OBE}{O}$

。。。

第2章　算法流程

2.1　uptrend State Value based RL

语言描述：

初始化轨迹数目 M，轨迹最大长度N，每轮增加的步数Δ，随机策略π_0

1.采用策略π_i 探索M条轨迹，每条最多N 步，得到轨迹集合 $trjs_i$

2.用$trjs_i$ 更新$Im(s)_i$ （Im 的计算方法要保证每次都走的路线不重复计入，也就是Im 学到了就固化在策略中，不再进行重复学习）

3.$Im(s)_i$代替r,训练策略，（或者找到其它的算法，将 $Im(s)_i$作为$V(s)_i$）

4.得到策略π_{i+1}

5.增加探索步骤 $N = N + \Delta$

6.重复步骤2-5，直到得到真正的奖励为止。

7.有了奖励之后，我们要按奖励值大比重更新我们的状态，并重新探索

8。如果我们想要得到最优策略，就要在得到奖励（要是成功的奖励）之后，减小探索步骤N,希望能更快到达终点

2.2　Evluation

评价方法有两种思路：

1.每个回合得到最终奖励的次数

2.固定次数完成一个任务的可能性