

CSDN首页 (<http://www.csdn.net?ref=toolbar>)学院 (<http://edu.csdn.net?ref=toolbar>)下载 (<http://download.csdn.net?ref=toolbar>)

更多 ▼

下载 CSDN APP (<http://www.csdn.net/app/?ref=toolbar>)写博客 (<http://write.blog.csdn.net/postedit?ref=toolbar>)登录 (<https://passport.csdn.net/account/login?ref=toolbar>) | 注册 (<http://passport.csdn.net/account/mobile/register?ref=toolbar&action=mobileRegister>)首页 (<http://blog.csdn.net/>) 移动开发 (<http://blog.csdn.net/mobile/newarticle.html>)

全部 ▼

CSDN (<http://www.csdn.net>)

目录 那么蒙特卡洛树搜索(Monte Calro Tree Search, MCTS)究竟是啥



喜欢

原创 2016年03月26日 14:41:21

8141

0

3



收藏

同时发布于：<http://www.longgaming.com/archives/214>
(<http://www.longgaming.com/archives/214>)

评论

Intro



分享 最近阿法狗和李师师的人机大战着实火了一把，还顺带捧红了柯杰，古力等一干九段。虽然我从小学的是象棋，对围棋也只是略知一二，但是棋魂还是对我影响颇深的启蒙漫画，于是还是凑热闹看了几盘大战。其中蒙特卡洛树搜索（Monte Calro Tree Search，MCTS）就多次被各路砖家提及。想必各位选过AI课的同学都知道Minimax tree和Alpha-beta剪枝等各种技巧。归根到底，这些传统AI技术还是暴力搜索，将游戏中所有的可能性表示成一棵树，树的第N层就代表着游戏中的第N步。树的node数是随着树的深度成指数增长的，不考虑剪枝，每个node都是需要进行估值的。

Minimax tree在诸如象棋，黑白棋等传统游戏中取得了巨大的成功，主要原因还是有两个，

1. 游戏本身的探索空间相对较小（比如象棋大概是 10^{50} ），配合剪枝，开局和杀棋棋谱，非平衡树探索等优化技术，加上并行计算和Iterative Deepening，使得探索到树的深层甚至底层成为可能。
2. 搜索的最终目的就是找出对自己最有利的一步，而判断是不是有利自然需要一定的评判标准。一般我们用一个评价函数来作为标准。象棋等游戏的子有不同的强弱，并且有明确的目的性（诸如杀死对方的王），容易人工设计出或者通过机器学习得出一个良好的评价函数来正确评估一步落子所引发的后续局面。

natsu1211 (<http://blog.cs...>)

+ 关注

(<http://blog.csdn.net/natsu1211>)

码云

未开通

原创

粉丝

喜欢

(https://gite
utm_sourc

31

24

0

他的最新文章

更多文章 (<http://blog.csdn.net/natsu1211>)

工作中能派上用场的GIT命令和HUB命令

(/natsu1211/article/details/51816246)

那么蒙特卡洛树搜索(Monte Calro Tree Search, MCTS)究竟是啥

(/natsu1211/article/details/50986810)

手把手教你在openshift上搭建
wordpress博客（二）

(/natsu1211/article/details/40248531)

手把手教你在openshift上搭建
wordpress博客（三）

(/natsu1211/article/details/40248831)

但是围棋为什么一直搞不定？主要原因也还是那两个，

1. 围棋的棋盘是19x19，除去不能下在眼里，围棋第N步的可能下法有 $362-N$ 种，探索空间要大很多(大概是 10^{171})。传统方法往下算几层基本就算不动了，难以正确找到最有利的下法。
2. 难以找到一个合适的评价函数。和象棋的子有强弱不同，围棋中的每个子很难给予一个明确的价值，所以对整个局面也难以做出一个正确的判断。

所以传统方法一直无法攻克围棋的难关。既然以前的路是死路一条（感兴趣的同学可以看看[2]，里面有前人在这条路上的各种探索，他们根据围棋的规则对盘面上的子进行各种抽象，创造出了非常复杂的评价函数，并且制定了一些IF-THEN规则来教程序下棋，可惜依然无法达到专业棋手的水平），那么只能另辟蹊径，这条蹊径就是蒙特卡洛树探索（下文MCTS）了。但是MCTS究竟是个什么玩意？蒙特卡洛方法用到树探索里？老实说看直播的时候我还不是很清楚。最后还是翻看了一些资料和论文，包括阿法狗的论文，才终于明白是怎么回事。



目录



喜欢



收藏



评论



分享

Pure Monte Carlo Go

首先，蒙特卡洛方法大家都知道，利用随机采样的样本值来估计真实值，理论基础是中心极限定理。先不考虑树搜索的事，就单纯的用蒙特卡洛方法来下棋[1, 2]（最早在1993年被提出，后在2001被再次提出）。我们可以简单的用随机比赛的方式来评价某一步落子。从需要评价的那一步开始，双方随机落子，直到一局比赛结束。为了保证结果的准确性，这样的随机对局通常需要进行上万盘，记录下每一盘的结果（比如接下来的落子是黑子，那就根据中国规则记录黑子胜了或输了多少子），最后取这些结果的平均，就能得到某一步棋的评价。最后要做的就是取评价最高的一步落子作为接下来的落子。也就是说为了决定一步落子就需要程序自己进行上万局的随机对局，这对随机对局的速度也提出了一定的要求。和使用了大量围棋知识的传统方法相比，这种方法的好处显而易见，就是几乎不需要围棋的专业知识，只需通过大量的随机对局就能估计出一步棋的价值。再加上诸如All-Moves-As-First等优化方法，基于纯蒙特卡洛方法的围棋程序已经能够匹敌最强的传统围棋程序。

Monte Carlo Tree Search

编辑推荐

最新专栏

[蒙特卡罗树搜索+深度学习 -- AlphaGo...](#)[AlphaGo背后的搜索算法：蒙特卡罗树...](#)[蒙特卡洛树搜索介绍 \(/z84616995z/arti...](#)[Introduction to Monte Carlo Tree Searc...](#)[清理注册表后导致Oracle listener起不来..](#)

在线课程

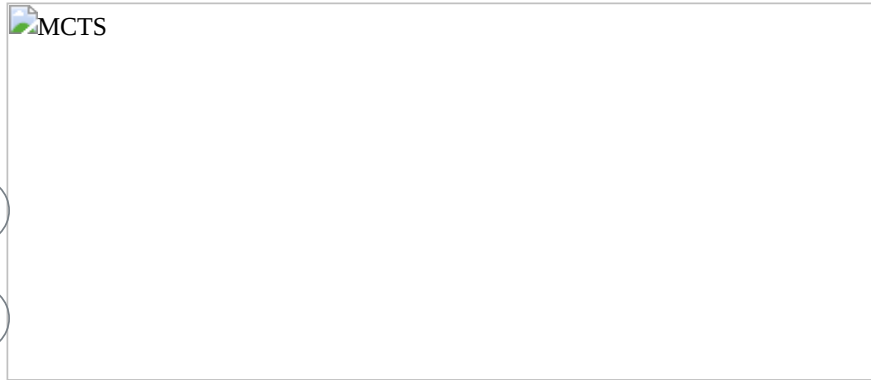


Python全栈工程师特训班
(http://edu.csdn.net/huiyi
utm_source=blog9)
Course/detail/569?
utm_source=blog9)



Blink在阿里集团的应用实
践
(http://edu.csdn.net/huiyi
Course/detail/531?
utm_source=blog9)

既然蒙特卡洛的路似乎充满着光明，我们就应该沿着这条路继续前行。MCTS也就是将以上想法融入到树搜索中，利用树结构来更加高效的进行节点值的更新和选择。一般来说，MCTS操作分为4个部分[3]，



([http://www.longgaming.com/wp-](http://www.longgaming.com/wp-content/uploads/2016/03/QQ图片20160325212131.png)

[content/uploads/2016/03/QQ图片20160325212131.png](http://www.longgaming.com/wp-content/uploads/2016/03/QQ图片20160325212131.png))

1. Selection

从root node开始，根据Tree Policy依次选择最佳的child node，直到到达leaf node。Tree Policy就是选择节点（落子）的策略，它的好坏直接影响搜索的好坏，所以是MCTS的一个研究重点。比如上面说的选择平均赢子数最多的走法就是一种Tree Policy。而目前广泛采用的策略有UCT，后面会详说。

2. Expansion

扩展leaf node，将一个或多个可行的落子添加为该leaf node的child node。具体如何暂开，各个程序各有不同。

3. Simulation

根据Default Policy从扩展的位置下棋到终局。完全随机落子就是一种最简单的Default Policy。当然完全随机自然是比较弱的，通过加上一些先验知识等方法改进这一部分能够更加准确的估计落子的价值，增加程序的棋力。

4. Backpropagation

将Simulation的结果沿着传递路径反向传递回root node。

UCT & UCB



目录



喜欢



收藏



评论



分享

那么目前广泛采用的UCT的又是个什么呢。UCT的全称是UCB for Trees [4], UCB指的是Upper Confidence Bounds [5], 不过光看名字果然还是不明所以。

其实这个算法是有很深刻的背景的, 老虎机大家都玩过吧, 假设有一排总共K台老虎机, 每台机子的期望收益是不一样的并且是固定的, 但是我们并不知道各台机子的设置(不知道收益的分布), 假设每次投入的本金都是一样的, 那么如何规划每次玩哪台机子来使自己获得最大收益呢, 这就是K-armed bandit problem了。

容易想到, 如果我们每次都能够将本金投入到期望收益最高的那台机子上, 自然可以获得理论上最大的收益。但是可惜我们并不知道哪台机子是最好的, 所以一方面我们需要分出成本去了解各个机子的收益情况, 但是在收益不高的机子上投入太多成本自然是亏损的, 所以也需要在当前发现的收益最高的老虎机上投入成本。这是一个

目录 exploitation-exploration (收获-探索) 困局。我们需要一个好的策略来掌握好探索和收获之间平衡。而UCB是一个

喜欢 (准确说是一类) 能有效解决这个问题方法。虽然要说清楚这个方法并不容易, 不过还是让我们再深入一点, 首先, 将由于没有选中最佳机器所造成的累积亏损记为regret(t), t为玩老虎机的总次数。这个问题有一个理论极限, 在t足够大的时候, 一定有办法将regret(t)抑制在ln(t)数量级上, 但是不可能比ln(t)数量级还要小。当然, 要达

收藏 到理论值是非常麻烦的, UCB能够用一个相对简单的方法让维持在ln(t)数量级上(虽然常数项要大一些)。UCB的基本想法是不选取平均收益最高, 而是选取置信上限最高的机器。为什么要这么做呢, 举一个极端的例子, 假如

有两台老虎机, 一台有0.8概率中奖的机子和一台只有0.2概率中奖的机子, 我们可以简单的把每台机子各玩两次, 但是很不幸的是概率0.8的那台机子两次都没能中奖, 而0.2概率的机子两次都中奖了, 如果我们采用取最高均值的

评论 策略, 那么0.2概率的那台机子将会被选中, 并且因为这台机子的平均值不会变为0, 胜率较高的那台机器就没有机

分享 会被选中了。这个方法的问题在于没有进行足够的探索, 而是将成本都投入到了收获中。实际上, 平均收益只是对实际收益期望的一个估计, 特别是采样数较少的时候, 均值很可能会与期望产生较大的偏离, 我们不应该直接把

样本的均值当成实际的期望, 而是要对期望给出一个置信区间和置信水平(概率)。至于要如何计算这个置信区间, 就要运用Chernoff-Hoeffding bound了(推导可以参见[6]),

$$P\left(\frac{1}{n} \sum_{i=1}^n X_i \leq \mu - a\right) \leq e^{-2a^2 n}$$

其中n代表样本数(放在老虎机问题里面就是该台机子被选中的次数), μ 代表期望, 稍微变形一下,

$$P\left(\mu \geq \frac{1}{n} \sum_{i=1}^n X_i + a\right) \leq e^{-2a^2 n}$$

这个式子可以用来确定期望的置信上界, UCB方法中的UCB1使用置信水平 $\frac{1}{t^4}$, 根据等式 $\frac{1}{t^4} = e^{-2a^2 n}$, 就可以解

$$出 a = \sqrt{\frac{2 \ln t}{n}},$$

加上目前的平均收益, 就得到了收益期望的置信上限, 同时也是UCB1的定义,

$$\bar{x}_j + \sqrt{\frac{2 \ln t}{n_j}}$$

其中j是老虎机的编号。

如果我们乐观一些，认为期望能够达到置信上限，那么选择具有最高UCB1的老虎机就成为了一个更好的策略。同时，我们可以注意观察一下UCB1的定义，前半部分是当前的平均收益，这代表着收益，后半部分则是相对于平均收益的偏移，被选中次数越少则偏移会变得越大，有更大的可能被选中，这就保证了当前收益不是那么高的机器也能够被探索到。这就很好的达成了收获和探索之间的平衡。当然，UCB的思想并不局限运用于老虎机，在围棋中，从多种可能落子中选择一步最好的同样是类似的问题，也需要掌握好收获与探索的平衡（但是也并不完全相同 1.老虎机问题的目的是要尽量小的累积regret，而围棋只要能找到最好的那一步就够了，并不在乎探索途中的

regret。2. 期望是会变动的。所以UCT对UCB1进行了一定的变动，实际使用的是 $\bar{x}_j + 2C_p \sqrt{\frac{\ln t}{n_j}}$ ，也就是给偏移部分加上了一个正常数项）。而UCT就是将上述的UCB的思想运用到了树搜索中，在Selection的阶段，总是选择UCB值最大的node，直至leaf node。

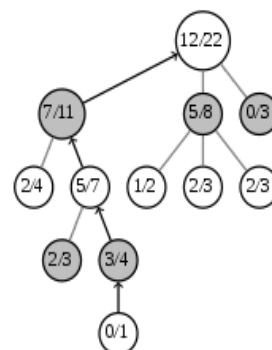
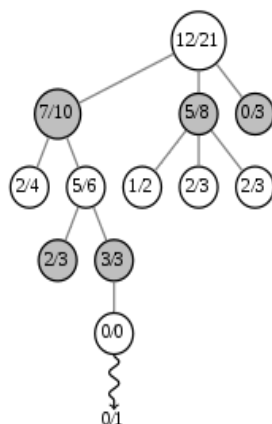
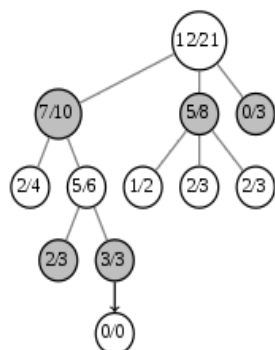
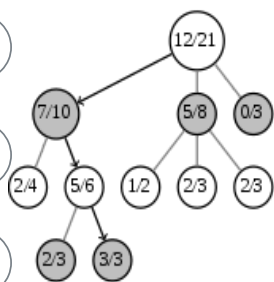
弄明白了MCTS和UCT的原理，这里我们直接用wiki上面的例子来走一遍MCTS的操作，Tree Policy用的就是UCT，

Selection

Expansion

Simulation

Backpropagation



首先，已经有一定量的节点被加入到树中，从root node开始逐层往下选择UCB值最高的node（图中的11/21, 7/10, 5/6, 3/3），直到遇到leaf node（图中的3/3），为该节点添加子节点（图中的0/0），随后从此节点开始进行simulation，得到结果并按照搜索下来的路径将结果返回（注意这里的结果和纯蒙特卡洛方法并不一样，只记录胜负结果而不是子数，胜返回1，负则返回0），更新在路径上的每一个节点，在到达结束条件之前，这些操作会一直循环进行下去。可以注意到MCTS是逐步构建搜索树的，并且树是非平衡发展的，更优的node有可能被搜索的更深。

确实MCTS的出现让围棋AI的棋力得到了进步，最强的围棋程序们也都统统采用了MCTS，但是它们的棋力离顶尖棋手依然有很远的距离。



目录



喜欢



收藏



评论



分享

AlphaGo

就在MCTS的发展似乎也要山穷水尽的时候，深度学习的发展让AlphaGo横空出世。但是仔细想想又不算是横空出世，一作Huang正是十年前最强的围棋程序CrazyStone的作者，为MCTS发展做出了贡献的Coulom的学生，David Silver也是早在10年前就开始将强化学习运用在MoGo等程序中。可谓十年磨一剑了。人机大战的结果我们也都看见了，相比之前的Zen，CrazyStone等程序，阿法狗的棋力有了飞跃性的提升（从nature的那篇论文[7]中可以看到，阿法狗对这些之前最强的程序的胜率几乎达到100%，应该可以说这是深度学习的威力）。



目录

我对深度学习并不了解，而且已经有一篇田博士的讲解文章了[8]。这里我只想说说对MCTS部分的认识，



喜欢

1. 在AlphaGo中，使用的Tree Policy可以算是变形的UCT，

$$a_t = \underset{a}{argmax} (Q(s_t, a) + u(s_t, a))$$



收藏

这里的s指的是一个state，代表一个盘面。a是指action，指下一步棋。从一个state做一个action，就会迁移到另一个state，当然本质上和我们之前用大白话做的分析没啥区别。Q是累积平均胜率，u则是对于Q的偏移，也就是选择让Q+u最大的一步棋。Q在论文中的定义如下，



评论

$$N(s, a) = \sum_{i=1}^n 1(s, a, i)$$

$$Q(s, a) = \frac{1}{N(s, a)} \sum_{i=1}^n 1(s, a, i) V(s_L^i)$$

$$V(s_L) = (1 - \lambda) v_\theta(s_L) + \lambda z_L$$



分享

$1(s, a, i)$ 函数代表第i次simulation中边(s, a)所通向的节点有没有被访问到（论文中用的是边(s, a)，也没有本质区别。因为之前的分析用的都是节点，这里为了方便也说是节点）， $N(s, a)$ 自然是所有的simulation中(s, a)所指向节点的访问次数。

而 $V(s_L)$ 则是一次simulation后，那个leaf node的胜率（也就是反向传递回去的结果），但是这里的胜率并不简单是simulation的结果(z_L)，而是综合了估值网络所给出的胜率($v_\theta(s_L)$)， λ 是权重系数。

而 $u(s_t, a)$ 的定义如下，

$$u(s, a) \propto \frac{P(s, a)}{1 + N(s, a)}$$

$P(s, a)$ 是策略网络输出的当前局面s下落子a的概率。正如[8]中所说，说明探索刚开始时，策略网络的影响比较大，而随着探索的进行，策略网络的影响慢慢降低，偏移慢慢变小，而真正在simulation中取得更好胜率的落子会更容易被选中。

2. [8]中提到的访问leaf node累积访问到40次才进行expansion，这是一个implementation上的细节，不过却是很重要的。

References

- [1] Monte Carlo Go, Bernd Brügmann, 1993
- [2] Computer Go: An AI oriented survey, Bruno Bouzy et al., 2001
- [3] A Survey of Monte Carlo Tree Search Methods, Cameron Browne et al., 2012
- [4] Bandit based Monte-Carlo Planning, Levente Kocsis et al., 2006
- [5] Finite-time Analysis of the Multiarmed Bandit Problem, PETER AUER et al., 2002
- [6] <https://www.zybuluo.com/qqiseeu/note/109942> (<https://www.zybuluo.com/qqiseeu/note/109942>)
- [7] Mastering the game of Go with deep neural networks and tree search, David Silver et al., 2016
- [8] <http://zhuanlan.zhihu.com/yuandong/20607684> (<http://zhuanlan.zhihu.com/yuandong/20607684>)



喜欢 (至于格式嘛, 请允许我偷个懒= =)

收藏

评论

分享

目录

版权说明: 本文为博主原创文章, 未经博主允许不得转载。

举报

标签: ai (<http://so.csdn.net/so/search/s.do?q=ai&t=blog>) / MCTS (<http://so.csdn.net/so/search/s.do?q=MCTS&t=blog>)

相关文章推荐

蒙特卡罗树搜索+深度学习 -- AlphaGo原版论文阅读笔记 (</dinosoft/article/details/50893291>)

原版论文是《Mastering the game of Go with deep neural networks and tree search》,有时间的还是建议读一读,没时间的可以看看我这篇笔记凑...



Dinosoft (<http://blog.csdn.net/Dinosoft>) 2016-03-23 01:01 14007

AlphaGo背后的搜索算法：蒙特卡罗树搜索 && alphago 代码

[:/real_myth/article/details/50884721](http://real_myth/article/details/50884721)

目录代码：<https://github.com/Rochester-NRT/AlphaGo> AlphaGo背后的搜索算法：蒙特卡罗树搜索 ...



Real_Myth (http://blog.csdn.net/Real_Myth) 2016-03-14 11:07 2747

喜欢



收藏



评论



《程序员看天下》实战：揭秘携程大数据的应用处理

一直以来，携程拥有海量数据，如何存储、分析和应用这些数据一直是部门痛点所在！携程大数据团队将会给出什么样的解决方案呢？开源产品的选型和运维又该如何抉择呢....



http://www.baidu.com/cb.php?c=lgF_pyfqHmsrHTYrHc0lZ0qnfK9ujYzP1D4P1m40Aw-544c3rHnYnHb0TAq15HfLPWRznjb0T1Y1uju-n1fLmWF9nWnvPADd0AwY5HDdnjTdPWcsn1f0lgF_5y9YIZ0lQzq-uZR8mLPbUB48ugfEpZNGXy-jULNzTvRETVnZpyN1gww-IA7GUatvPjqdAdxTvqdThP-5yF_UvTkn0KzujYk0AFV5H00TZcqn0KdpyfqHRLPjnvnfKEpyfqHc4rj6kP0KWpyfqP1civrHnz0AqLUWYs0ZK45HcsP6KWThnqPjTLnjb

蒙特卡洛树搜索介绍 (/z84616995z/article/details/50915952)


2015年9月7日周一 由Jeff Bradberry留 与游戏AI有关的问题一般开始于被称作完全信息博弈的游戏。这是一款对弈玩家彼此没有信息可以隐藏的回合制游戏且在游戏技术里没有运气元...



z84616995z (<http://blog.csdn.net/z84616995z>) 2016-04-03 12:41 2638

Introduction to Monte Carlo Tree Search (/amds123/article/details/71056942)



<https://jeffbradberry.com/posts/2015/09/intro-to-monte-carlo-tree-search/> Introduction to Mon...

 AMDS123 (<http://blog.csdn.net/AMDS123>) 2017-05-01 17:27  5639

清理注册表后导致Oracle listener起不来 (/winderain/article/details/1325666)


清理注册表后导致Oracle listener起不来a.现象LSNRCTL> start启动tnslsnr：请稍候...Failed to start service, error 3.TNS-12

5...

 winderain (<http://blog.csdn.net/winderain>) 2006-10-08 11:42  936
目录

该找工作了 (/chengzm723/article/details/1325668)


喜欢

 马上就要去实习了，说要一下子离开这，还有点舍不得

 chengzm723 (<http://blog.csdn.net/chengzm723>) 2006-10-08 11:42  171
收藏

堆和栈的区别 (/natsu1211/article/details/8518387)

评论


 堆 (heap)是计算机科学中一类特殊的数据结构的统称。堆通常是一个可以被看做一棵树的数组对象。堆总是满足下列性质：
堆中某个节点的值总是不大于或不小于其父节点的值； 堆总是一...

分享

 natsu1211 (<http://blog.csdn.net/natsu1211>) 2013-01-18 19:45  408

蒙特卡洛树算法 (MCTS) (/jaster_wisdom/article/details/50845090)

实质上可以看成一种增强学习 蒙特卡罗树搜索(MCTS)会逐渐的建立一颗不对称的树。可以分为四步并反复迭代：(1)选择 从根节点，也就是要做决策的局面R出发向下选择一个最迫切需要...

 Jaster_wisdom (http://blog.csdn.net/Jaster_wisdom) 2016-03-10 11:29  4394

蒙特卡洛算法 (/acdreamers/article/details/44978591)

今天开始研究Sampling Methods，接下来会分为四部分进行讲解。本文是开篇文章，先来讲讲蒙特卡洛算法。 Contents

1. 蒙特卡洛介绍 2. 蒙特卡洛的应用 ...



ACdreamers (<http://blog.csdn.net/ACdreamers>) 2015-04-12 17:23 22770

蒙特卡罗树搜索+深度学习 -- AlphaGo原版论文阅读笔记 (/ycl295644/article/details/52040143)



蒙特卡罗树搜索+深度学习 -- AlphaGo原版论文阅读笔记 原版论文是《Mastering the game of Go with deep neural networks
目录an...



ycl295644 (<http://blog.csdn.net/ycl295644>) 2016-07-26 22:13 779

喜欢



蒙特卡洛树搜索(MCTS)进行模拟的实现流程 (/u010296599/article/details/59111470)



首先，要明确的一点是，算法并不需要了解游戏的领域知识。 在一个游戏模拟过程中，相关决策的组合可能是一个很大的
数，我们如何控制这个模拟行为是满足一定时间上的限制的。我们允许一个参数来控制时间...



评论 u010296599 (<http://blog.csdn.net/u010296599>) 2017-03-01 16:58 231



分享



(<http://download.csdn.net/detail/u013784415/7562673>)

binary search tree 二叉搜索树的C++实现，有插入、删除、查找、查找最大最小等功能
(<http://download.csdn.net/detail/u013784415/7562673>)

2014-06-27 22:27 4KB

下载



(http://download.csdn.net/detail/xk_casanova/6496003)

monte calro介绍 (http://download.csdn.net/detail/xk_casanova/6496003)



2013-11-03 15:56 756KB

[下载](#)

蒙特卡洛 (Monte Carlo) 积分的入门 (/minenki/article/details/8916503)

转自 <http://www.opengpu.org/forum.php?mod=viewthread&tid=198> 概率密度描述了一个随机变量的值在区域的概率分布。

概率密度函数 (p...



 minenki (<http://blog.csdn.net/minenki>) 2013-05-12 10:55  930

目录

♥蒙特卡洛方法 (Monte Carlo) (/u014665416/article/details/51985879)


喜欢转载于: <http://www.ruanyifeng.com/blog/2015/07/monte-carlo-method.html> 蒙特卡罗方法入门 本文通过五个例子, 介绍蒙特

卡...

收藏  u014665416 (<http://blog.csdn.net/u014665416>) 2016-07-21 19:19  494

评论



蒙特卡洛方法 (Monte Carlo Method) (/coffee_cream/article/details/66972281)

 蒙特卡洛 (Monte Carlo, MC) 方法——增强学习方法之一

分享  coffee_cream (http://blog.csdn.net/coffee_cream) 2017-03-27 15:36  506

蒙特卡洛 (Monte Carlo) 法求定积分 (/baimafujinji/article/details/53869358)

蒙特卡洛 (Monte Carlo) 法是一类随机算法的统称。随着二十世纪电子计算机的出现, 蒙特卡洛法已经在诸多领域展现出了超强的能力。在机器学习和自然语言处理技术中, 常常被用到的MCMC也是由此发展而来...

 baimafujinji (<http://blog.csdn.net/baimafujinji>) 2016-12-25 15:50  3722

