**3.14**

**a.**错误。由于A\*启发函数的不同，可能会存在深度优先搜索扩展结点少于A\*结点。例如一个n层二叉树，解位于第n层的左边节点。对于深度优先搜索，其只需要探索2n-1个结点，A\*至少需要探索2n-1个节点。于是此时深度优先搜索拓展的结点小于等于A\*探索的结点。

**b.**正确。h(n)=0满足非负的要求，此时等价于宽度优先搜索，对于八数码问题是可采纳的启发式。

**c.**错误。尽管感知器、状态、行动是连续的，但是可以通过一定程度的离散化趋近连续，故A\*算法在机器人学中有用处。

**d.**正确。宽度优先搜索在有限深度、有限状态情况下，总能保证寻找到最优路径，是完备的。

**e.**错误。由于有车无法跳跃棋子，曼哈顿距离并不能作为可采纳的启发式。

**3.16**

**a.**形式化

状态：状态由轨道块的组合方式决定

初始状态：任选一个工件均可作为初始状态

行动：与下一个工件进行拼接(拐角对应，拼合无缝隙)

转移模型：行动会产生期待的后果

目标测试：所有零件拼接成铁路，无重叠的轨道，无松动

路径耗散：无

**b**.深度优先搜索。由于问题有解，于是通过深度优先方式就能够得到最优解，只是最糟糕的情况变成了遍历求解，问题求解速度可能比较慢。

**c**.要是想组成一张能够运行的铁路路线，那么所有的积木块紧密贴合，即图上的扇入等于扇出。图3.32一共有12+16+2\*2+2个扇出，12+16+2+2\*2个扇入，此时扇入等于扇出，问题有解。如果拿出任何一个开叉块，扇入不等于扇出，问题无解。

**d**.考虑最差的情况-遍历，当积木不可翻转时，一共有32!种可能。由于翻转的作用，上界应为32!\*2\*2\*(2\*2)=8\*32!

**3.20**

**a**.深度优先搜索-图搜索

**b.**记有灰尘坐标位置为(1,1)(1,2)(1,3)，Agent坐标(2,2)。走一步的代价为1，清扫灰尘代价为1，扫干净奖励10。Agent拥有全局探测灰尘的能力。

最优行动序列为:

(2,2)->(2,3)->(1,3)->(1,2)->(1,1)

(2,2)->(2,1)->(1,1)->(1,2)->(1,3)

走路总代价为4，清扫代价为3，奖励30。性能23

**c**.随即生成灰尘坐标位置，仍为b中的(1,1)(1,2)(1,3)，agent坐标(2,2)。走一步的代价为1。当前agent只能搜索以自身为中心3\*3的灰尘位置。初始时进行搜索，当清扫完上一个时间搜索的地方时，进行下一次搜索。每次搜索代价为2，走一步代价为1，清扫灰尘代价为1，扫干净奖励10。

最优行动序列为：

(2,2)->(2,3)->(1,3)->(1,2)->(1,1)

(2,2)->(2,1)->(1,1)->(1,2)->(1,3)

走路总代价为4，清扫代价为3，搜索代价为4，奖励30。性能19

**d.**设环境仍是3\*3，最好的搜索agent性能为19。

由于搜索agent走了4步，假设简单的随机反射型agent也走了4步，那么性能为23(性能最好)的概率为,性能为-4的概率至少为，则性能期望一定小于19。在这种情况下，最好的搜索agent性能高于简单的随机反射型agent。

**e.**当为n\*n时，灰尘的状态空间有。当时，通过搜索agent求解复杂度非常高。此时随机反射型agent的效率会比较高。

**3.25**

当w=1时，算法是最优的。

W=0时，算法等价于不计路径耗散的搜索

W=1时，算法等价于A\*算法

W=2时，算法等价于贪婪算法