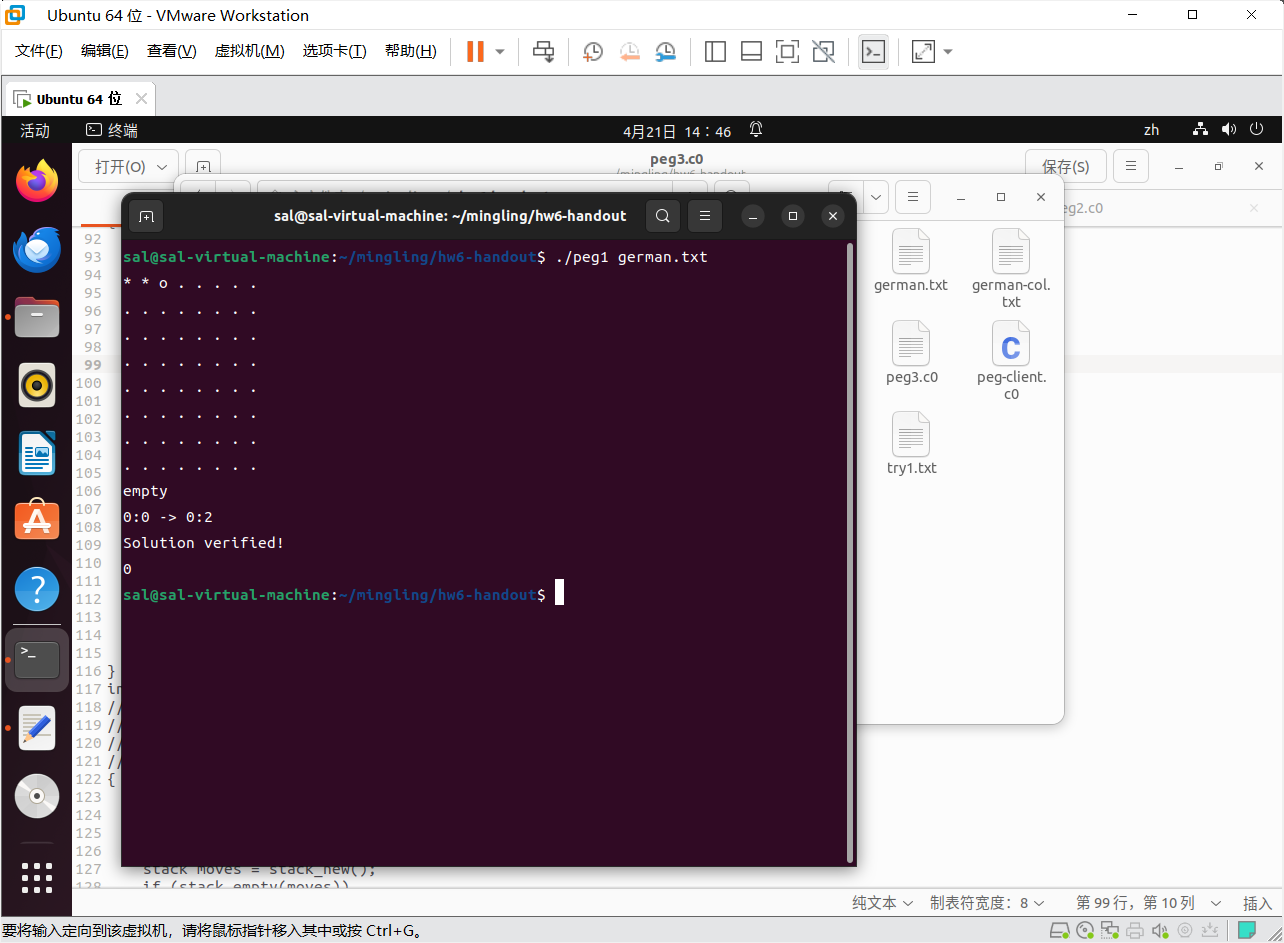
### 任务1

solve(B, S)：

对于棋盘B，穷举所有棋子，来找到当前棋盘状态下可能的移动方式，然后采用此移动方式得到新的棋盘B0，然后递归调用solve（B0，S）；当棋盘上棋子数为1时，将此时的移动方式压栈，并返回。

./peg1 german.txt



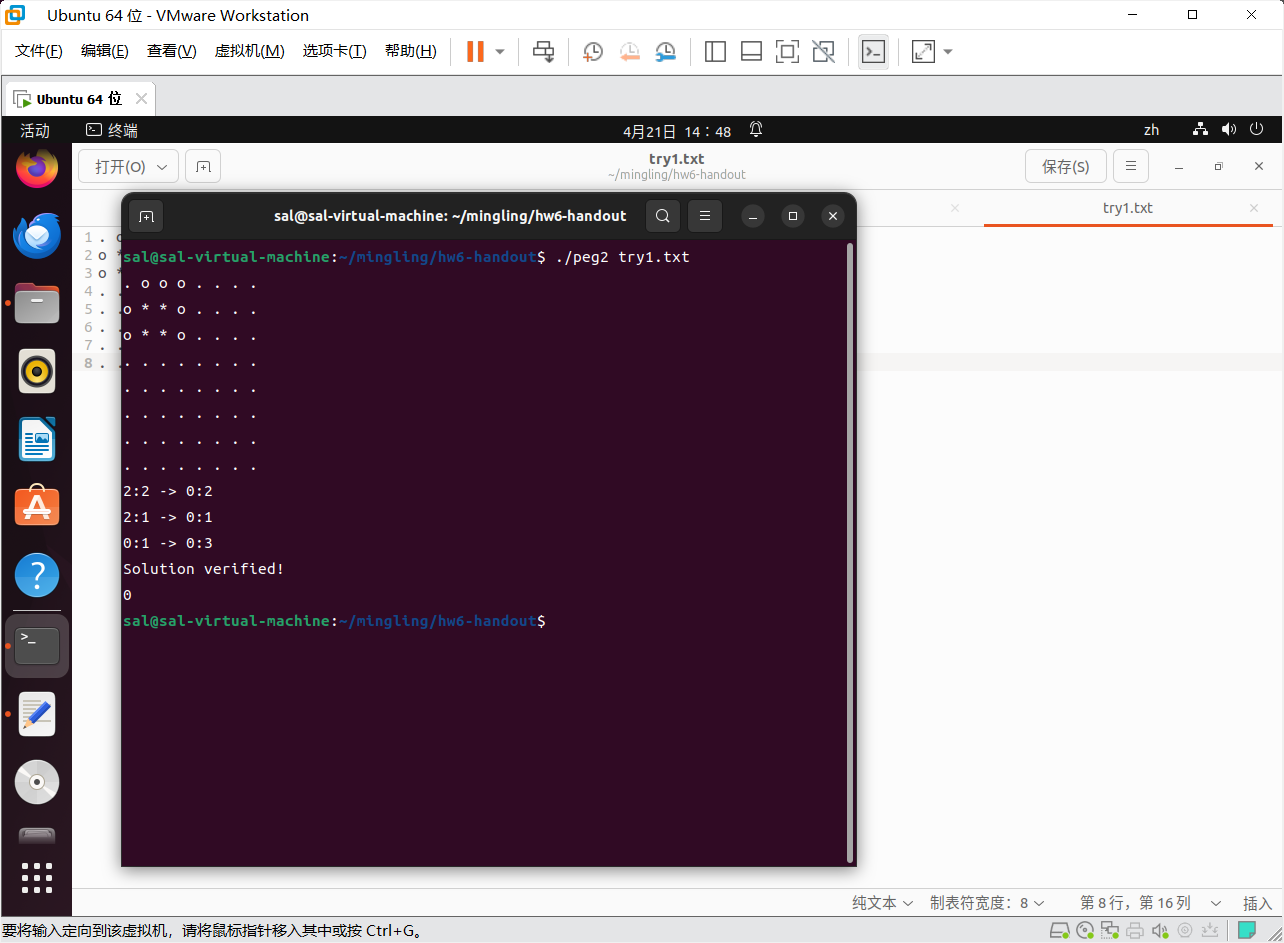
## 2 解一个不确定的孔明棋局

由于棋盘的移动状态并不唯一，则使用一个栈来保存所有的移动方式，每次使用栈顶的移动方式进入递归，如果失败，则逆向使用此移动方式还原棋盘。直至栈空为止。

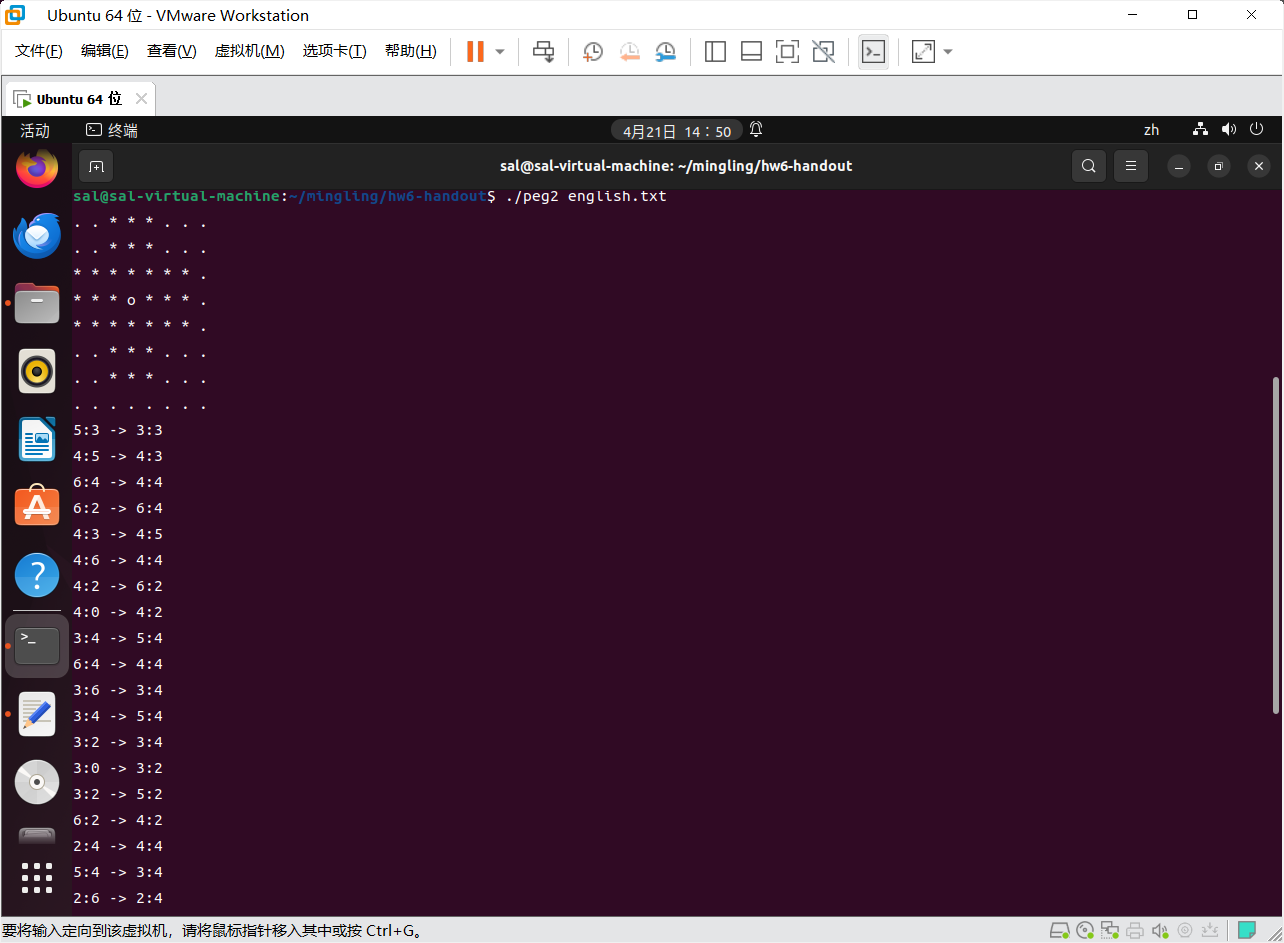
对于一个简单案例：

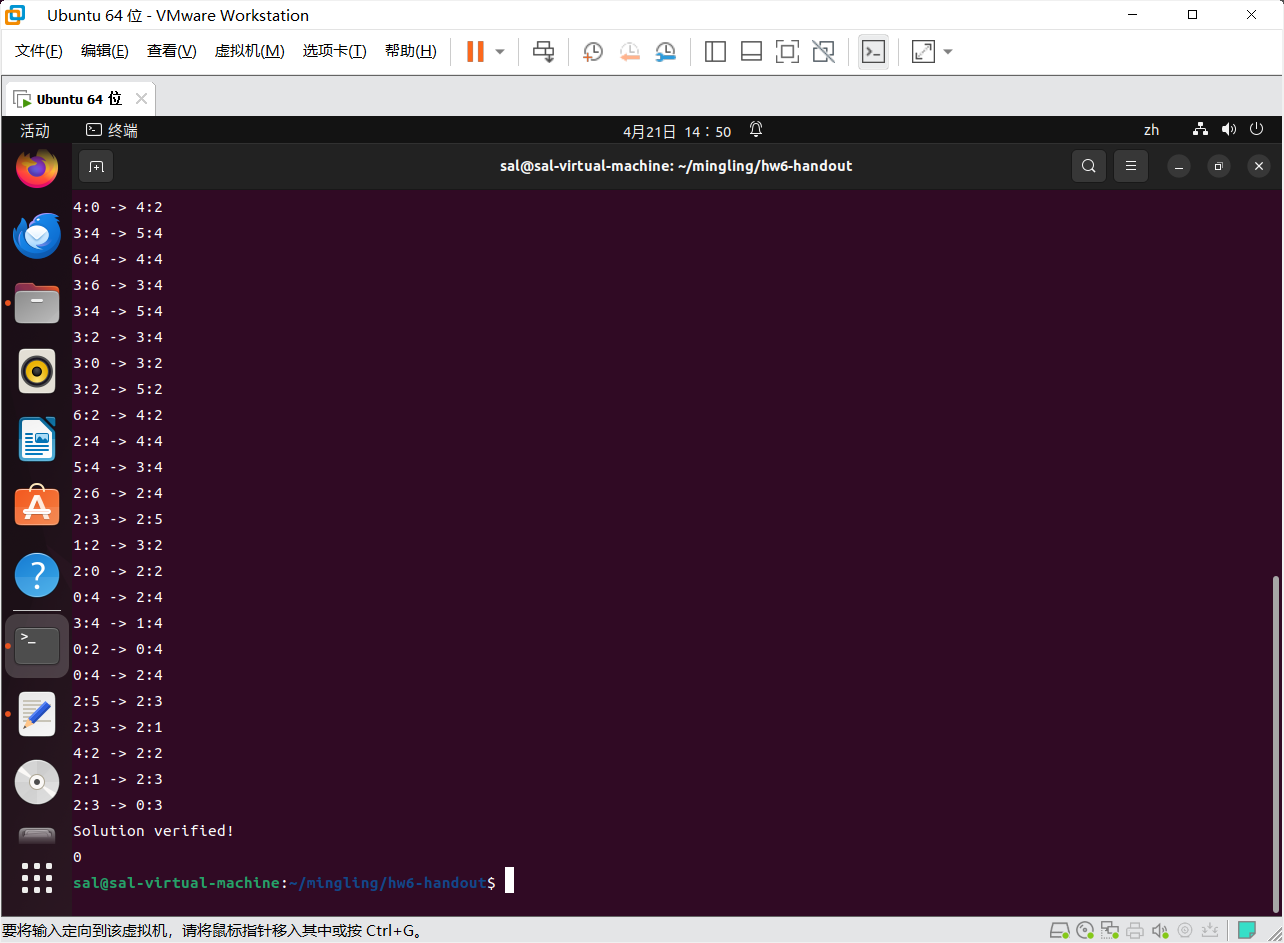


./peg2 try1.txt



./peg2 english.txt



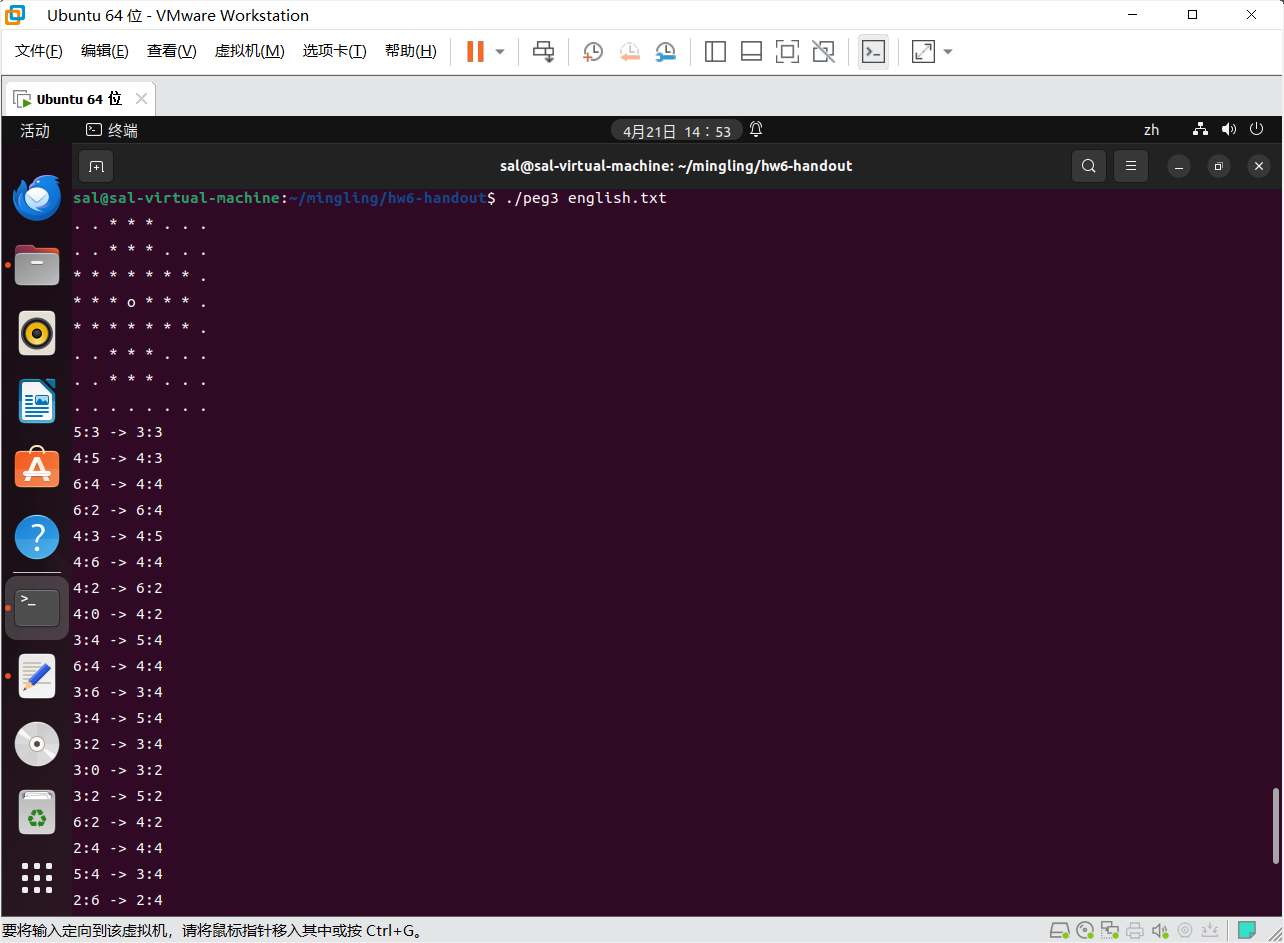


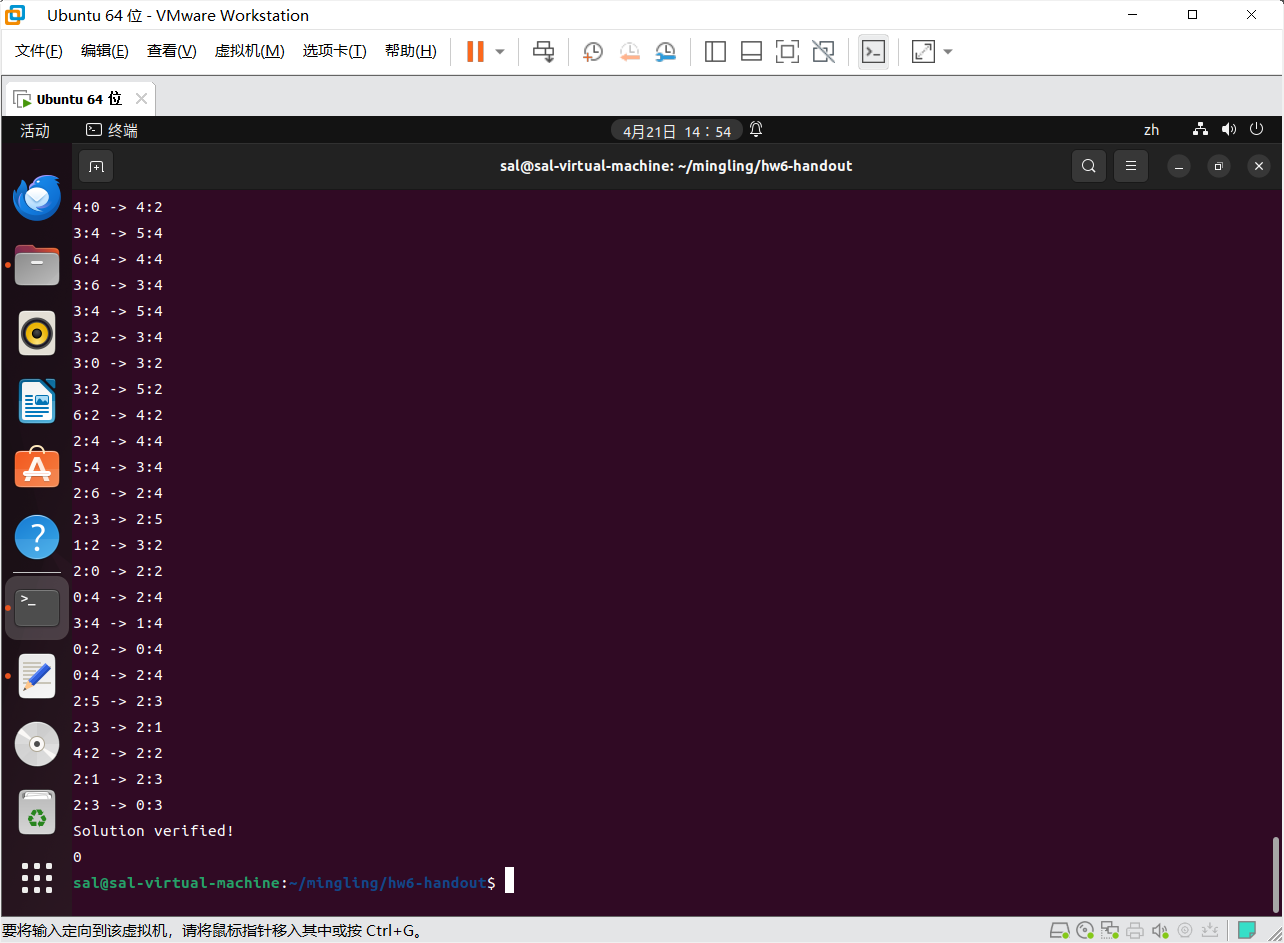
虽然正确解出答案，但是时间比较慢，经测试大概需要9s

## 3 附加题：记忆化

每次将棋盘状态都保存在hash表里面，每次构成新的棋盘就在hash表寻找，如果找到了就说明已经考虑过这个状态，就不需要再次考虑。

./peg3 english.txt





经过测试，不到1s就找到了解。