

信网应用层大作业项目报告

课程名称： 信息网络综合专题研究课

学生姓名： 白方

学 号： 19211226

班 级： 通信1904

队友姓名： 张绮璇

任课教师： 陈一帅，郭宇春

目录

[一、 选题 3](#_Toc18370)

[二、课题设计详情 3](#_Toc30382)

[2.1课题设计目的 3](#_Toc32478)

[2.2题目解决的问题 3](#_Toc23205)

[2.3需要完成的任务 3](#_Toc4284)

[三、本人完成任务及分析 4](#_Toc24292)

[3.1问题0 4](#_Toc32020)

[3.2问题1a 5](#_Toc15039)

[四、工作内容 6](#_Toc25211)

[4.1选择的内容 6](#_Toc3262)

[4.2本人完成的部分 6](#_Toc7805)

[4.3实现的功能 6](#_Toc6364)

[五、遇到的问题 6](#_Toc14170)

[5.1 Python编程基础学习的问题 6](#_Toc7615)

[5.2不知如何看懂代码，以及程序的编程结构 6](#_Toc9442)

[5.3 pycharm的使用的生疏及不懂程序文件、函数之间的调用的方法 6](#_Toc8369)

[5.4编写8皇后代码时的困难 6](#_Toc30985)

[六、总结与收获 7](#_Toc20068)

# 选题

本次选择的题目是斯坦福C221当中的解决课程安排的问题的编程作业——Course Scheduling。

# 二、课题设计详情

## 2.1课题设计目的

首先让选题人在问题 0 中熟悉 CSP 的基础知识。在问题 1 中，实践操作实现启发式算法，这将使 CSP 求解速度更快。最后，在问题 2 中，创建课程安排CSP 并使用前面部分的代码解决它。以此联系达到对于CSP求解问题的更深入了解，锻炼学者的实践操作能力。

## 2.2题目解决的问题

该作业的目的是编写一个程序，该程序会根据选课者的偏好和限制为其自动安排课程。该程序会将课程安排问题 (CSP) 转换为约束满足问题 (CSP)，然后使用回溯搜索来解决该 CSP，从而为选课者提供最佳课程安排。

## 2.3需要完成的任务

①问题0：现在让我们考虑一个一般情况：给定一个带n个变量的因子图x1…xn和n-1二元约束t1…tn-1，Xi={0,1}，ti为异或函数。通过创建以XOR为因子的通用链csp来实现create\_chain\_csp()函数。

②问题1a：让我们创建一个未加权的CSP来解决n皇后问题：给定一块板，我们希望在此板上放置皇后，这样就不会有两个皇后位于同一行、列或对角线上。通过添加变量和一些二进制因子来实现create\_nqueens\_csp()函数。

③问题1b：讨论的一个启发式方法是使用最受约束的变量（MCV）：要选择一个未分配的变量，请选择与当前部分赋值一致的值数目最少的（对于它，satisfies\_constraints()返回True）。在条件self.mcv = True下的get\_unassigned\_variable()中实现此启发式。您需要1361次操作才能找到8个皇后区CSP的所有最佳分配，这将减少30%！

④问题2a:实现函数add\_quarter\_constraints()。这是指您的个人资料指定选课者希望申请的课程参加的季度。这并不是说必须选修其中一门课程，但如果是，则必须在指定的任何一个季度选修。还请注意，此约束将应用于该请求中的所有课程。

# 三、本人完成任务及分析

## 3.1问题0

**3.1.1题目**

现在让我们考虑一个一般情况：给定一个带n个变量的因子图x1…xn和n-1二元约束t1…tn-1，Xi={0,1}，ti为异或函数为二元约束。通过创建以XOR为因子的通用链csp来实现create\_chain\_csp()函数。

**3.1.2分析**

问题0是为了让学者联系熟悉csp问题，需要解决的问题就是使用已经给出的代码，在submission.py文件里自己编写代码实现create\_chain\_csp()函数。该函数功能是创建一个CSP链表。考虑到创建一个初始化的CSP链表需要涉及到添加变量、为变量幅值等功能。在了解已经给出的涉及到的模块的代码的功能之后，发现创建一个CSP链表将会使用到Util.py文件当中的添加变量CSP.add\_variable()函数和给变量添加可取值CSP.add\_binary\_factor()函数。接下来就是对create\_chain\_csp()函数的功能实现的编程,先是创建variables含有n个变量的变量列表通过add\_variable函数对变量值域初始化，再通过使用Util.py模块中的CSP类当中的函数add\_binary\_factor将variables当中的每个变量通过一个二元约束条件XOR函数lambda x, y: x != y进行约束赋予相应的值实现简单的CSP求解。

**3.1.3代码实现**

# Problem 0

# Hint: Take a look at the CSP class and the CSP examples in util.py

def create\_chain\_csp(n):

# same domain for each variable

domain = [0, 1]

# name variables as x\_1, x\_2, ..., x\_n

variables = ['x%d'%i for i in range(1, n+1)]

csp = util.CSP()

# Problem 0c

# BEGIN\_YOUR\_CODE (our solution is 5 lines of code, but don't worry if you deviate from this)

for index,variable in enumerate(variables):

csp.add\_variable(variable,domain)

if index>0:

csp.add\_binary\_factor(variables[index-1],variables[index],lambda x, y: x != y)

# END\_YOUR\_CODE

return csp

## 3.2问题1a

**3.2.1题目**

创建一个未加权的CSP来解决n皇后问题：给定一块板，我们希望在此板上放置皇后，使得不会有两个皇后位于同一行、列或对角线上。在submission.py当中编程实现。create\_nqueens\_csp()函数。

**3.2.2分析**

该题与问题0有部分相似，都需要创建一个变量列表，但是还是有些不同。针对求解8皇后问题，我们将每个皇后的位置坐标（X0,X1）看作变量，初始化就是先创建有8个变量的通过添加变量和一些二元约束条件：x,y:x[1]!=y[1]、(x[0]-x[1])!=(y[0]-y[1])、(x[0]+x[1])!=(y[0]+y[1])分别表示任意两个皇后不在同一列，不在同一右对角线，不在同一左对角线，以及index!=index2表示两个皇后不在同一行，以此实现求解8皇后的CSP问题，从而实现create\_nqueens\_csp()函数。

**3.2.1代码实现**

def create\_nqueens\_csp(n = 8):

"""

Return an N-Queen problem on the board of size |n| \* |n|.

You should call csp.add\_variable() and csp.add\_binary\_factor().

@param n: number of queens, or the size of one dimension of the board.

@return csp: A CSP problem with correctly configured factor tables

such that it can be solved by a weighted CSP solver.

"""

csp = util.CSP()

# Problem 1a

# BEGIN\_YOUR\_CODE (our solution is 7 lines of code, but don't worry if you deviate from this)

variables=['x%d' % i for i in range(1,n+1)]

for index,variable in enumerate(variables):

csp.add\_variable(variable,[(index,i) for i in range(0,n)])

for index,variable in enumerate(variables):

for index2,variable2 in enumerate(variables):

if index!=index2:

csp.add\_binary\_factor(variable,variable2,lambda x,y:x[1]!=y[1])

csp.add\_binary\_factor(variable,variable2,lambda x,y:(x[0]-x[1])!=(y[0]-y[1]))

csp.add\_binary\_factor(variable,variable2,lambda x,y:(x[0]+x[1])!=(y[0]+y[1]))

# END\_YOUR\_CODE

return csp

# 四、工作内容

4.1选择的内容

本次内容由我和队友来分工完成，我们刚开始分工是由我来完成问题0和问题1，队友张绮璇完成问题2。

4.2本人完成的部分

本次任务我只完成了问题0和问题1a，队友则完成了问题2。

4.3实现的功能

对于自己负责的部分感觉逻辑上是没有问题的，但是运行时却显示不能成功运行，提示框显示的是题目给出的部分有问题，我们研究了很久但是由于时间有限，还有我们自身能力问题，很多给出的代码没法理解，最终还是没有运行出来。

# 五、遇到的问题

5.1 Python编程基础学习的问题

由于对Python编程语言并不是很熟，这次的大作业对于我来讲是一个很大的挑战，想要开始大作业首先就还得从基础学习，掌握基本的编程语言的逻辑及语法之后才能够读懂代码，才能够编写写代码。

由于时间非常有限，除去初始学习的时间，还得留出完成大作业的时间。所以自己并不能够全方位从基础的每个知识开始学，考虑到为了最基础的读懂代码和此次选题可能用到的知识点，我选择在b站和CSDN等各大网站查找资料，针对性学习所需的知识。在基本语法不说都能够完全掌握，但至少看到知道是什么，在哪见过能够查找到相关资料解决之后，才开始着手大作业。

5.2不知如何看懂代码，以及程序的编程结构

拿到大作业的题目，由于是英文，在翻译之后理解有些地方还是不是能够明白，在多次阅读理解知道需要完成的任务之后，就是需要对题目给出的代码进行阅读和理解，只有找到大体的逻辑结构和框架之后，在遇到问题才能够更有方向去寻找、解决问题。但是比较困难的地方也是这里，如果不能读懂代码，就没有办法入手编写代码。经过网上查找资料，通过别人分享的经验，学会在阅读时选中函数对象跳转查看的函数功能，经过阅读相关的程序，能够大概理解其编程的框架。

5.3 pycharm的使用的生疏及不懂程序文件、函数之间的调用的方法

针对Python类、面向对象的编程这方面理解不是非常清楚，在阅读代码的时候这是一个非常大的阻碍，在经过观看b站教学视频之后再阅读代码才没有那么吃力。同时在理解到这一点之后对于理解该编程的大体框架也有了很大的帮助。通过这段时间的琢磨大体能够知晓每个模块函数的作用，能够看出这个程序逻辑性很强，以及大体的结构。

5.4编写8皇后代码时的困难

对于8皇后的CSP问题编程自己做了很久，首先就是根据题目说给出一个8\*8的棋盘的解，在CSP问题当中首先面临的就是选择变量的问题，我想着可以选择这8\*8的每个棋盘为变量，就是64个变量，每个变量的初始值域就是0和1，但是在设置约束条件和利用约束条件进行CSP问题求解时，就非常繁杂，代码长度也超过了限制要求。再后面就想到更换变量对象，将每一个皇后看作一个变量，他们的坐标取值限制在8\*8的棋盘上，再根据约束条件进行CSP问题的求解。

5.5 GitHub提交代码问题

上B站查找了各种上传代码的视频，找到后发现需要在pycharm上下载插件，由于电脑一直加载不出来就没法继续下去，找到同学帮忙又因为其他原因导致不能够成功上传代码，只能够直接把代码交给老师了。

# 六、总结与收获

本次大作业让自己有机会和动力去更加投入学习Python这门编程语言，从自学到尝试完成大作业整个过程是很长的一个过程，也是一个循序渐进的过程，更是一个能力逐步提升的过程！突然自己喜欢上这门编程语言，它的灵活以及逻辑性让我很感兴趣，既然开了头我将继续学习下去。

再者是这次综合专题的学习真的收获了很多，学到很多很有意思的算法，通过老师的讲解以及同学们的展示，对这各种算法有更深的理解。针对上次的阅读展示而言，增加了自己自学的经验，通过展示给大家又增加了自己的表达方面的经验。有很多不足但是陈老师都耐心指出，很多时候别人的反馈对我们是至关重要的，是别人的意见让我们知道需要改进的地方。上上次通原研讨也是老师给了很多反馈的意见，自己才意识到需要针对性改进的地方，同样这次也是，非常感谢陈老师反馈意见！