中山大学

硕士学位论文

护士排班问题的算法与模型研究

姓名：刘曦

申请学位级别：硕士

专业：计算机软件与理论

指导教师：郭嵩山

20100601

论文题目：护士排班问题的算法与模型研究
专　　业：计算机软件与理论
硕 士 生：刘曦
指导教师：郭嵩山 教授

摘 要

护士排班问题是一个具有挑战性的题目，由于医疗机构和护理工作的特殊性，护士排班问题通常具有各种各样的约束条件。良好的排班方案有助于鼓舞团队的士气，营造良好的工作氛围，从而对护理质量提供有力的保证，进而保障病人的健康与安全，具有重大的实际意义。当前大部分医院的排班工作是以手工方式完成的，需要耗费排班人员较多的精力且难以保证排班的质量。基于计算机的自动化排班有助于提高排班的效率和质量，从而使得人力资源得到有效的利用。

本论文基于上述背景对护士排班问题展开了算法与模型的研究。首先，本论文对一个已有的护士排班问题提出了一个两阶段的求解算法，该算法的第一阶段使用分枝限界算法寻找可行的日夜分班方案，第二阶段使用模拟退火算法对每个可行方案进行优化。该算法的特点是，把问题转化到网络流模型上，通过求解最大流问题求得原问题的一个可行解，并以该解为起点，通过在残余网络上寻找回路来调整流量得到不同的最大流，以遍历原问题不同的解，在优化的过程中，本论文提出的基于调整最大流的邻域操作保证了搜索总在可行域内进行，提高了优化的效率。然后，本论文在原问题基础上加入了避免护士降级工作的约束条件，通过在原模型基础上增加常数个变量与不等式，提出了一个满足新约束条件要求的改进模型；同时，本论文对原有建模方法进行了分析，引入行约束条件与列约束条件的概念，在此基础上对原有建模方法进行了推广，得到了一个形式统一的通用模型，并应用到本问题中。最后，本论文通过对 30 组标准测试数据的实验验证了上述两阶段求解算法的有效性以及使用上述模型求解问题的效率。

关键词：护士排班、网络流、分枝限界、模拟退火、模型

Title: Research on Algorithm and Modeling of Nurse Scheduling Problem
Major: Computer Software and Theory
Name: Xi Liu
Supervisor: Prof. Songshan Guo

Abstract

Nurse scheduling problem is a challenging problem with various constraints, due to the specialness of hospital and nursing services. The quality of schedules has a significant impact on the morale of staff and the working environment, further on the health of patients. Most personnel scheduling problems in hospitals are solved manually currently, which turns out to be time consuming. Automatic approaches have great benefits in saving time and improving the quality of schedules.

Contributions of this thesis are twofold. First, a two-stage approach is proposed to solve a realistic nurse scheduling problem. Branch and bound is used to search for feasible partitioning of nurses in terms of day and night shifts in the first stage, and simulated annealing algorithm is used to improve the solution within each feasible partitioning found in the first stage. Network flow model is utilized to find an initial feasible solution, and flow adjusting based neighborhood, which guarantees feasibility after each move, is proposed to be used in the process of simulated annealing. Secondly, a new constraint on avoiding downgrading of nurses is introduced, and an improved model with constant number of additional variables and inequalities to the original model is proposed, moreover, by analyzing the original modeling method, concepts of row-based and column-based constraints are introduced and a generalized modeling method is developed, resulting in a problem-independent general model. Experiments with 30 real world data instances show the efficiency of the two-stage approach and solving problems by the proposed models.

Key Words: Nurse Scheduling, Network Flow, Branch and Bound, Simulated Annealing, Model

论文原创性声明

本人郑重声明：所呈交的学位论文，是本人在导师的指导下，独立进行研究工作所取得的成果。除文中已经注明引用的内容外，本论文不包含任何其他个人或集体已经发表或撰写过的作品成果。对本文的研究作出重要贡献的个人和集体，均已在文中以明确方式标明。本人完全意识到本声明的法律结果由本人承担。

学位论文作者签名：刘曦

日　　期：2010年6月1日

学位论文使用授权声明

本人完全了解中山大学有关保留、使用学位论文的规定，即：学校有权保留学位论文并向国家主管部门或其指定机构送交论文的电子版和纸质版，有权将学位论文用于非赢利目的的少量复制并允许论文进入学校图书馆、院系资料室被查阅，有权将学位论文的内容编入有关数据库进行检索，可以采用复印、缩印或其他方法保存学位论文。

学位论文作者签名：刘曦　　导师签名：郭雪飞

日期：2010年 6月 1日　　日期：2010年 6月 6日

第 1 章 绪论

本章首先介绍了护士排班问题的研究背景和实际意义，然后对相关文献进行了综述，接着概括了本论文的主要创新点，最后描述了本论文后续内容的组织结构。

1.1 问题背景和研究意义

排班问题已经被研究了数十年，吸引着来自管理科学、运筹学和计算机科学等相关学科的众多学者的注意。

在各种排班问题当中，护士排班问题是尤其具有挑战性的课题之一。不同于一般的机构，大部分医院需要全年全天候地运作，因此要求合理地安排不同班次之间的更替，充分考虑排班对护士的身体健康和作息规律的影响。另一方面，由于护士的护理服务质量直接关系到病人的身体健康与生命安全，因此在排班中应通过各种方式保证他们的服务质量。

护士排班问题有各种各样的约束条件，其中比较常见的约束条件包括以下 7 种类型：

1. 与法律法规相关的约束条件，如法定假期的安排，值班护士人数与床位数的比例，等等；
2. 医院本身制定的约束条件，如夜班之后必须安排足够的休息时间，尽量避免单独的工作天，尽量安排完整的周末，等等；
3. 与护士个人劳动合同相关的约束条件，如每周工作的总天数或总时长，是否专职服务于特定班次，等等；
4. 与护士个人偏好相关的约束条件，如在特定班次或者特定工作天的休假请求，对日班或者夜班的偏爱，等等；
5. 与护士的技能等级相关的约束条件，如特定班次需要具有特定技能的护

士值班，较高等级的护士可以替代较低等级的护士工作，等等；

6. 各种人性化的约束条件，如尽量平均地分配夜班等比较辛苦的工作；
7. 与部门相关的约束条件，如内科病房对护理工作需求比较稳定，因此排班要求比较固定，使护士可以熟悉病人的情况，也有利于增加病人对护士的亲切感，加强护患沟通，而对于急诊科等部门则要求排班要有一定的弹性，以应对忙闲不均的情况。

总的来说，一个好的排班方案首先要保证每个班次都有足够数量的医护人员值班，然后尽量提高护士们对该排班方案的满意程度。

当前，大部分医院的护士排班工作都是由专门的人员（如护士长）通过手工排班解决的。如果不是采用简单的轮班制，每次的排班工作都需要耗费相当多的时间和精力，而且由于负责排班的人员只能根据个人经验和直观的判断来安排工作，因此难以保证排班的合理性。

综上所述，护士排班是一个很有现实意义的问题。良好的排班方案有助于鼓舞团队的士气，营造良好的工作氛围，提高工作效率，从而对护理质量提供有力的保证，进而保障病人的健康与安全。而基于计算机的自动化排班将可以提高排班的效率和质量，从而使得人力资源得到有效的利用。

1.2 文献综述

早期的文献致力于提出相应的数学规划模型来求解护士排班问题[1][2]。

Warner 首先提出了与本论文所研究的问题模型类似的数学模型求解一个医院的实际问题[3]。该问题中要求为医院制定 4 周或者 6 周的工作安排，但实际处理中以每两周（14 天）为一个单位进行排班，每天有 3 种班次，护士分为两个等级。该文献提出先把每个护士可能的工作安排枚举出来，从而把问题建模为一个单项选择规划[4]的数学模型，并提出了一个两阶段的求解算法，该算法在第一个阶段致力于寻求可行解，在第二个阶段致力于降低目标函数值。

Burke 等人针对一个关于比利时医院的护士排班问题作了大量研究。首先，他们采用了多邻域搜索的方法求解该问题[5]，通过结合多个针对问题设计的邻

域，他们的方法能够在搜索过程中有效地跳出局部最优点，并访问到尽量多的搜索区域，取得了不错的效果。其后，他们采用一种混合的禁忌搜索算法来求解该问题[6]，在该算法中，他们把人工改进的方法结合到基本的禁忌搜索算法当中。在上述混合禁忌搜索算法的基础上，他们进一步结合 Memetic 算法来改进对该问题的求解结果[7]。

Bellanti 等人采用基于贪心的迭代式邻域搜索和禁忌搜索算法求解一个意大利医院中的实际问题，得到了不错的效果并且已经应用到实际当中[8]。

Burke 等人结合启发式排序和多邻域搜索算法求解护士排班问题[9]。该算法把所有待安排的班次按照难易程度排序，从难到易地把它们以贪心的原则分配给每个护士，从而求得一个初始解，然后使用多邻域搜索算法对其进行改进，最后采用破坏并修复的策略作进一步的改进。

运用人工智能的方法，Beddoe 等人采用基于案例推理（CBR）的方法来解决护士排班问题[10]。他们把以前被违反的约束和用于修复的操作记录下来以指导以后的排班决策。被违反的约束被记录成特征向量，而特征的提取以及每个特征对应的权值是通过遗传算法计算得到的。

Bard 等人把基于数学规划的启发式算法应用到护士排班问题中。他们利用列生成算法，在每次迭代中为每个护士生成可供选择的模式，并通过专业的软件求解相应的整数规划模型[11]。另外，他们运用拉格朗日启发式方法求解了一个循环班次安排问题[12]。同样利用列生成算法，他们针对允许护士降级工作的情况展开了研究[13]。

本论文所研究的问题最早由 Dowsland 等人提出[14]。他们把排班问题分为三个阶段：在第一个阶段中，通过求解一个有界的背包问题来判断是否有足够的护士满足当前的需求；在第二个阶段中，采用禁忌搜索算法求解主要的排班问题[15]；在第三个阶段中，通过网络流模型把护士最终分配到特定的班次中。其中，第二阶段的排班问题是本论文研究的问题。该禁忌搜索算法的主要特点是：在搜索过程中，有策略地在可行域与非可行域之间来回移动，使得每进入一个可行域，都尽量探索其中的局部最优点，然后通过摆动到非可行域来跳出局部最优，如此不断循环。为了改进搜索效果，该算法采用了链式移动的邻域结构来替代简单的单步移动的邻域结构。

Aickelin 等人采用遗传算法求解本问题[16]，其中每个个体直接编码了一个具体的排班方案。他们指出，遗传算法能否在一个问题的求解中取得效果，很大程度上取决于该问题的解是否具有块状性（Building block），即好的解的基因里面有连续的一段优秀的基因。然而对于排班类的问题，好的解往往不具有块状性，直接使用遗传算法通常不能得到好的结果，在很多情况下甚至无法找到可行解。该文献于是针对问题特性提出按照护士的等级把整个种群分为若干个子群，然后以一定的策略让子群共同进化。

Aickelin 等人也提出了间接的遗传算法[17]，即每个个体编码的不是一个排班方案，而是待安排的护士的一个排列，解码的时候，以这个排列为基础，利用启发式的方法依次为每个护士安排班次，从而构造出具体的排班方案，并以此评价个体的优劣。根据侧重点的不同，不同的启发式方法被提出和测试，实验结果表明，侧重于对覆盖率的满足的启发式效果最佳。

基于上述两个遗传算法，Aickelin 等人提出了一种比较方法，在可能找不到可行解的情况下，比较不同的算法之间的优劣[18]。

Li 等人提出了一种基于分量的模仿演化算法性质的启发式搜索算法[19]。该算法从一个随机的初始解开始，进行若干次迭代，每次迭代都是一个破坏后修复的过程。具体来说，在每次迭代中，算法对每个护士分别计算一个分数，表示他对当前解的影响程度。影响包括两个方面。第一个方面是该护士在当前解中被指派的模式的权值，这是该护士对目标函数的直接贡献；第二个方面是该护士当前被指派的模式对当前需求的满足性的影响，主要体现在该护士当前被指派的模式所覆盖的班次中是否有冗余的覆盖。直观来看，分值越高的护士的当前指派更合理。该算法模仿遗传算法的优胜劣汰原则，根据分数的高低，按照一定的概率把分值较低的护士的安排从当前解中清除掉，接着模仿遗传算法中的变异原则，以一个较小的概率完全随机地清除掉某些护士的班次安排，最后对于这些需要被重新安排的护士，利用启发式的方法来为他们指派新的模式。

Li 等人利用贝叶斯优化和分类系统来求解这个问题[20]。Burke 等人采用了超启发式方法来求解该问题[21]，即利用禁忌搜索算法来选择使用哪个低层次的启发式算法。Aickelin 等人提出使用分布估算算法求解该问题[22][23]。

Goodman 等人采用随机化的贪心搜索算法求解此问题[24]。该算法进行多次

迭代，在每次迭代中先以随机化贪心的方式构造初始解，然后以该初始解为起点进行局部搜索来改进解的质量。在贪心构造初始解的过程中，该算法结合了Dowsland 等人提出的方法[14]，通过求解一个有界的背包问题来判断当前的贪心决策是否会导致最终无法得到可行解。

Burke 等人和 Cheang 等人分别对护士排班问题作了详细的综述[25][26]。

1.3 本论文的主要创新点

首先，本论文对一个已有的护士排班问题提出了一个两阶段的求解算法。该算法的第一个阶段通过分枝限界算法寻找可行的日夜分班方案，第二个阶段对每个可行的日夜分班方案通过模拟退火算法进行优化。本算法的特点是，把问题转化到网络流模型上，通过求解最大流问题求得原问题的一个可行解，然后以该最大流为起点使用模拟退火算法对其进行优化，通过在残余网络上寻找回路来调整流量得到不同的最大流，以遍历原问题不同的解。在优化的过程中，本论文提出的基于调整最大流的邻域操作保证了搜索总在可行域内进行，提高了优化的效率。在寻找回路时，本论文在一般的宽度优先搜索和深度优先搜索框架上引入了随机因素以加强优化的效果。

然后，本论文在原问题上加入了要求避免护士降级工作的约束条件，通过在原有模型上增加常数个变量与不等式，本论文提出了一个满足新约束条件要求的改进模型。同时，本论文对原有的建模方法进行了分析，引入了行约束条件与列约束条件的概念，指出原有的建模方法实质上是把排班表中每行可能的取值事先枚举出来，并通过预处理把所有行约束条件转化为一个权值附加在每个行模式上，从而简化了最终的数学模型，然而当列约束条件比较复杂时，使用原有的建模方法可能会得到比较复杂的数学模型。于是本论文对原有建模方法进行推广，提出同时把列模式枚举出来并且预处理所有的列约束条件，然后引入一组新的等式来保证行列模式选取的一致性，使得最终得到的数学模型是一个形式统一的单项选择规划模型，而不依赖于问题的约束条件。本论文进一步地引入了超级行、超级列的概念，指出可以根据具体问题的特点把无区别的行或者列合并为超级行

或者超级列来处理，从而减小模型的规模，更有利于求解。

1.4 后续内容的组织结构

第2章具体地描述了本论文所研究的护士排班问题及其数学模型，第3章描述了本文提出的两阶段求解算法并对算法的每个组成部分展开了讨论，第4章在原问题基础上引入了新的约束条件并对其展开了模型的研究，第5章列出了本论文的实验结果并对其进行了分析，第6章总结全文并指出了今后研究的方向。

第 2 章 问题描述及数学模型

本章具体地介绍了本论文所研究的护士排班问题，讨论了它的建模过程并给出了它的数学模型，为后面的内容作了铺垫。

2.1 问题描述

本问题有如下5个要素：

1. 要求为护士们安排一周的工作；
2. 每天有三种班次，其中两种是日班，分别记为“earlies”和“lates”，另外是一个时间比较长的夜班（后面将介绍实际只需分为日夜班两种）；
3. 护士分为3个等级，其中等级1最高，等级2次之，等级3最低；
4. 每天的每个班次对每个等级的护士人数有一定的需求；
5. 高等级的护士可以替代低等级的护士工作。

如Dowsland等人在文献中所述，该问题可以分为3个阶段处理[14]：

1. 通过求解一个有界的背包问题判断是否有足够的护士满足本周的需求，如果不够则需要聘用“流动护士”；
2. 为每个护士安排每天的工作，即日班/夜班/休假；
3. 通过一个网络流模型把每天安排到日班的护士具体分配到earlies和lates对应的班次上。

在上述阶段1中使用到的模型如模型2-1所示。

模型2-1 判断是否有足够护士的模型[14]

$$\sum_{i=1}^{T} e_i y_i \geq E \tag{2-1}$$

$$\sum_{i=1}^{T} d_i (N_i - y_i) \geq D \tag{2-2}$$

$$\sum_{i=1}^{T} y_i \leq N_i \tag{2-3}$$

$$y_i \text{ is integer} \tag{2-4}$$

在模型 2-1 中，

- T 表示护士的种类数；
- N_i 表示第 i 类护士的总人数；
- d_i 和 e_i 是第 i 类护士的工作天数，表示第 i 类护士每周要么工作 d_i 个日班，要么工作 e_i 个夜班；
- D 是一周内日班需求的总数；
- E 是一周内夜班需求的总数；
- y_i 是决策变量，表示第 i 类护士中被分配到夜班的人数；
- 式（2-1）要求总共工作的夜班数之和大于等于夜班需求的总数 E；
- 式（2-2）要求总共工作的日班数之和大于等于日班需求的总数 D；
- 式（2-3）和（2-4）约束了 y_i 的取值范围。

如果把式（2-2）改写为式（2-5）的等价形式，

$$\sum_{i=1}^{T} d_i y_i \le \sum_{i=1}^{T} N_i d_i - D \tag{2-5}$$

那么，由式（2-3）（2-4）（2-5）所构成的模型就是一个标准的有界背包问题（Bounded Knapsack Problem），该问题已经有成熟的解法。如果该有界背包问题的解（即 $\sum_{i=1}^{T} e_i y_i$ 的值）小于 E，那么原问题肯定无可行解。但文献里说“如果该有界背包问题的解大于或等于 E，那么原问题有可行解” [14]，本论文认为这种说法是有漏洞的。从模型的描述可以看出，该模型以护士能工作的总数是否大于或等于需求的总数来判断原问题是否有可行解，相当于把二维（一维是班次，一维是需求）的问题压缩为一维（把班次的维度压缩了）来判断，因此这是一个必要而非充分的判定条件。

命题 2-1：模型 2-1 判断原问题是否有可行解的条件是必要非充分性的。

证明 2-1：必要性是显然的，因为原问题的任意一个可行解都可以直观地构造出模型 2-1 的一个解，并且满足该解的值大于或等于 E。

下面证明非充分性。考虑只有两个日班、两个护士的简单情况。这两个护士都属于同一个类型，即每周工作两个日班。第一个日班需要三个护士，第二个日

班需要一个护士。这两个护士能工作的日班总数为2+2=4，日班需求的总数为3+1=4，即护士能工作的总数大于等于需求的总数，于是利用模型 2-1 的方法会判断为有可行解，而事实上这个例子是没有可行解的。因此使用模型 2-1 判断原问题是否有可行解是必要非充分的。

证毕。

尽管上面描述的把二维压缩到一维的判断方法是必要非充分的，但是由于一般情况下每天的需求都是比较平均的，很少出现证明 2-1 中的那种有峰值存在的情形，因此使用这种方法判断原问题是否有可行解的实际效果应该是很好的，本论文的实验结果证实了这一观点，因为本论文在第 3 章所描述的分枝限界算法中也采用了类似的方法进行剪枝。

对于上述的阶段 3，即使用网络流模型把日班的护士分配到 earlies 和 lates 对应的班次上，由于网络流问题已经有各种成熟的多项式算法，于是该阶段所涉及的问题不在本论文的讨论范围之中。

本论文的研究都是针对上述的阶段 2 的，即决定每个护士每天的工作安排（日班/夜班/休假），下面将介绍对这个阶段的问题的建模方法。

如上所述，该问题是要为护士们安排一周的工作，而每天的安排有三种可能，即日班、夜班和休假，因此每个护士一周内可能的工作安排不超过$3^7=2187$种。进一步来说，对于每个护士，假设他属于第i类，那么他每周要么上d_i个日班，要么上e_i个夜班。因此，每个护士一周内可能的工作安排不超过$2\times\binom{7}{3}=70$种。例如，对于全职护士来说，他们每周要么工作 5 个日班，要么工作 4 个夜班，因此他们可能的工作安排就只有$\binom{7}{5}+\binom{7}{4}=56$种。

定义 2-1：模式（Pattern）

每种工作安排称为一个模式。在本问题中，用一个 14 位的二进制串表示，其中前 7 位表示 7 个日班班次，后 7 位表示 7 个夜班班次，对应位置为 1 表示被安排了该班次。

考虑到每个护士可能的模式数量并不多，于是可以事先把它们枚举出来，并根据各种约束条件为每个模式分别计算一个权值 p，以表示该模式（对于该护士）的好坏，或者说该护士对该模式的喜好/厌恶程度。具体来说，这个权值是根据

以下 3 个方面计算得到的：

1. 该模式本身的特点，这是模式的内在属性，可以理解为所有人对这个模式的一致的看法。例如，有单独工作天（即前一天和后一天都是休假的工作天）的模式一般来说不太受欢迎。
2. 护士的个人请求，这是因人而异的。例如，某个护士希望在本周五休假，那么对于他来说，所有对周五安排了工作的模式都是不太好的。
3. 最近的工作安排历史，这是纵向的考虑。例如，某个护士在上周或者连续的前几周都被安排了夜班，那么本周的日班模式可能更适合他；又比如说某个护士在上周的最后一天上的是夜班，那么在本周的第一天安排了日班的模式都是应该禁止的（设置为较大的权值）。

经过上述的处理，原护士排班问题被简化为如下 3 个要求：

1. 对于每个护士，从他所有可能的模式中选择一个且仅一个；
2. 选择的模式要保证每个班次都有足够的护士人数；
3. 使被选择的模式的权值之和最小。

上面描述的过程即为本问题的建模过程，其特点是把对护士人数的需求以外的所有约束条件都通过预处理转化到了一个权值中，从而简化了最后得到的数学模型。具体的数学模型将在 2.2 节中介绍。

2.2 数学模型

模型 2-2 本问题的数学模型[14]

min
$$z=\sum_{i\in N}\sum_{j\in F(i)} p_{ij}x_{ij} \tag{2-6}$$

subject to
$$\sum_{j\in F(i)} x_{ij}=1,\forall i\in N \tag{2-7}$$
$$\sum_{i\in N_r}\sum_{j\in F(i)} x_{ij}a_{jk}\geq R_{rk},\forall r\in[1,3],k\in[1,14] \tag{2-8}$$
$$x_{ij}\in\{0,1\},\forall i\in N,j\in F(i) \tag{2-9}$$

在模型2-2中，

- i是护士的索引；
- j是模式的索引；
- k是班次的索引；
- r是护士等级的索引；
- N表示所有护士的集合；
- N_r表示所有等级r或以上的护士的集合；
- $F(i)$表示护士i的可选模式的集合；
- p_{ij}表示模式j对于护士i的权值；
- a_{jk}表示模式j是否覆盖班次k，为1表示覆盖，否则为0；
- R_{rk}表示班次k需要等级r或以上的护士的数量，由于允许高等级的护士替代低等级的护士工作，因此这里采用累计的表示方式；
- x_{ij}是决策变量，为1表示护士i被指派了模式j，否则为0；
- 式（2-6）是要求最小化的目标函数，即所有被选择的模式的权值之和；
- 式（2-7）保证了在每个护士的可选模式集合中选择其中的一个且仅一个模式；
- 式（2-8）保证了每个班次对不同等级的护士数量的需求被满足；
- 式（2-9）是对变量取值范围的约束，即取值为0或者1。

该模型是一个整数线性规划（Integer Linear Programming）模型，模型中涉及大概2000个变量和70个约束条件。进一步来说，该模型可以被看作一个单项选择规划（Multiple Choice Programming）的集合覆盖（Set Covering）问题，其中式（2-7）体现了Multiple Choice的性质，式（2-8）体现了Set Covering的性质。

这种建模方法在护士排班问题中是比较常见的，它通过预处理把各种复杂的约束条件转化为一个简单的权值附加在模式中，以增加变量个数的代价简化了最终的模型。早在1976年，Warner就首先以类似的方法对一个护士排班问题进行了建模并求解[3]。

在第4章中将继续讨论这个模型和建模方法，并且以此为基础提出新的模型以求解带有新的约束条件的问题。

第 3 章 两阶段的求解算法

本章介绍了本论文所完成的第一部分主要工作，即针对第 2 章所描述的问题提出的一个两阶段的求解算法。3.1 节首先介绍了本算法的设计思路并描述了算法的整体框架，后续部分分别讨论了算法的每个组成要素。

3.1 算法的整体描述

如 2.1 节所述，一周内一个护士能工作的日班数一般不等于他能工作的夜班数，因此如果以总班次数来衡量护士的工作量，那么每个护士一周内的工作量是不确定的（有两种可能），这将给问题的求解带来困难。

针对上述困难，本论文提出的解决思路是，通过某种手段先让每个护士的总工作量确定下来，即首先确定护士的日夜分班方案。

日夜分班方案的确定将为问题的求解带来如下方便：

1. 可以把原问题转换到一个网络流的模型中，通过求解该网络上的最大流问题来判断原问题是否有可行解，并且求出其中一个可行解，这个过程可以在多项式时间内完成，而且有多种成熟且高效的算法；
2. 在上述网络流模型中，每个最大流都可以对应到原问题的一个可行解，因此可以把搜索原问题的解的过程转化为在该网络上遍历不同的最大流的过程；
3. 通过在残余网络上寻找回路并沿着该回路进行流量调整可以把网络上的一个最大流变成另外一个最大流，而寻找回路的过程可以在与该网络的边数成线性关系的时间内完成，这提供了一种只在可行域中跳转的邻域搜索方案，有助于高效地对解空间进行遍历以寻找最优点。

当然，对于每一个具体的问题来说，可行的日夜分班方案可以有很多种，而其中的哪一种分班方案可以导致全局的最优解是难以被事先确定的，因此，必须

找到这些可行的日夜分班方案并分别对其进行优化。

综上所述，本论文提出了一个两阶段的求解算法：

Stage 1. 找出可行的日夜分班方案；

Stage 2. 对每个可行的分班方案进行优化。

这两个阶段可以以不同的方式组织起来。例如，以完全分离的方式组织，即首先把所有可行的日夜分班方案都找出来，然后有选择地对其中的某些方案进行优化。这样做的好处是，由于已经得到了所有的可行方案，因此可以先按照一定的准则对这些方案进行筛选，从而减少进行优化的次数，更进一步来说，还可以根据对这些方案的不同估价而分配给他们不同的优化策略。

然而对于本问题而言，对于每一个日夜分班方案，在对其进行优化之前缺乏有效的手段去评估该方案的优劣，因此，本论文以交替进行的方式把这两个阶段组织起来，即每当找到一个（或一些）可行的日夜分班方案后，便对这个（或这些中的某一个）分班方案进行优化。这样做的好处是，经过对若干分班方案的优化以后，可以在算法的进行过程中得到越来越紧的上界（即更接近最优解的当前最优解），从而有助于把明显不可能导致最优解的分班方案排除在外，减少对其进行不必要的优化过程。

具体而言，本论文在第一阶段采用分枝限界算法搜索可行的日夜分班方案，每当找到一个可行的方案便对其进行优化，在搜索的过程中通过剪枝函数把不可能导致全局最优解的分班方案剔除。

对于第二阶段的优化方法的选择，主要有以下的考虑。由于本阶段的算法可能被调用多次，因此要求它有比较高的执行效率。出于效率的考虑，最基本的局部搜索算法是一个不错的选择。然而，局部搜索算法的效果在较大程度上依赖于初始解的选取，但是利用最大流算法求出的初始可行解很难保证其质量。从另外一个角度来说，由于每个日夜分班方案只会被优化一次，而且显然任意两个日夜分班方案所确定的解空间是没有交集的，即在本次优化过程中被遗漏的解在以后也不可能再被访问到，因此，要求所使用的优化方法能在保证一定效率的前提下，尽可能多地在该日夜分班方案所确定的解空间内进行探寻。

具体而言，本论文在算法的第二阶段中采用了模拟退火算法对每个日夜分班方案进行优化。

综合以上的讨论，本论文提出的两阶段求解算法的整体框架如图 3-1 所示。

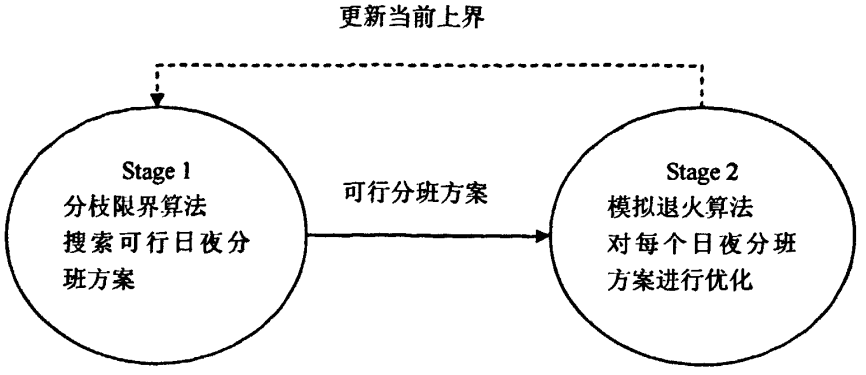

图 3-1 两阶段求解算法的整体框架

从图 3-1 可以看出，两个阶段的算法之间是相辅相承的。第一阶段的分枝限界算法的有效剪枝可以减少第二阶段的模拟退火算法被调用的次数，从而允许每次优化有更充足的搜索时间；反过来说，第二阶段的模拟退火算法对每个日夜分班方案进行充分的优化可以为第一阶段的分枝限界算法提供更紧的上界，从而有助于发挥剪枝的作用。

每个阶段的算法的具体细节将在本章的后续部分讨论。

3.2 使用分枝限界算法搜索可行日夜分班方案

本节介绍了分枝限界算法并对其中用到的策略进行了讨论。

3.2.1 分枝限界算法

分枝限界算法是一个通用的搜索框架。对于要求最小化的问题而言，它的主要思想是，把已经找到的当前最优解作为上界，对于搜索树上的每个节点，通过对问题进行松弛（Relaxation）求出它的一个下界，如果节点的下界大于或者等于当前上界那么可以对以该节点为根的子树进行剪枝，从而提高搜索的效率。

分枝限界算法已经被广泛应用在多种最优化问题的求解中，它的搜索效果主

要跟以下两个因素有关：

1. 分枝策略
2. 剪枝策略

3.2.2 节将对分枝策略进行讨论，3.2.3 节将讨论剪枝策略。

3.2.2 分枝策略

分枝限界算法的分枝策略，是指在搜索的过程中以怎样的准则在当前搜索树上选择下一个用于扩展的节点以及扩展的方法。

分枝策略对搜索的整体效率有很大的影响。直观而言，如果优先执行那些更有可能导致最优解的分枝，那么将更快找到好的解，从而对后续搜索过程中的剪枝有很大的帮助。最理想的情况是，每次分枝都向着最优解的方向前进，那么搜索的效果将是最好的。

常用的分枝策略有以下两种：

1. 每次选择一个最新生成的节点，即采用深度优先的搜索方式。该策略的优点是节省内存空间（只需要维护一个大小与搜索树深度成正比的栈空间），而且每次执行分枝操作的效率高，适用于需要搜索所有解的情况。
2. 每次选择当前下界最好的节点，即采用最优值优先的搜索方式。该策略往往能更快地找到最优解，但缺点是需要占用比较多的空间（需要维护一个大小与搜索树节点数成正比的优先队列），适用于寻找最优解的情况。特别地，如果在搜索过程中的每一步都是单位费用，该策略等价于广度优先的搜索方式。

本论文采用基于深度优先的搜索方式，原因主要包括以下两点：

1. 本阶段算法的目的是要找出所有有希望导致全局最优解的可行日夜分班方案；
2. 在当前的搜索中，每个节点的下界难以准确地估算（因为每个节点的最优值必须通过第二阶段的模拟退火算法来探索）。

对于深度优先的搜索方式，它的实际搜索效率又跟以下两个因素有关：

1. 待搜索的节点的顺序，具体到本问题而言，就是待安排日夜分班的护士的顺序；

2. 对一个节点的扩展顺序，具体到本问题而言，就是对一个护士优先尝试安排日班还是夜班。

由于在本文的算法中最耗时的是第二阶段的优化过程，而且该过程需要被多次调用，因此希望在第一阶段中利用最优性剪枝尽量减少第二阶段算法被调用的次数。基于以上考虑，本文采取启发式的搜索策略，优先尝试把那些对日班（或夜班）有偏爱的护士分配到日班（或夜班）。

对于这种搜索策略，直观的理解是，如果尽量优先满足护士对日班或夜班的偏爱，那么将更有可能尽快地找到比较好的可行解，从而在更短的时间内得到一个更紧的上界，有利于对后续搜索的剪枝。

具体而言，算法首先根据式（3-1）对每个护士 i 计算一个分数 $nightPref(i)$，其中 $NP(i)$ 是护士 i 可选的夜班模式的集合，接着按照 $nightPref(i)$ 的值从小到大的顺序对护士进行排序，然后以此顺序为基础进行深度优先搜索，对于搜索到的每个护士，优先尝试把他安排到夜班。

$$nightPref(i)=\frac{\sum_{j\in NP(i)} p_{ij}}{|NP(i)|} \tag{3-1}$$

本小节讨论了分枝限界算法中所使用的分枝策略，下一小节将讨论关于剪枝的策略。

3.2.3 剪枝策略

根据针对性的不同，剪枝策略主要有两类：

1. 可行性剪枝
2. 最优性剪枝

可行性剪枝是指在搜索树的每个节点处，判断以该节点为根的子树中是否可能有可行解，如果该子树中不可能有可行解，那么该子树可以被剪掉。

最优性剪枝是指在搜索树的每个节点处，判断以该节点为根的子树中是否可能有优于当前最优解的解。对于要求最小化的问题，搜索过程中每找到一个解，就可以用它来更新当前的上界，而对于搜索树上的每个非叶子节点通过对问题进行松弛估算出一个下界，如果该下界大于或者等于当前的上界，那么这个节点所对应的整棵子树中就不可能有优于当前最优解的解存在，因此对其进行搜索是没

有意义的，于是该子树可以被剪掉。

剪枝策略的设计是一个分枝限界算法是否有效的关键因素，因为如果不能有效地进行剪枝，那么分枝限界算法将退化为朴素的枚举法，也就失去了使用它的意义。

为了方便描述，本节把搜索过程中的每个节点记为一个三元组（P，Q，R），其中：

- ✧ P 表示当前已经被分配到日班的护士的集合
- ✧ Q 表示当前已经被分配到夜班的护士的集合
- ✧ R 表示当前还没确定日夜分班的护士的集合

由于护士上日班的工作天数和上夜班的工作天数一般来说是不一样的，因此对于节点（P，Q，R）来说，对其可行性的估算的困难在于集合 R 中的护士。本论文采用的方法是，假设把集合 R 中的护士都安排到日班（或夜班），然后判断日班（或夜班）的需求是否能被满足。显然，如果在集合 R 中的护士都被安排到日班（或夜班）的情况下，日班（或夜班）的需求都不能被满足，那么以节点（P，Q，R）为根的子树中肯定不可能有可行解，于是可以对其进行剪枝。

对于日班（或夜班）的需求能否被一个集合 A 中的护士满足的问题，本论文研究了两种不同的判断方法，它们各有优劣。

第一种判断方法记为 Greedy Check，它类似于 2.1 节所描述的利用模型 2-1 判断原问题是否有可行解的方法，把二维的班次需求压缩到一维（即把所有班次的需求值累加起来），然后根据集合 A 中的护士工作的总天数是否大于或者等于班次需求的总数来判断需求是否能够被满足。

命题 3-1：Greedy Check 的判断条件是必要非充分的。

证明 3-1：同证明 2-1。

Greedy Check 的优点是执行效率高，缺点是可能会出现误判的情况。由于它的判断条件是非充分的，因此 Greedy Check 可能会把没有可行解的情况判断为有解，从而错过了剪枝的机会。但是正如 2.1 节所述，由于在一般情况下，每天的班次需求都是比较平均的，很少出现如证明 2-1 中所描述的有峰值的情形，因此 Greedy Check 的实际效果是很好的。

第二种判断方法记为 Flow Check，它使用网络流模型判断需求是否能被满

足。关于该网络流模型及其判断方法，将在3.3节具体介绍。在此先不加证明地给出命题3-2。

命题3-2：Flow Check的判断条件是充分必要的。

证明3-2：见3.3节。

Flow Check的优点是它具有充分性，即只要它判断的结果是肯定的，那么需求一定能被满足，但是它的缺点是执行效率比较低，因为需要构造网络并执行求解最大流的算法。

实验结果表明，Greedy Check的剪枝效果和Flow Check是一致的，而且效率更高，因此本论文采用了Greedy Check作为可行性剪枝策略。

关于上述的可行性剪枝，在程序的实现中可以进行优化，尽量减少Greedy Check被调用的次数。具体而言，每当搜索到一个节点时，不需要分别对日班和夜班的需求的可满足性进行检查。如果上一个护士被安排在日班，那么对于本节点而言只需要检查夜班的需求能否被满足，因为日班的需求肯定能被满足；同理，如果上一个护士被安排在夜班，那么只需要检查日班的需求能否被满足。这是因为，既然搜索能进行到当前节点，表明之前执行的Greedy Check都是成功的，而且当时上一个护士还处于没确定日夜分班的状态，那么当上一个护士被确定安排到日班（或夜班）后，再检查日班（或夜班）需求的可满足性是没有意义的。

进一步研究发现， Greedy Check被调用的次数还可以进一步减少。在搜索的过程中记录如下两个量：

1. alreadyOkDay——为1表示当前已经被安排到日班的护士已经足够满足日班的需求，否则为0；
2. alreadyOkNight——为1表示当前已经被安排到夜班的护士已经足够满足夜班的需求，否则为0。

显然，只有在alreadyOkDay为0的情况下才有需要检查日班的可满足性，同理只有在alreadyOkNight为0的情况下才有需要检查夜班的可满足性。这两个量可以在每次执行Greedy Check的过程中被更新。

以上讨论了可行性剪枝策略及其实现的优化，接着讨论最优性剪枝。

本论文采用一个非常具有贪心性质的最优性剪枝，记为Greedy Pruning。它的主要思想是，假设每个护士都可以被安排到他最喜欢的模式，即每个护士对目

标函数值都只贡献了他的可选模式集中满足日夜分班方案的权值最小的模式的值。

具体而言，对于集合 P 中的每个护士，假设都分别把他们安排到各自最喜欢的日班模式；对于集合 Q 中的每个护士，假设都分别把他们安排到各自最喜欢的夜班模式；对于集合 Q 中的每个护士，假设都分别把他们安排到各自最喜欢的任意一个模式。

形式化地描述为，对于每个护士 i，记录如下 3 个量：

1. $minPref_i$——护士 i 的可选模式集中权值最小的模式的值；
2. $minPrefDay_i$——护士 i 的可选模式集中权值最小的日班模式的值；
3. $minPrefNight_i$——护士 i 的可选模式集中权值最小的夜班模式的值。

则节点（P，Q，R）的下界 LB 根据式（3-2）计算得到。

$$LB(P,Q,R)=\sum_{i\in P} minPrefDay_i+\sum_{i\in Q} minPrefNight_i+\sum_{i\in R} minPref_i \qquad (3\text{-}2)$$

如果节点（P，Q，R）的下界大于或者等于当前已经找到的最优解的值，那么可以对该节点进行剪枝。

本节所讨论的分枝限界算法的伪代码如算法 3-1 所示。

```
Branch and Bound Algorithm
Search(P, Q, R)
  if LB(P, Q, R) >= best_s
    return
  if R == ∅
    s := SA(P) + SA(Q)   // Improve Day and Night respectively by
                         // Simulated Annealing algorithm
    if s < best_s
      best_s := s
    return
  let x be the next nurse in R
  if Greedy Check(P ∪ (R - {x}))
    Search(P, Q ∪ {x}, R - {x})
  if Greedy Check(Q ∪ (R - {x}))
    Search(P ∪ {x}, Q, R – {x})
```

算法 3-1 分枝限界算法的伪代码

3.3 转化到网络流模型

本节介绍了网络流模型以及一个重要的相关概念——残余网络，并介绍了把原问题转化到网络流模型上的方法。

3.3.1 网络流模型

本节介绍网络流模型相关的概念。

定义 3-1： 网络（Network）

弧上带有非负权值的有向图称为一个网络 G。记它的顶点集合为 V，弧集合为 E。弧 $<a,b>$ 上的权值称为该弧的容量，记为 c_{ab}。网络上有两个特殊的顶点，

一个称为源点，记为S，另一个称为汇点，记为T。

定义3-2：流（Flow）

如果一个实值函数f满足式（3-3）和式（3-4），那么它称为网络G上的一个流。

$$f(i,j)\le c_{ij},\forall i,j\in V \tag{3-3}$$

$$\sum_{i\in V} f(i,u)=\sum_{i\in V} f(u,i),\forall u\neq S\wedge u\neq T \tag{3-4}$$

其中，

- 式（3-3）称为流量约束条件，它要求每条弧上的流量不能超过它的容量；
- 式（3-4）称为流量守恒条件，它要求除了源点和汇点以外，每个顶点的总流入量等于总流出量；
- $\sum_{i\in V} f(S,i)$或者$-\sum_{i\in V} f(i,T)$称为流f的值。

定义3-3：最大流问题（Maxflow Problem）

最大流问题是指对于给定的网络G，求该网络上的一个具有最大流量值的流的问题。

定义3-4：残余网络（Residual Network）

对于网络G上的一个流f，它在该网络上对应的残余网络R被定义如下：

- R具有和G一样的顶点集合；
- 对于G中的每条弧$<a,b>$，如果$f(a,b)>0$，则在R中添加容量为$f(a,b)$的弧$<b,a>$，如果$f(a,b)<c_{ab}$，则在R中添加容量为$c_{ab}-f(a,b)$的弧$<a,b>$。

顾名思义，残余网络也是一个网络，而且它是由流f在网络G上导出的网络，因此它跟f有紧密的联系。直观上来看，残余网络里的弧表示流f可以调整的流向，弧的容量表示流的可调整量。

定义3-5：增广路径（Augmenting Path）

残余网络R上的从源点S到汇点T的路径称为增广路径。

顾名思义，沿着增广路径调整流量可以使总的流量增加。不断地寻找增广路径并沿着它使流量增大的方法是一个求解最大流问题的通用途径。

显然，如果当前流f已经是网络G上的一个最大流，那么f对应的残余网络R上不可能存在增广路径，否则，沿着该增广路径调整流量还可以使当前流量增大，这与当前流是最大流的前提是矛盾的。

命题3-3：在网络G上的一个最大流f对应的残余网络R上沿着任意一个回路调整流量得到的流f'仍然是该网络上的一个最大流。

证明3-3：由于流量的调整是沿着一个回路进行的，因此该回路上的每个顶点的总流入量和总流出量在调整前后是不变的，不在该回路上的顶点的总流入量和总流出量当然也不变，因此在流f和流f'中，源点S的总流出量是一样的，即流f'的值等于流f的值，因此f'仍然是该网络上的一个最大流。证毕。

3.3.2 网络的构造与求解方法

本小节讨论把原问题转化到网络流模型上的方法。

显然，在一个特定的日夜分班方案中，被分配到日班的护士只能为日班的各个班次服务，同理被分配到夜班的护士只能为夜班的各个班次服务。因此，原问题可以分解为两个独立的且性质相同的子问题分别求解。

本节只针对日班的子问题展开讨论，对于夜班的情况是类似的。

在讨论网络的构造方法之前，本节首先给出分离需求的概念。在原问题的描述中，每个班次的需求是以累计的形式表示的，如模型2-2所示。相应地，对需求的覆盖值也是以累计的形式表示的。在这种表示方式下，如果一个护士被安排在某个班次工作，那么他可能同时改变多个覆盖值（对该护士所在等级以及更低的等级的需求的覆盖）。然而这样的情况在网络流模型中是比较难处理的，因此需要把对护士人数的需求分离为对每个等级各自的需求。

分离后的需求Q_{rk}根据式（3-5）计算得到，其含义是在第k个班次需要多少个护士来完成等级为r的工作任务。

$$Q_{rk}=\begin{cases}R_{rk}-R_{r-1,k}, r>1\\ R_{1k}, r=1\end{cases} \tag{3-5}$$

构造网络的过程中将使用上述分离后的需求。

如果把每个护士分别看作一个源点，把每个班次对每个等级的工作量的需求分别看作一个汇点，把每个护士在每个班次工作看作是流出一个单位的流量，那

么需求能否被满足的问题就等价于在这个多源点多汇点的网络上，所有源点提供的流量是否能满足所有汇点的总需求量的问题，可以通过求解该网络上的最大流问题来解决。而对于多源点多汇点的网络流问题，可以通过添加一个超级源点连接原网络上的每个源点并添加一个超级汇点连接原网络上的每个汇点，把问题转化到普通的单源点单汇点网络流问题上，这里不作详细讨论。

综上所述，对原问题可行解的求解被转化为一个最大流的问题。

下面说明构造网络的具体方法。

假设共有 n 个护士被安排到日班。

网络的顶点分为 5 层：

第 1 层称为源层，只有一个顶点，即超级源点 S。

第 2 层称为护士层，有 n 个顶点，每个顶点分别对应一个被安排到日班的护士，在后面的讨论中称这些顶点为“护士顶点”。

第 3 层称为分流层，有 3×7=21 个顶点，其中第 1 个到第 7 个顶点表示等级为 1 的护士对相应班次的选择，第 8 个到第 14 个顶点表示等级为 2 的护士对相应班次的选择，第 15 个到第 21 个顶点表示等级为 3 的护士对相应班次的选择。由于高等级的护士可以替代低等级的护士工作，因此对于同一个班次来说，从每个护士顶点流出的流量可以流向该班次对应不同等级的工作量需求所代表的顶点，然而每个护士只能从中选择最多一个这样的顶点，否则会导致同一个护士在同一个班次中承担了多个人的工作量的不合理情况。分流层的作用就是用来接收护士顶点的流出的流量，表示该护士被安排到了该班次，然后分流到该班次对不同等级的工作量的需求点上，以避免上述不合理情况的出现。后面的讨论中称这些顶点为“分流顶点”，并且称第 1 个到第 7 个分流顶点为等级为 1 的分流顶点，第 8 个到第 14 个分流顶点为等级为 2 的分流顶点，第 15 个到第 21 个分流顶点为等级为 3 的分流顶点。

第 4 层称为需求层，有 3×7=21 个顶点，其中第 1 个到第 7 个顶点对应着相应班次对等级为 1 的工作量的需求，第 8 个到第 14 个顶点对应着相应班次对等级为 2 的工作量的需求，第 15 个到第 21 个顶点对应着相应班次对等级为 3 的工作量的需求，在后面的讨论中称这些顶点为“需求顶点”。

第 5 层称为汇层，只有一个顶点，即超级汇点 T。

下面说明网络中的弧的构造方法。所有弧都只存在于相邻的层中，且均由编号较小的层指向编号较大的层。

第1层到第2层的弧共有n条。超级源点S到每个护士顶点各有一条弧，容量为该护士需要工作的总天数。

第2层到第3层的弧共有n×7条。每个护士顶点到分流层上对应他的等级的7个顶点各有一条弧，容量为1，表示每个护士只能在每个班次中承担最多一个等级的工作任务。

第3层到第4层的弧共有7×3+7×2+7=42条。分流层中等级为1的7个分流顶点分别向需求层中等级为1、2和3的相应顶点连一条弧，容量为无穷大；分流层中等级为2的7个分流顶点分别向需求层中等级为2和3的相应顶点连一条弧，容量为无穷大；分流层中等级为3的7个分流顶点分别向需求层中等级为3的相应顶点连一条弧，容量为无穷大。

第4层到第5层的弧共有3×7=21条。每个需求顶点到超级汇点各连一条弧，容量为该需求顶点对应着的分离后的需求数。

例3-1：在这个简化了的例子中，只有两个护士（护士a和护士b）和3个待安排的工作天。其中，护士a的等级为1，要工作3天，护士b的等级为2，要工作1天。该例子对应的需求表如表3-1所示，其中的需求量是以累计形式表示的，根据式（3-5）计算得到的分离后的需求表如表3-2所示。对于这个例子，由上述方法构造得到的网络如图3-2所示。

表3-1 例3-1的需求表（分离需求前）

	班次1	班次2	班次3
等级1	1	1	0
等级2	1	2	1

表3-2 例3-1的分离后的需求表

	班次1	班次2	班次3
等级1	1	1	0
等级2	0	1	1

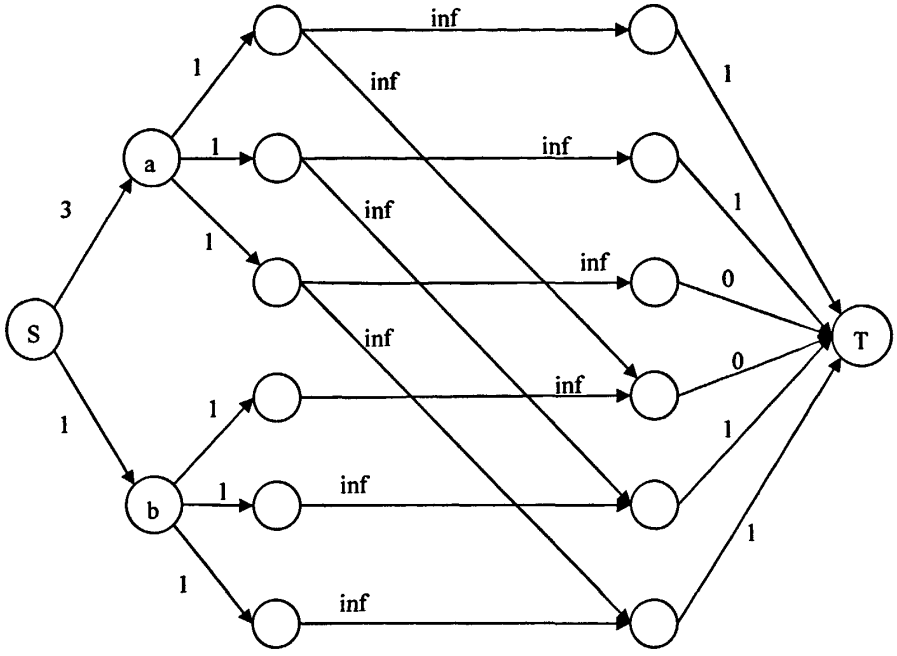

图 3-2 例 3-1 对应的网络

命题 3-4：对于一个特定的日夜分班方案，如果由上述方法构造得到的网络上的一个最大流使得需求层到汇层的所有弧满流，那么，需求可以被满足，而且可以根据该最大流构造出一个可行的方案。

证明 3-4：如果该网络上的最大流使得需求层到汇层的所有弧满流，那么意味着每个需求顶点的流出量都等于它所代表的需求量，根据流量守恒条件，它的流入量等于流出量，即有足够的护士为其提供流量（工作量），而且根据网络的构造方法，每个护士顶点的流出量不会超过他所需要工作的总量，即没有护士需要超时间工作，于是命题 3-4 的前半部分得证。

在构造可行解的时候，对于每个护士顶点，扫描它指向分流层的 7 条弧，其中满流的弧表示该班次被安排了，不满流（流量为 0）的弧表示该班次没有被安排。对于实际被安排班次总数等于所需工作天数的护士而言，他们的模式实际上已经被确定；而对于实际被安排班次总数少于所需工作天数的护士而言，他们可以在各自的可选模式集里选择权值最小的而且能覆盖到当前已安排给他的所有班次的一个模式，程序实现中可以通过预处理，使上述选择权值最小且能覆盖当前安排的模式的过程在常数时间内完成。

证毕。

对于例 3-1 而言，求得的最大流如图 3-3 所示。因为该最大流使得需求层到汇层的所有弧满流，所以该问题有可行解。根据证明 3-4 中所描述的构造方法，得到排班方案为：护士 a 被安排了模式 111，护士 b 被安排了模式 010。

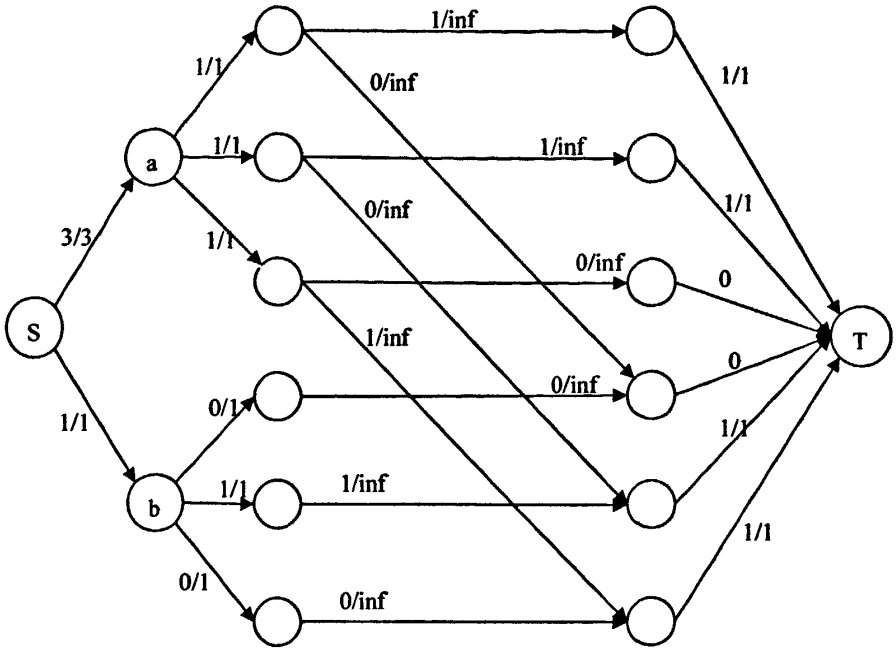

图 3-3 例 3-1 对应的一个最大流

3.4 使用模拟退火算法进行优化

本节介绍模拟退火算法以及其在本算法中的具体应用。

3.4.1 调整最大流以遍历不同的解

本节说明了本阶段的优化算法的基本思想，即通过在网络上不断地调整最大流以遍历原问题的不同的解，从而寻找当中的最优解。

命题 3-5：如果在 3.3.2 节所构造的网络上的一个最大流可以使所有需求被满足，那么该网络上的任意一个最大流都可以使所有需求被满足。

证明 3-5：采用反证法。

假设在该网络上存在一个最大流不能满足需求，即不能使需求层到汇层的所有弧均为满流状态，由于汇点的所有入弧都来源于需求层，因此，该流的值必然小于需求层到汇层的所有弧的容量之和，即小于原来的最大流的值，这与该流也是最大流的前提相矛盾，因此命题 3-5 得证。

利用命题 3-3 和命题 3-5 的性质，本论文提出通过在残余网络上寻找回路以调整最大流，从而遍历原问题的不同的解，以寻找原问题在一个特定的日夜分班方案中的最优解。

如 3.1 节所述，本论文在本阶段采用模拟退火算法进行优化。下面将介绍模拟退火算法的基本原理以及本论文所采用的相关参数设定。

3.4.2 模拟退火算法

模拟退火算法是局部搜索算法的扩展，它不同于局部搜索之处是以一定的概率选择邻域中费用值（目标函数值）较大的状态。从理论上来说，它是一个全局最优算法。模拟退火算法最早的思想由 Metropolis 在 1953 年提出，Kirkpatrick 在 1983 年成功地把它应用在组合优化问题上[27]。

退火是一种物理过程，金属物体在加热至一定的温度后，它的所有分子在状态空间中自由运动。随着温度的下降，这些分子逐渐停留在不同的状态。在同一个温度下，分子停留在能量小的状态的概率比停留在能量大的状态的概率要大。当温度趋向 0 时，分子停留在最低能量状态的概率趋向 1[27]。

模拟退火算法根据上述原理，把组合优化问题类比为金属物体，其中的每个解类比为分子的不同状态，最优解类比为能量最低的状态，目标函数值类比为状态的能量，于是，把组合优化问题 $z=\min\{f(x)\mid g(x)\geq 0, x\in D\}$ 的求解过程类比为退火过程[27]。

模拟退火算法的基本流程如算法 3-2 所示。

Step 1　任选一个初始解 s0，当前解 s:=s0，当前温度 t:=t_{init}（初始温度）。

Step 2　如果在当前温度下达到内循环终止条件，则转到 Step 3；

否则，

从当前解 s 的邻域 N(s)中随机选择一个解 s'；

计算 s'与 s 的目标函数值之差 Δ；

如果 $\Delta \le 0$，则直接接受新解，s:=s'；

否则，如果 $\exp(-\frac{\Delta}{t}) > random(0,1)$，则 s:=s'，即以概率 $\exp(-\frac{\Delta}{t})$ 接受新解；

转到 Step 2。

Step 3　执行降温操作 t:=d(t)；

如果满足全局终止条件，算法结束；

否则，转到 Step 2。

算法 3-2 模拟退火算法的基本流程

在算法 3-2 所示的模拟退火算法流程中，包含一个外循环和一个内循环。外循环控制着温度的下降变化，以及全局的终止条件。内循环则让算法在同一个温度下在不同状态之间进行随机搜索。在搜索的过程中，如果遇到更好的解则直接接受该解，否则以概率 $P=\exp(-\frac{\Delta}{t})$ 接受。从该概率的表达式可以看出，当温度很高时，接受任意解的概率约为 1，这使得搜索在开始阶段可以自由地在解空间的不同区域之间游走；而当温度很低时，接受劣解的概率几乎为 0，这使得搜索在最后阶段可以专注于在一个局部区域寻优。

模拟退火算法是一个通用的算法框架，其中涉及到如下要素：

- ✧ 初始温度的选取
- ✧ 降温函数的选取
- ✧ 同一温度下的停止准则
- ✧ 全局终止条件

本论文采用式（3-6）估算初始温度。

$$t_{init} \ge -\frac{\Delta_{max}}{\ln \theta} \tag{3-6}$$

式（3-6）的含义是，为了保证在初始阶段算法接受任意解的概率不低于一个给定值 θ，则要求 $\exp(-\frac{\Delta_{max}}{t_{init}}) \geq \theta$，即 $t_{init} \geq -\frac{\Delta_{max}}{\ln\theta}$，其中 Δ_{max} 根据 $n \times (pref_{max} - pref_{min})$ 估算，其中 n 是护士人数，$pref_{max}$ 和 $pref_{min}$ 分别是模式的权值取值范围的最大值和最小值。实验中使用的初始温度为 14000。

本论文采用式（3-7）作为降温函数，使温度在每一次降温中以相同的比率下降。

$$d(t) = t \times \alpha \tag{3-7}$$

其中 α 是一个略小于 1 的数，实验中取值为 0.9。

对于内循环的停止准则，本论文采用固定的迭代长度，即让算法在每个温度下迭代固定的次数。实验中取值为固定迭代 60 次。

而对于全局终止条件，本论文采取的方式是当温度降低到接近 0（$t < \varepsilon$）时算法终止。实验中 ε 的取值为 1e-8。

3.4.3 邻域结构

本论文提出了基于网络流模型的邻域结构，对于当前解 s（即当前网络上的一个最大流），所有可以通过一次单位流量调整得到的最大流属于 N(s)。

具体而言，本论文采用了两种邻域操作。

第一种邻域操作，记为 Greedy Neighborhood，尝试贪心地改进某个护士的当前安排，其基本流程如算法 3-3 所示。

```
Greedy Neighborhood
Input: none.
Output: an adjusted maxflow if found, null otherwise.
  for each nurse node u in nurse layer
    for each edge1 from u to middle layer
      if edge1 is full
        for each edge2 from u to middle layer
          if edge2 is empty
            if is_better(-edge1, u, edge2)
              loop:=find_loop_start_with(-edge1, u, edge2)
              if loop is not null
                for each edge in loop
                  adjust the flow in edge
                return the adjusted maxflow
  return null
```

算法 3-3 Greedy Neighborhood 的基本流程

在算法 3-3 中，is_better(-edge1, u, edge2)的含义是，如果把 edge1 的流量退回顶点 u 并送到 edge2，其结果对顶点 u 所代表的护士来说是否更好，即如果把该护士原有的一个工作天改到另外一天是否能得到对于他而言更好的模式。而 find_loop_start_with(-edge1, u, edge2)的含义是在残余网络上寻找一个回路，使得该回路必须包括以 edge1 的反向边到达 u 并从 u 向 edge2 出发的一段。

设计该邻域操作的目的包括以下两个方面：

1. 让护士避开他不希望工作的某一天；
2. 把单独的工作天合并到一个连续段中。

如 2.1 节所述，上述两个方面都是影响模式权值的因素，Greedy Neighborhood 试图针对这两个方面对一个护士进行贪心的改进。

第二种邻域操作，记为 Random Neighborhood，尝试随机地选择一个护士顶点并以它为起点调整流，其基本流程如算法 3-4 所示。

```
Random Neighborhood
Input: none.
Output: an adjusted maxflow if found, null otherwise.
  randomly select a nurse node u in nurse layer
  loop:=find_loop_start_with(u)
  if loop is not null
    for each edge in loop
      adjust the flow in edge
    return the adjusted maxflow
  else
    return null
```

算法 3-4 Random Neighborhood 的基本流程

在算法 3-4 中，find_loop_start_with(u)的含义是在残余网络上寻找以顶点 u 为起点的回路。

Random Neighborhood 是一种完全随机的邻域操作，它被设计为增强搜索的扩散性。

Greedy Neighborhood 和 Random Neighborhood 都需要在残余网络上寻找一个回路，本论文采用了两种寻找方法。

第一种方法是随机化的广度优先搜索 Randomized BFS。与一般的广度优先搜索 BFS 的区别是，Randomized BFS 每次随机地从当前队列中选择一个元素用于扩展，而不遵循先进先出的原则。

第二种方法是随机化的深度优先搜索 Randomized DFS。与一般的深度优先搜索 DFS 的区别是，Randomized DFS 对于每一个待扩展的节点以随机的顺序扩展出它的子节点。

这两种方法都在原有的图遍历方法（BFS 和 DFS）中引入了随机化的因素。这是因为，在模拟退火算法的整个过程中，该网络的拓扑结构是固定的，如果以一般的 BFS 或者 DFS 的固定搜索方法只能找到部分的回路，而无法遍历整个解空间，因此需要引入随机化。

上述两种邻域操作的另外一个共同点是，它们都要求找到的回路中必须包含某个护士顶点。这是因为，目标函数值是由护士层到分流层的弧的流量决定的，如果沿着一条不经过任何护士顶点的回路调整流量，那么护士层到分流层的所有弧的流量都不会改变，从而调整后的流与原来的流具有一样的目标函数值。为了避免这种情况，两种邻域操作都要求回路中必须包含某个护士顶点。

3.4.4 引入禁忌表

禁忌搜索（Tabu Search）算法是局部邻域搜索算法的推广，是人工智能在组合优化算法中的一个成功应用。Glover 在 1986 年提出这个概念，进而形成了一套完整算法。禁忌搜索算法的特点是采用了禁忌技术，用以禁止重复前面的工作。具体来说，禁忌搜索算法用一个禁忌表记录下已经到达过的局部最优点或者达到局部最优的一些过程，在下一次搜索中，利用禁忌表中的信息不再或有选择地搜索这些点或过程，以此来跳出局部最优点[27]。

禁忌搜索算法主要有如下要素：

- ✧ 禁忌对象的选取，即对什么进行禁忌；
- ✧ 禁忌长度的确定，即对上述对象禁忌多长的时间；
- ✧ 藐视准则，即在什么情况下可以无视禁忌的状态。

本论文并不直接使用禁忌搜索算法，只是借用它的思想，通过在模拟退火算法中引入禁忌表来避免循环搜索。具体来说，在模拟退火算法的迭代过程中，每当算法接受一个新解时，把导致该新解时用于调整流量的回路中的一条弧设为禁忌状态，以使得在接下来的若干次迭代中不会把流量还原为原来的状态，避免循环搜索。

本论文采用固定的禁忌长度，实验中取值为 7，不采用藐视准则。

3.5 本章小结

本章介绍了本论文对一个已有的护士排班问题所提出的一个两阶段的求解算法，该算法在第一阶段中使用分枝限界算法寻找可行的日夜分班方案，在第二

阶段中使用模拟退火算法对每个可行的分班方案进行优化，以寻找全局的最优解。该算法的特点是把问题转化到网络流模型上，通过求解最大流问题求得原问题的一个可行解，并以该最大流为起点，通过在残余网络上寻找回路并调整最大流的邻域操作进行优化，该邻域操作保证了搜索总在可行域中进行，因而提高了优化的效率。本章对算法中的每个组成部分展开了讨论。该算法的实际效果将在第5章中通过实验结果来验证。

第 4 章 新约束条件与模型的研究

本章介绍了本论文完成的第二部分主要工作，即针对新约束条件与模型的研究。本论文在原问题的基础上增加了一个约束条件，要求尽量避免护士降级工作的情况出现。针对该约束条件，本论文一方面通过修改原有模型得到了一个满足要求的改进模型，另一方面通过分析并推广原有的建模方法得到了一个更通用的模型并应用到本问题中。本论文使用ILOG CPLEX[1] 11.1求解本章所提出的模型。

4.1 增加新的约束条件

在原问题中，允许高等级的护士替代低等级的护士工作，这在护士人数不足的情况下是无奈之举。然而当护士人数足够的时候，这种情况是应当避免的。因为让高等级的护士替代低等级的护士工作，一方面造成资源的浪费，另一方面可能引起高等级的护士对工作安排的不满。基于以上考虑，本论文在原问题的基础上，提出一个新的约束条件，要求尽量避免护士降级工作的情况出现。具体的实现方法是对每个降级工作的情况增加一定的惩罚值。

4.2 改进原有模型

上述关于降级工作的约束条件要求不仅要决定每个护士在哪些班次工作，而且还要决定每个护士在每个班次中具体负责了哪个等级的工作。一个直观的想法是引入新的决策变量 y_{ikr}，用以表示护士 i 在第 k 个班次负责了等级 r 的工作，由此得到模型 4-1。

[1] http://www.ilog.com/products/cplex/

模型 4-1 对原模型的改进 I

min
$$z=\sum_{i\in N}\sum_{j\in F(i)}p_{ij}x_{ij}+\sum_{i\in N}\sum_{r\in[g_i+1,3]}w_{g_i r}\sum_{k\in[1,14]}y_{ikr} \tag{4-1}$$

subject to
$$\sum_{j\in F(i)}x_{ij}=1,\forall i\in N \tag{4-2}$$
$$\sum_{i\in N_r}\sum_{j\in F(i)}x_{ij}a_{jk}y_{ikr}\geq Q_{rk},\forall r\in[1,3],k\in[1,14] \tag{4-3}$$
$$\sum_{r\in[g_i,3]}y_{ikr}=\sum_{j\in F(i)}x_{ij}a_{jk},\forall i\in N,k\in[1,14] \tag{4-4}$$
$$x_{ij}\in\{0,1\},\forall i\in N,j\in F(i) \tag{4-5}$$
$$y_{ikr}\in\{0,1\},\forall i\in N,k\in[1,14],r\in[1,3] \tag{4-6}$$

在模型 4-1 中，

- y_{ikr} 是新引入的决策变量，为 1 表示第 i 个护士在第 k 个班次负责第 r 个等级的工作，否则为 0；
- w_{ab} 是让等级 a 的护士降级负责等级 b 的工作时的惩罚值，特别地，$w_{aa}=0$；
- g_i 表示护士 i 的等级；
- Q_{rk} 是分离后的需求量，表示第 k 个班次需要多少人来负责等级 r 的工作，由式（3-5）计算得到；
- 其他符号的含义同模型 2-2；
- 式（4-1）表示目标函数在式（2-6）的基础上增加了降级工作的惩罚值；
- 式（4-3）表示护士 i 不仅要被安排在第 k 个班次工作，而且还要被指定为负责第 r 个等级的工作时，才能对 Q_{rk} 有贡献；
- 式（4-4）表示，如果护士 i 确实在第 k 个班次工作（等式右边为 1），那么他必须在班次 k 负责一个且仅一个低于或等于他自身等级的任务（使得等式左边也为 1）；如果护士 i 没有在第 k 个班次工作（等式右边为 0），那么他就不能在第 k 个班次负责任何一个等级的任务（使得等式左边也为 0）。

模型 4-1 是一个整数二次约束规划模型，使用 CPLEX 求解该模型的效率非常低，因此需要寻找更有效的模型。

建立模型4-1的思路是决定护士工作班次的同时决定他负责哪一个等级的工作，因此使得在模型中出现了二次的约束条件，导致求解的困难。实际上可以换一个思路建立模型，即先决定护士在哪些班次工作，然后再决定护士在每个班次中在等级之间的“流动”，由此得到模型 4-2。

模型 4-2 对原模型的改进 II

min
$$z=\sum_{i\in N}\sum_{j\in F(i)}p_{ij}x_{ij}+\sum_{i\in N}\sum_{r\in[g_i+1,3]}w_{g_ir}\sum_{k\in[1,14]}y_{ikr} \tag{4-7}$$

subject to
$$\sum_{j\in F(i)}x_{ij}=1,\forall i\in N \tag{4-8}$$
$$\sum_{i\in N_r}y_{ikr}\geq Q_{rk},\forall r\in[1,3],k\in[1,14] \tag{4-9}$$
$$\sum_{r\in[g_i,3]}y_{ikr}=\sum_{j\in F(i)}x_{ij}a_{jk},\forall i\in N,k\in[1,14] \tag{4-10}$$
$$x_{ij}\in\{0,1\},\forall i\in N,j\in F(i) \tag{4-11}$$
$$y_{ikr}\in\{0,1\},\forall i\in N,k\in[1,14],r\in[1,3] \tag{4-12}$$

在模型 4-2 中，

- ✧ y_{ikr}是决策变量，为 1 表示护士 i 在第 k 个班次“流动”到第 r 个等级，否则为 0；
- ✧ 其他符号的含义同模型 4-1；
- ✧ 式（4-9）的含义是，对于所有等级高于或者等于 r 的护士，如果他们在第 k 个班次有“流动”到等级 r，那么就能对Q_{rk}的需求有贡献；
- ✧ 式(4-10)的含义是，如果护士 i 确实在第 k 个班次工作，那么他必须“流动”到低于或等于他自身等级的某个等级；否则，他不能有任何“流动”。

模型 4-2 把模型 4-1 改进为一个整数线性规划模型，使用 CPLEX 已经可以很快地求解。模型 4-2 比模型 2-2 多了|N|×3×14 个变量与|N|×14 个不等式。

进一步研究可以发现，模型 4-2 可以被改进为模型 4-3。

模型 4-3 对原模型的改进 III

$$\min \quad z=\sum_{i\in N}\sum_{j\in F(i)} p_{ij}x_{ij}+\sum_{r\in[1,2]}\sum_{s\in[r+1,3]} w_{rs}\sum_{k\in[1,14]} d_{krs} \tag{4-13}$$

subject to

$$\sum_{j\in F(i)} x_{ij}=1,\forall i\in N \tag{4-14}$$

$$\sum_{s\in[1,r]} d_{ksr}\geq Q_{rk},\forall r\in[1,3],k\in[1,14] \tag{4-15}$$

$$\sum_{s\in[r,3]} d_{krs}=\sum_{i\in M_r}\sum_{j\in F(i)} x_{ij}a_{jk},\forall r\in[1,3],k\in[1,14] \tag{4-16}$$

$$x_{ij}\in\{0,1\},\forall i\in N,j\in F(i) \tag{4-17}$$

$$d_{krs} \text{ is integer},\forall k\in[1,14],r\in[1,3],s\in[r,3] \tag{4-18}$$

在模型 4-3 中，

- ✧ d_{krs} 是决策变量，表示在第 k 个班次，等级为 r 的护士“流动”到等级 s 的总人数；
- ✧ M_r 表示等级为 r 的所有护士的集合；
- ✧ 式（4-15）的含义是，对于每个班次 k，所有高于或等于等级 r 的护士“流动”到等级 r 的人数之和，应该大于或者等于需求量 Q_{rk}；
- ✧ 式（4-16）的含义是，在第 k 个班次，等级 r 的护士“流动”到低于或等于 r 的各等级的人数之和应该等于在第 k 个班次工作的等级为 r 的护士的总人数。

模型 4-3 的思路是把同一个等级的护士作为一个整体来“流动”，从而进一步简化了模型 4-2。模型 4-3 比模型 2-2 多了 14×3×3 个变量与 3×14 个不等式，即只增加了常数个变量与不等式，优于模型 4-2。

至此，本论文针对避免降级工作的新约束条件，通过对原模型的改进，得到了模型 4-3，使用 CPLEX 求解该模型的效率非常高，具体的实验结果将在第 5 章介绍。

本章的后续部分介绍了本论文对该问题的另外一种尝试，在分析原有建模方法的基础上将其推广为一个更通用的版本，并应用到本问题中。

4.3 对原有建模方法的分析

为了方便描述，本节首先定义排班表以及其中的行约束条件和列约束条件。

定义 4-1：排班表

排班表是指把具体的排班方案写成的一个二维表格，其中每一行代表一个护士的排班结果，每一列代表一个特定的班次中有哪些护士值班，如表 4-1 所示。如果排班表中第 i 行第 j 列的元素为 1，那么表示第 i 个护士要在第 j 个班次工作，否则该元素为 0。

表 4-1 包含 3 个护士和 4 个班次的排班表的例子

	班次 1	班次 2	班次 3	班次 4
护士 1	1	0	1	0
护士 2	0	1	1	0
护士 3	0	0	1	1

定义 4-2：行约束条件

行约束条件是指仅以排班表中的行为单位就可以描述的约束条件，即只针对单个护士的各种约束条件，如工作总天数的要求、连续工作天数的限制、不同班次之间应该有的时间间隔、当前护士对某天的休假请求、当前护士在上一个排班阶段的排班对当前排班阶段的影响，等等。

定义 4-3：列约束条件

与行约束条件对应，列约束条件是指必须以排班表中的列为单位描述的约束条件，即针对在同一个班次值班的护士之间的各种约束条件，如要求某些护士一起工作或避免他们同时工作、尽量避免让高等级的护士替代低等级的护士工作，等等。

原有的建模方法本质上可以概括为以下 3 点：

1. 对排班表中的每一行，即每个护士的排班结果，枚举其所有可能的取值，把原本需要的多个决策步骤简化为一个单项选择的问题；
2. 对每行的每个可能的取值，即行模式，根据各种行约束条件计算一个权值，把原有的各种线性或非线性的约束条件简化成一个数值附加在行模式上；

3. 把列约束条件保留到最终的模型中，实际上原问题中唯一的列约束条件就是每个班次对不同等级的护士人数的需求量。

这种建模方法具有通用性。只要规模允许，即每行可能的取值不太多的情况下，该建模方法可以直接应用在各种具有不同行约束条件的护士排班问题上，而且可以直接使用各种针对该简化后的模型的通用算法求解，如使用专业的整数规划模型求解器等。如果问题的规模太大，无法事先枚举出所有可能的行取值，则可以结合列生成算法，逐渐生成可选的行模式集合，如 Bard 等人提出的方法[11]。

然而，上述建模方法把列约束条件都保留在最终的模型当中，对于那些有比较复杂的列约束条件的护士排班问题，运用上述建模方法可能会得到一个比较复杂的模型，例如一个非线性的整数规划模型。根据本论文的实验观察，CPLEX 求解非线性模型的效率非常低，而对于形式简单的整数线性规划模型，即使变量数比较多，其求解速度依然很快。受此启发，本文对原有建模方法进行了关于列约束条件的推广。

4.4 推广原有的建模方法

在原有的建模方法基础上，本论文把它推广到同时预处理列约束条件。推广后的建模方法可以概括为以下 5 个步骤：

1. 对于每一行，枚举所有可能的行模式；
2. 根据行约束条件，为每行的每个行模式计算一个权值；
3. 对于每一列，枚举所有可能的列模式；
4. 根据列约束条件，为每列的每个列模式计算一个权值；
5. 把问题建模为一个单项选择规划的整数线性规划模型，其中的约束条件包括两类，第一类约束条件要求每行每列分别选择一个且仅一个模式，第二类约束条件要求行列模式的选择要一致，不能相互冲突，目标函数则是所有被选择的行模式和列模式的权值之和。

使用上述建模方法得到的一个通用模型如模型 4-4 所示。

模型 4-4 推广得到的通用模型

min
$$z=\sum_{r\in R}\sum_{i\in FR(r)} p_{ri}x_{ri}+\sum_{c\in C}\sum_{j\in FC(c)} q_{cj}y_{cj} \tag{4-19}$$

subject to
$$\sum_{i\in FR(r)} x_{ri}=1,\forall r\in R \tag{4-20}$$
$$\sum_{j\in FC(c)} y_{cj}=1,\forall c\in C \tag{4-21}$$
$$\sum_{i\in FR'(r,c)} x_{ri}=\sum_{j\in FC'(c,r)} y_{cj},\forall r\in R,c\in C \tag{4-22}$$
$$x_{ri}\in\{0,1\},\forall r\in R,i\in FR(r) \tag{4-23}$$
$$y_{cj}\in\{0,1\},\forall c\in C,j\in FC(c) \tag{4-24}$$

在模型 4-4 中，

- r 是行的索引；
- c 是列的索引；
- p_{ri} 表示第 r 行的第 i 个行模式的权值；
- q_{cj} 表示第 c 列的第 j 个列模式的权值；
- R 表示所有行的集合；
- C 表示所有列的集合；
- FR(r)表示第 r 行的可选行模式集；
- FC(c)表示第 c 列的可选列模式集；
- FR'(r, c)表示第 r 行的可选行模式集中，能覆盖到第 c 列的所有行模式的集合；
- FC'(c, r)表示第 c 列的可选列模式集中，能覆盖到第 r 行的所有列模式的集合；
- x_{ri} 是行决策变量，表示第 r 行的第 i 个行模式是否当选；
- y_{cj} 是列决策变量，表示第 c 列的第 j 个列模式是否当选；
- 式（4-19）是要求最小化的目标函数，即被选择的所有行模式与列模式的权值之和；
- 式（4-20）要求每行必须选择一个且仅一个行模式；
- 式（4-21）要求每列必须选择一个且仅一个列模式；

✧ 式（4-22）保证了所选择的行模式与列模式之间保持一致，相互不冲突；

✧ 式（4-23）和式（4-24）要求决策变量的取值为0或者1。

命题4-1：模型4-4可以保证行列模式的选取没有冲突。

证明4-1：采用反证法。

假设存在模型4-4的一个可行解S使得该解中选择的行模式与列模式之间有冲突，那么起码存在一个位置（r，c）的取值是有冲突的，不失一般性，设在解S中，第r行选择的行模式要求位置（r，c）的值为0，即第r行选择了一个不能覆盖第c列的行模式，而第c列选择的列模式要求位置（r，c）的值为1，因此有 $\sum_{j\in FC'(c,r)} y_{cj}=1$ 且 $\sum_{i\in(FR(r)-FR'(r,c))} x_{ri}=1$，而由于S是可行解，根据式（4-20）有 $\sum_{i\in FR(r)} x_{ri}=1$，则 $\sum_{i\in FR'(r,c)} x_{ri}=\sum_{i\in FR(r)} x_{ri}-\sum_{i\in(FR(r)-FR'(r,c))} x_{ri}=0$，即 $\sum_{i\in FR'(r,c)} x_{ri}\neq\sum_{j\in FC'(c,r)} y_{cj}$，不满足式（4-22），这与S是可行解的前提相矛盾。因此，命题4-1得证。

值得注意的是，在模型4-4中没有出现常见的关于护士人数的需求的约束条件，因为该约束条件属于列约束条件，已经在预处理阶段被隐式地处理了。实现中可以有不同的方法。例如，直接忽略那些不能满足护士人数需求的列模式，使得每一个可选的列模式都是已经满足需求的，这对于保证有可行解的问题是合理的；另一种方法是，把不能满足护士人数需求的列模式的权值设为一个比较大的值，这对于那些不一定有可行解或者允许出现非可行解的场合是适用的。

模型4-4是把原有建模方法推广得到的结果，在规模允许的前提下，该方法比原有的方法更具有通用性，可以把各种复杂的列约束条件也简化为一个权值附加在列模式上，使得整个模型中不涉及任何与问题相关的不等式约束，是一个形式统一的模型，有利于开展针对该模型的通用算法的研究。

4.5 对模型的进一步推广

对于避免护士降级工作的约束条件来说，同一个等级的护士之间是没有区别的，因此可以模仿建立模型4-3的思路，把同一个等级的护士当作一个整体来处理。于是对于每一列来说，不同的列模式是指不同的三元组（A1，A2，A3），

其中 A1 为被安排到该班次的等级为 1 的护士人数，A2 为被安排到该班次的等级为 2 的护士人数，A3 为被安排到该班次的等级为 3 的护士人数。于是需要枚举的列模式个数就由原本的 2^n 缩减到不超过$(n/3)^3$，这是一个非常可观的改进。改进后的模型如模型 4-5 所示。

模型 4-5 推广模型的改进

min
$$z=\sum_{r\in R}\sum_{i\in FR(r)} p_{ri}x_{ri}+\sum_{c\in C}\sum_{j\in FC(c)} q_{cj}y_{cj} \tag{4-25}$$

subject to
$$\sum_{i\in FR(r)} x_{ri}=1,\forall r\in R \tag{4-26}$$
$$\sum_{j\in FC(c)} y_{cj}=1,\forall c\in C \tag{4-27}$$
$$\sum_{r\in s}\sum_{i\in FR'(r,c)} x_{ri}=\sum_{j\in FC'(c,s)} I_{cjs}y_{cj},\forall s\in S,c\in C \tag{4-28}$$
$$x_{ri}\in\{0,1\},\forall r\in R,i\in FR(r) \tag{4-29}$$
$$y_{cj}\in\{0,1\},\forall c\in C,j\in FC(c) \tag{4-30}$$

在模型 4-5 中，

- S 表示超级行的集合，所谓超级行即由若干行所组成的整体；
- s 是超级行的索引；
- I_{cjs} 表示第 c 列的第 j 个列模式与超级行 s 相交的次数；
- $FC'(c,s)$ 表示第 c 列的列模式中能覆盖到超级行 s（相交次数大于 0）的列模式的集合；
- 其他符号含义同模型 4-4。

在模型 4-5 中，引入了超级行的概念，这是对问题的进一步抽象，把无区别的多个行作为一个整体来看待。由于引入了超级行，因此在行列一致性的约束中引入了相交次数的概念，即式（4-28）中的 I_{cjs}。其实在模型 4-4 中，相交次数的概念也是存在的，只是由于在该模型中每行与每列相交最多一次，因此系数 1 被省略了。

模型 4-5 是模型 4-4 的一个更通用的版本，因为只要把模型 4-4 中的每一行作为一个超级行，就可以把它归约到模型 4-5。

进一步来说，还可以类似地引入超级列的概念，让模型变得更加通用。

4.6 本章小结

本章在原问题的基础上增加了避免护士降级工作的约束条件，并针对该约束条件在原有模型基础上进行修改，提出了三个改进的模型，这些模型都可以满足新约束条件的要求。在第三个改进模型中，变量与不等式的数量只比原模型多了常数个。另外，通过分析原有的建模方法，本章引入了行约束条件、列约束条件等概念，并对原有建模方法进行了推广，提出了一种更通用的建模方法并把它应用到本问题中。使用本章提出的推广后的建模方法可以得到一个形式统一的不依赖于具体问题的单项选择规划模型，如果可以针对该模型研究出有效的通用算法，将可以为护士排班问题提供一个通用的解决方案。在第5章中将通过实验验证应用这些模型求解问题的效率。

第 5 章 实验结果与分析

本章列出了本论文的实验结果并对其进行了分析。

对于第 3 章提出的两阶段求解算法，实验中使用了 30 组标准测试数据，其他文献也曾经对同样的数据进行过实验[17][18][19][20][22][23]。本章将把本算法的实验结果与这些文献中提供的结果进行比较。

本章所描述的所有实验的运行环境为：Intel Core2 Duo CPU T6400@2.00GHZ、2G RAM、Laptop；操作系统为 Windows XP；对于第 3 章的两阶段求解算法，程序的实现使用了 C++语言；在对第 4 章提出的新模型的实验中，使用 Java 语言编程并调用 ILOG CPLEX 11.1 的数学规划模型求解器求解。

5.1 分枝限界算法的可行性剪枝效果

本节验证了两阶段求解算法中分枝限界算法的可行性剪枝的效果。

该部分相关的实验结果统计如表 5-1 所示。其中，DFS 是指不带可行性剪枝的朴素的深度优先搜索算法，B&B 是指带可行性剪枝的分枝限界算法。实验过程中直接搜索所有可行的日夜分班方案，不进行第二阶段的优化，也不使用最优性剪枝。目的是检验可行性剪枝带来的搜索效率的提高。

在表 5-1 的第二列中，标记为“>60”的位置表示 DFS 算法在 60 秒以内无法完成搜索，因此，在最后一行的平均值中，第二列的“>30.4”其实也是不准确的，只能作为一个下界来看待。

对于 30 组测试数据，使用了可行性剪枝以后，平均运行时间从大于 30 秒降为 1.7 秒，即平均仅需要朴素 DFS 算法的 4.1%的时间，其中，有 26 组数据在 1 秒内完成，有 28 组数据在 3 秒内完成，运行时间最长的两组数据分别需要 16 秒和 20 秒，而 DFS 算法在对应的这两组数据上分别需要多于 60 秒和 55 秒。由此可见，本文提出的分枝限界算法的可行性剪枝是有效的，避免了很多没有意义的搜索。

表 5-1 分枝限界算法的可行性剪枝效果

Case#	DFS 运行时间（秒）	B&B 运行时间（秒）	B&B 时间/DFS 时间 %
1	>60	3	5
2	>60	16	27
3	23	2	8.7
4	55	20	36
5	23	1	4.3
6	23	1	4.3
7	5	1	20
8	5	0	0
9	43	1	2.3
10	37	1	2.7
11	23	0	0
12	9	0	0
13	14	0	0
14	15	0	0
15	15	0	0
16	>60	0	0
17	5	0	0
18	>60	0	0
19	35	1	2.9
20	>60	1	1.7
21	>60	1	1.7
22	23	1	4.3
23	5	0	0
24	43	1	2.3
25	23	0	0
26	9	0	0
27	14	0	0
28	40	0	0
29	>60	0	0
30	5	0	0
Avg.	30.4	1.7	4.1

5.2 分枝限界算法的最优性剪枝效果

本节验证了两阶段求解算法中分枝限界算法的最优性剪枝的效果。

该部分相关的实验结果统计如表 5-2 所示。其中第二列的数值是每组测试数

据中可行的日夜分班方案数，第三列的数值是经过最优性剪枝后需要进入第二阶段优化的可行分班方案数，第四列的剪枝率是按照式（5-1）计算的。

$$剪枝率=\frac{可行日夜分班方案数-有效日夜分班方案数}{可行日夜分班方案数}\times 100\% \qquad (5\text{-}1)$$

在30组测试数据当中，平均剪枝率为98.01%，其中，有15组数据的剪枝率大于等于99%，有29组数据的剪枝率大于等于94%，剪枝率最低的一组数据的剪枝率为83%。由此可见，本算法中使用的最优性剪枝是有效的，节省了大量的计算时间。

表5-2 分枝限界算法的最优性剪枝效果

Case#	可行日夜分班方案数	有效日夜分班方案数	剪枝率%
1	15947	18	99.89
2	105448	194	99.82
3	17612	33	99.81
4	132457	2	99.99
5	11090	4	99.96
6	7433	3	99.96
7	10944	109	99
8	4621	8	99.83
9	10570	29	99.73
10	6443	118	98.17
11	1620	76	95.31
12	810	137	83.09
13	2928	42	98.57
14	1650	2	99.88
15	1429	71	95.03
16	845	17	97.99
17	426	9	97.89
18	314	13	95.86
19	7792	27	99.65
20	6386	7	99.89
21	7932	154	98.06
22	7433	3	99.96
23	4621	21	99.55
24	10570	63	99.4
25	1620	40	97.53
26	810	47	94.2
27	2928	64	97.81
28	439	11	97.49
29	845	16	98.11
30	426	5	98.83

Avg.	12812.97	44.77	98.01

5.3 模拟退火算法优化的效果

本节验证了两阶段求解算法中使用模拟退火算法对每个可行日夜分班方案进行优化的效果。

该部分的实验结果统计如表5-3所示。其中第二列的数值是指不调用第二阶段的优化方法所获得的结果，第三列显示的是调用了第二阶段的优化方法后获得的结果，第四列的优化率是按照式（5-2）计算的。从中可以看出，通过使用模拟退火算法进行优化，求得的目标函数值平均降低了80.69%，因此，算法第二阶段的优化是有效的。

$$优化率=\frac{初始结果-优化后的结果}{初始结果}\times 100\% \quad (5\text{-}2)$$

表5-3 模拟退火算法优化的效果

Case#	初始结果	优化后的结果	优化率%
1	30	8	73.33
2	118	49	58.47
3	90	50	44.44
4	45	17	62.22
5	61	11	81.97
6	123	2	98.37
7	76	1	98.68
8	95	7	92.63
9	58	0	100
10	47	25	46.81
11	30	0	100
12	47	1	97.87
13	24	0	100
14	113	48	57.52
15	177	2	98.87
16	169	63	62.72
17	116	15	87.07
18	140	35	75
19	240	63	73.75
20	195	41	78.97
21	43	10	76.74

22	149	13	91.28
23	99	19	80.81
24	113	3	97.35
25	84	3	96.43
26	56	3	94.64
27	39	4	89.74
28	156	27	82.69
29	270	107	60.37
30	195	74	62.05
Avg.	106.6	23.37	80.69

5.4 本文算法与其他文献算法的结果比较

本节把本算法的实验结果与最优解以及其他文献的算法的结果进行了比较。实验中使用20个不同的随机种子对每组测试数据分别运行20次。

5.4.1 与最优解的比较

本算法的结果与最优解的比较如表5-4所示。其中，第二列（IP）的值是指使用精确算法求解模型2-2的整数线性规划模型所得到的结果，即最优解；第三列（Best）的值是20次运行中的最好结果；第四列（Avg.）的值是20次运行的结果的平均值；第五列（Avg. Gap%）的值是根据式（5-3）计算的。对于IP最优解是0的情况，如果直接采用式（5-3）计算会出现除数为0的情况，因此需要特别处理，但是由于在这些情况中被除数也恰好都为0，因此可以认为Gap是0。

$$\text{Avg.Gap}=\frac{\text{本算法平均值}-\text{IP最优值}}{\text{IP最优值}}\times 100\% \qquad (5\text{-}3)$$

表5-4 本算法的解与最优解的比较

Case#	IP	Best	Avg.	Avg. Gap%
1	8	8	8	0
2	49	49	49.41	0.84
3	50	50	50	0
4	17	17	17	0
5	11	11	11	0
6	2	2	2	0

7	1	1	1	0
8	7	7	7.03	0.49
9	0	0	0	0
10	25	25	25	0
11	0	0	0	0
12	1	1	1	0
13	0	0	0	0
14	48	48	48	0
15	2	2	2	0
16	63	63	63	0
17	15	15	15	0
18	35	35	35.72	2.07
19	62	62	64.48	4.00
20	40	40	40.93	2.33
21	10	10	10	0
22	13	13	13.59	4.51
23	19	19	19.38	2
24	3	3	3	0
25	3	3	3	0
26	3	3	3	0
27	4	4	4	0
28	27	27	27.21	0.77
29	107	107	107.07	0.06
30	74	74	74.21	0.28
Avg.	23.3	23.3	23.5	0.58

从表5-4中可以看出，通过使用不同的随机种子运行20次并取其中最好的结果的方法，本算法对于30组数据均能求得最优解。另外，表5-4的第五列（Avg. Gap）验证了本算法以不同的随机种子分别运行的效果的稳定性，30组数据中有20组数据的Avg. Gap为0%，即在每一次运行中都求得了最优解，所有数据的平均Avg. Gap为0.58%，而对于Avg. Gap比较大的数据，其平均值与最优值的绝对差值不大于3，根据Li等人在文献中所述，与最优解的值相差3以内的解对于医院来说都是可以接受的[19]。

5.4.2 与已有文献的比较

本部分把本算法的结果与Li等人在文献中列出的其他算法的结果[19]进行对比，如表5-5所示，其中：

- ✧ GA是指Aickelin等人提出的遗传算法[18]；
- ✧ IGA是指Aickelin等人提出的间接的遗传算法[17]；

- LCS 是指 Li 等人提出的分类系统的方法[20]；
- EDA 是指 Aickelin 等人提出的分布估算算法[22]；
- HEDA 是指 Aickelin 等人提出的结合了局部搜索的分布估算算法[23]；
- CHEE 是指 Li 等人提出的基于分量的进化式破坏与修复的算法[19]。

其中列“CHEE”的数据是该算法 20 次运行中的最优结果，作为比较，列“本算法”中的数据也是本算法 20 次运行中的最优结果，即表 5-4 中列“Best”所示的数据。在表 5-5 中，每一行的最优值被加粗显示。

表 5-5 与已有文献的结果比较

Case#	GA	IGA	LCS	EDA	HEDA	CHEE	本算法
1	**8**	**8**	9	**8**	**8**	**8**	**8**
2	50	51	60	56	55	50	**49**
3	**50**	51	68	**50**	**50**	**50**	**50**
4	**17**	**17**	**17**	**17**	**17**	**17**	**17**
5	**11**	**11**	15	**11**	**11**	**11**	**11**
6	**2**	**2**	4	3	3	**2**	**2**
7	**1**	**1**	32	10	9	**1**	**1**
8	**8**	7	7	7	7	7	7
9	**0**	**0**	6	1	1	**0**	**0**
10	**25**	**25**	38	26	26	**25**	**25**
11	**0**	**0**	3	1	**0**	**0**	**0**
12	**1**	**1**	**1**	**1**	**1**	**1**	**1**
13	**0**	**0**	**0**	**0**	**0**	**0**	**0**
14	**48**	**48**	93	52	51	**48**	**48**
15	**2**	4	19	28	18	**2**	**2**
16	**63**	64	67	65	65	**63**	**63**
17	17	**15**	56	109	23	**15**	**15**
18	**35**	38	41	38	38	**35**	**35**
19	95	65	123	159	111	66	**62**
20	41	42	42	43	41	**40**	**40**
21	12	12	15	11	11	**10**	**10**
22	16	15	30	25	23	**13**	**13**
23	48	25	34	24	22	**19**	**19**
24	**3**	**3**	15	6	6	**3**	**3**
25	6	6	28	7	7	**3**	**3**
26	**3**	**3**	**3**	**3**	**3**	**3**	**3**
27	**4**	**4**	18	5	5	**4**	**4**
28	29	30	37	30	30	**27**	**27**
29	110	110	110	109	109	**107**	**107**
30	75	74	125	171	107	96	**74**
Avg.	26	24.4	37.2	35.87	28.6	24.2	**23.3**

从表5-5中可以看出，本算法的效果优于上述除CHEE算法以外的其他算法，在20次运行的最优值的比较上与CHEE算法的效果相当（CHEE算法只对其中的3组数据没有达到与本算法一样的解）。

图5-1通过20次运行的平均值比较了CHEE算法与本算法的运行效果。需要说明的是，CHEE算法在对第18组和第25组数据的20次运行中，各有两次运行没有找到可行解，在计算平均值的时候对这种情况设定了一个特定的惩罚值255。尽管如此，从图5-1中可以看出，除了第18组和第25组数据以外，在第14组、第17组、第19组、第22组、第23组、第27组和第30组数据中，CHEE算法20次运行的平均值与本算法的平均值相差较大。因此，本算法比CHEE算法更稳定。

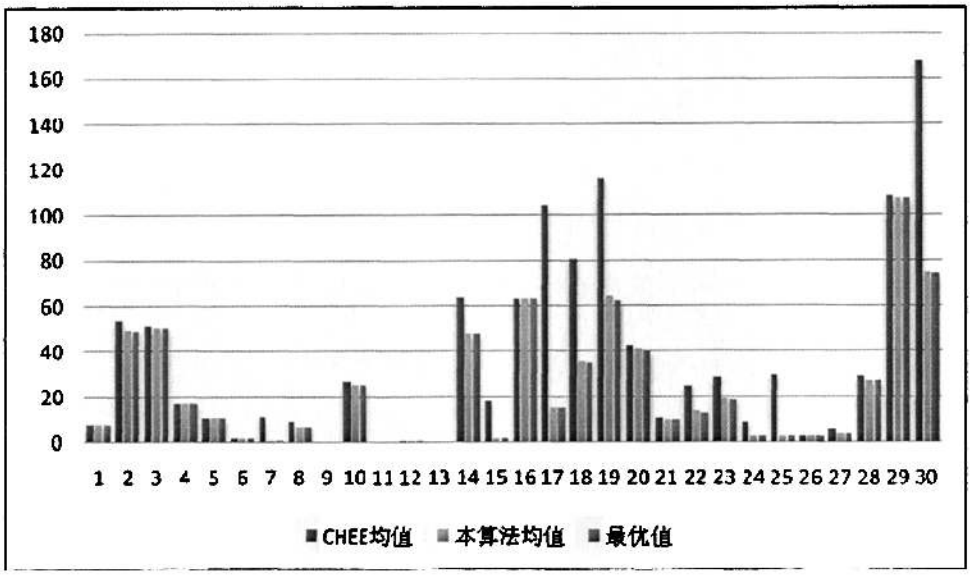

图5-1 CHEE算法与本算法的20次运行平均值的比较

5.4.3 运行时间的比较

表5-6列出了本算法对每一组测试数据的20次运行的平均运行时间。对30组测试数据的求解中，每组数据所需的平均运行时间为4.7秒，其中，有16组数据的平均运行时间在2秒以内，有25组数据的平均运行时间在10秒以内，最慢的平均运行时间为23秒。根据文献中的描述，CHEE算法每次运行的时间为几毫秒到20秒之间，而该文献的实验是在Pentium 4 2.1 GHz的机器上进行的[19]。由此可以看出，本文的算法与CHEE算法的运行效率相当。

表 5-6 本算法的平均运行时间

Case#	本算法平均运行时间（秒）
1	2
2	23
3	3.2
4	0
5	0
6	0.3
7	11
8	0.1
9	1.7
10	14
11	10
12	17
13	5
14	0
15	7
16	2
17	0
18	0.3
19	2.1
20	0.7
21	16
22	0
23	1.5
24	6.2
25	4
26	4.7
27	7
28	1
29	1
30	0
Avg.	4.7

5.5 利用改进模型求解带降级工作惩罚值的问题

本部分的实验基于上述 30 组标准测试数据，其中设定降级工作惩罚值为 $w_{12}=w_{23}=1$，$w_{13}=2$。

使用模型 4-3 和模型 4-5 分别求解带新约束条件的问题所需的时间如表 5-7 所示。使用 ILOG CPLEX 11.1 分别求解模型 4-3 和模型 4-5，对所有数据都可以求得最优解。由于模型 4-3 是针对性地从原模型改进而来的，因此其求解效率非常高，每组测试数据的平均求解时间为 0.35 秒。而模型 4-5 作为一个通用的模型，其求解效率也是不错的，对每组测试数据平均求解时间为 2.61 秒，在 30 组测试数据中有 20 组测试数据可以在 2 秒内完成求解，最慢的一组数据需要 9.84 秒。

表 5-7 使用模型 4-3 和模型 4-5 求解带新约束条件的问题的时间

Case#	求解时间（秒）	
	模型 4-3	模型 4-5
1	0.3	1.99
2	0.31	9.06
3	0.3	1.36
4	0.28	3.34
5	0.3	5.14
6	0.31	1.55
7	0.31	7.28
8	0.27	0.66
9	0.33	1.88
10	0.3	0.58
11	0.31	0.88
12	0.27	0.56
13	0.3	1.33
14	0.36	3.34
15	0.28	0.61
16	0.3	0.86
17	0.42	0.69
18	0.27	1.06
19	0.95	9.84
20	0.3	2.5
21	0.31	5.11
22	0.33	8.33
23	0.28	1.2
24	0.3	1.13
25	0.27	0.44
26	0.27	0.55
27	0.28	0.59
28	0.3	4.55
29	0.3	1.19
30	1	0.74
Avg.	0.35	2.61

第 6 章 总结与展望

本章对本论文的内容进行了总结，分析了本论文的主要贡献与不足之处，并指出了今后研究的方向。

6.1 总结

护士排班问题是一个具有现实意义的题目。良好的排班方案有助于鼓舞团队士气，营造良好的工作氛围，提高工作效率，从而对护理质量提供有力的保证，进而保障病人的健康与安全。而基于计算机的自动化排班将可以提高排班的效率和质量，从而使得人力资源得到有效的利用。

护士排班问题是一个具有挑战性的题目。它具有各种各样的约束条件，涉及到不同班次的衔接、护士的技能等级要求和工作时长等方面，另外，在排班中还要考虑到护士本身的意愿，尽量提高他们对工作安排的满意程度。

本论文对护士排班问题的研究现状进行了综述，并对其展开了算法与模型的研究。

首先，本论文对一个已有的护士排班问题提出了一个两阶段的求解算法。该算法的第一个阶段通过分枝限界算法寻找可行的日夜分班方案，第二个阶段对每个可行的日夜分班方案通过模拟退火算法进行优化。本算法的特点是，把问题转化到图论的网络流模型上，通过求解最大流问题求得原问题的一个可行解，然后以该最大流为起点使用模拟退火算法对其进行优化，通过在残余网络上寻找回路来调整流量得到不同的最大流，以遍历原问题不同的解。在优化的过程中，本论文提出的基于调整最大流的邻域操作保证了搜索总在可行域内进行，提高了优化的效率。在寻找回路时，本论文在一般的宽度优先搜索和深度优先搜索框架上引入了随机因素以加强优化的效果。

然后，本论文在原问题上加入了避免护士降级工作的约束条件，并通过在原有模型上增加常数个变量与不等式，提出了一个满足新约束条件要求的改进模型。同时，本论文对原有的建模方法进行了分析，引入了行约束条件与列约束条

件的概念，指出原有的建模方法实质上是把排班表中每行可能的取值事先枚举出来，并通过预处理把所有行约束条件转化为一个权值附加在每个行模式上，从而简化了最终的数学模型，有利于对模型的求解，然而当列约束条件比较复杂时，使用原有的建模方法可能会得到比较复杂的数学模型。于是本论文对原有建模方法进行推广，提出同时把列模式枚举出来并且预处理所有的列约束条件，然后引入一组新的等式来保证行列模式选取的一致性，使得最终得到的数学模型是一个形式统一的单项选择规划模型，而不依赖于问题的约束条件。本论文进一步地引入了超级行、超级列的概念，指出可以根据具体问题的特点把无区别的行或者列合并为超级行或者超级列来处理，从而减小模型的规模，有利于对其进行高效的求解。具体到本论文所考虑的新约束条件，通过引入超级行的概念，模型的变量数从2^n缩减到不超过$(n/3)^3$，其中n是护士的人数，这是一个非常可观的改进。

最后，本论文使用30组标准测试数据进行了实验。实验结果表明，本文提出的两阶段求解算法对于所有测试数据都能求得最优解，同时平均运行时间较短。而对于新的约束条件，本论文通过对原有测试数据进行了适当的修改，分别使用有针对性的改进模型和更具有通用性的推广模型对其进行求解，实验结果表明，有针对性的改进模型的求解效率优于通用模型，但是通用模型的求解效率也是可以接受的，考虑到该模型的通用性，本论文认为该模型是有意义的。

6.2 今后的研究方向

在本论文提出的两阶段求解算法中，使用了模拟退火算法作为优化手段，其他优化方法有待研究；而在第一阶段的分枝限界算法中，更高效的估界函数和剪枝策略有待进一步的研究；另外，可以研究使用非精确的元启发式算法作为第一阶段的算法的效果；还可以尝试在每次优化时按照一定的准则来权衡该次优化操作的计算时间和求解质量。

而对于本论文提出的通用模型，由于其具有与问题本身无关的统一形式，因此针对该模型的有效的通用算法的研究将具有非常重大的意义，可以为护士排班问题提供一个通用的解决方案。

参考文献

[1] D. Warner, J. Prawda. A mathematical programming model for scheduling nursing personnel in a hospital. Management Science, 1972, 19(4): 411-422

[2] H. Miller, W. Pierskalla, G. Rath. Nurse scheduling using mathematical programming. Operations Research, 1976, 24(5): 857-870

[3] D. Warner. Scheduling Nursing Personnel According to Nursing Preference: A Mathematical Programming Approach. Operations Research, 1976, 24(5): 842-856

[4] W. Healy. Multiple Choice Programming. Operations Research, 1964, 12(1): 122-138

[5] E. Burke, P. Causmaecker, S. Petrovic, G. Berghe. Variable neighbourhood search for nurse rostering problems. Metaheuristics: computer decision-making. Norwell, MA, USA: Kluwer Academic Publishers, 2004. 153-172

[6] E. Burke, P. Causmaecker, G. Berghe. A Hybrid Tabu Search Algorithm for the Nurse Rostering Problem. Lecture Notes In Computer Science, 1998, 1585: 187-194

[7] E. Burke, P. Cowling. A memetic approach to the nurse rostering problem. Applied Intelligence, 2001, 15(3): 199-214

[8] F. Bellanti, G. Carello, F. Croce, R. Tadei. A greedy-based neighborhood search approach to a nurse rostering problem. European Journal of Operational Research, 2004, 153: 28-40

[9] E. Burke, T. Curtois, G. Post, R. Qu, B. Veltman. A hybrid heuristic ordering and variable neighbourhood search for the nurse rostering problem. European Journal of Operational Research, 2008, 188: 330-341

[10] G. Beddoe, S. Petrovic. Selecting and weighting features using a genetic algorithm in a case-based reasoning approach to personnel rostering. European Journal of Operational Research, 2006, 175: 649-671

[11]J. Bard, H. Purnomo. Preference scheduling for nurses using column generation. European Journal of Operational Research, 2005, 164: 510-534

[12]J. Bard, H. Purnomo. A cyclic preference scheduling of nurses using a Lagrangian-based heuristic. Journal of Scheduling, 2007, 10(1): 5-23

[13]J. Bard, H. Purnomo. A column generation-based approach to solve the preference scheduling problem for nurses with downgrading. Socio-Economic Planning Sciences, 2005, 39(3): 193-213

[14]K. Dowsland, J. Thompson. Solving a Nurse Scheduling Problem with Knapsacks, Networks and Tabu Search. The Journal of the Operational Research Society, 2000, 51(7): 825-833

[15]K. Dowsland. Nurse scheduling with tabu search and strategic oscillation. European Journal of Operational Research, 1998, 106: 393-407

[16]U. Aickelin, K. Dowsland. Exploiting problem structure in a genetic algorithm approach to a nurse rostering problem. Journal of Scheduling, 2000, 3(3): 139-153

[17]U. Aickelin, K. Dowsland. An indirect Genetic Algorithm for a nurse-scheduling problem. Computers & Operations Research, 2004, 31: 761-778

[18]U. Aickelin, P. White. Building Better Nurse Scheduling Algorithms. Annals of Operations Research, 2004, 128: 159-177

[19]J. Li, U. Aickelin, E. Burke. A Component-Based Heuristic Search Method with Evolutionary Eliminations for Hospital Personnel Scheduling. INFORMS Journal on Computing, 2009, 21(3): 468-479

[20]J. Li, U. Aickelin. The application of Bayesian optimization and classifier systems in nurse scheduling. Lecture Notes in Computer Science, 2004, 3242: 581-590

[21]E. Burke, G. Kendall, E. Soubeiga. A tabu-search hyperheuristic for timetabling and rostering. Journal of Heuristics, 2003, 9: 451-470

[22]U. Aickelin, J. Li. An estimation of distribution algorithm for nurse scheduling. Annals of Operations Research, 2007, 155: 289-309

[23]U. Aickelin, E. Burke, J. Li. An estimation of distribution algorithm with

intelligent local search for rule-based nurse rostering. Journal of the Operational Research Society, 2007, 58: 1574-1585

[24]M. Goodman, K. Dowsland. A grasp-knapsack hybrid for a nurse-scheduling problem. Journal of Heuristics, 2009, 15: 351-379

[25]E. Burke, P. Causmaecker, G. Berghe, H. Landeghem. The state of the art of nurse rostering. Journal of Scheduling, 2004, 7: 441-499

[26]B. Cheang, H. Li, A. Lim, B. Rodrigues. Nurse rostering problems--a bibliographic survey. European Journal of Operational Research, 2003, 151: 447-460

[27]邢文训, 谢金星. 现代优化计算方法(第2版). 北京: 清华大学出版社, 2005

作者简历

- 郭嵩山, 刘祖立, 刘曦, 涂德健. 国际大学生程序设计竞赛例题解（七）——中山大学 ICPC 集训队内部选拔赛试题（2005-2006）. 北京: 电子工业出版社, 2010
- 2008 年 ACM 国际大学生程序设计竞赛吉隆坡赛区金奖第四名, 哈尔滨赛区银奖

致 谢

由衷地感谢本人的导师郭嵩山教授，本文是在郭老师的悉心指导下完成的。在研究的开展和论文的撰写过程中，郭老师给予了本人莫大的支持和帮助。同时，郭老师辅导本人参加国际大学生程序设计竞赛，培养了本人解决问题的能力。此外，在日常的学习生活中，郭老师对本人非常关怀，教会了本人许多为人处世的道理。

感谢本科和研究生阶段的所有老师们，谢谢各位老师的教导和栽培，使本人牢固地掌握了学科知识，从而有能力完成论文的写作。

感谢信息科学与技术学院的所有领导和老师们对本人的关心和帮助。

感谢香港城市大学的林良才教授、李彦志助理教授、博士生谢振豪和秦虎，以及香港科技大学的博士生朱文斌，感谢各位在学术研究的道路上对本人提供的指引和帮助，本人从中学到了许多，也开阔了视野。

感谢林瀚讲师为本人提供了许多有意义的研究资料。

感谢室友刘祖立同学一直以来在学习、生活、科研与求职等道路上对本人的支持和帮助。

感谢本实验室的翁雨键、涂德健、梁志荣等同学，感谢师弟张钊毅，感谢各位与本人在研究课题上的有启发性的讨论。

感谢计算机软件与理论专业的全体同学们，感谢这个和睦融洽的大集体，使大家可以在其中快乐地学习与成长。

感谢中山大学 ACM/ICPC 集训队的全体成员，感谢各位志同道合的战友们与本人并肩作战，为本人留下了许多难忘的回忆。

特别要感谢父母多年来的培育，感谢父母无微不至的关怀与照料，父母是本人坚实的精神支柱，鼓舞着本人在成长的道路上不断的克服困难并取得进步。

感谢所有关心本人的亲人和朋友。

最后，感谢论文评审委员会的全体成员，感谢各位委员抽出宝贵的时间评审本论文并提出宝贵的意见。