COMPUTATIONAL INTELLIGENCE FOR OPTIMIZATION

用于最优化的计算智能

Nirwan Ansari Edwin Hou 著
李军 边肇祺 译

清华大学出版社
http://www.tup.tsinghua.edu.cn

用于最优化的计算智能

Nirwan Ansari Edwin Hou 著
李军 边肇祺 译

清华大学出版社

(京)新登字 158 号

内 容 简 介

本书从讨论组合优化中的基本问题——NP 问题入手，系统地讲述了近年来所发展起来的智能最优化的各种技术和方法，其中包括启发式搜索、Hopfield 神经网络、模拟退火和随机机、均场退火以及遗传算法等；并在此基础上，通过一些典型的应用问题，如旅行商问题、模式识别中的点模式匹配问题、通信和任务调度等问题进一步阐明以上一些基本方法怎样用来解决这些原来具有 NP 性质的困难问题。本书是作者在美国新泽西州理工学院多年讲授有关课程的基础上写成的。全书深入浅出，理论联系实际。为帮助学生掌握基本概念，提高学习能动性，各章编写了习题。

本书可作为通信、计算机、控制各专业的高年级学生和研究生学习有关课程的教材。它对于广大科研工作者也是一本很有实际价值的参考书。

Computational Intelligence for Optimization

Nirwan Ansari Edwin Hou

北京市版权局著作权合同登记号：01-98-0945

书　名：用于最优化的计算智能

作　者：Nirwan Ansari Edwin Hou 著　李军　边肇祺　译

出版者：清华大学出版社(北京清华大学学研楼，邮编 100084)

http://www.tup.tsinghua.edu.cn

印刷者：清华大学印刷厂印刷

发行者：新华书店总店北京发行所

开　本：787×1092 1/16　**印张**：8.75　**字数**：204 千字

版　次：1999 年 12 月第 1 版　　1999 年 12 月第 1 次印刷

书　号：ISBN 7-302-03635-7/TP・2019

印　数：0001～3000

定　价：18.00 元

译者的话

洪伟年(Nirwan Ansari)与侯瑞海(Edwin Hou)两位教授集多年教学和科研成果合著的《用于最优化的计算智能》一书,是反映近年来最优化技术发展的专著。此书取材新颖、内容丰富、结构合理、论述清楚,在有限的篇幅中对所有重要方法都进行了透彻的阐述,不仅对于研究生掌握有关概念和方法十分有益,对通信、计算机、控制等领域的科研和工程技术人员也很有参考价值。

最近数年来计算智能技术发展迅速,在此基础上,本书论述了最优化基础理论和实际应用。书中涉及的多种智能最优化方法均已成为人工智能、模式识别和其它有关领域行之有效的方法,并已与经典的统计模式识别技术和结构模式识别技术一起,构成了现代模式识别方法的理论基础。鉴于国内有关智能计算在最优化中应用的资料大多散见于各种学术期刊和会议文集,相信本书中译版的出版将会有助于这一学科的发展,并推动相关领域的教学、科研和工程实践。

为了便于读者根据不同的学习和工作需要较快地掌握有关的算法,著者采用了一种相对独立的模块式结构。书中第 2～6 章每章各介绍一种基本的方法,首先对有关概念给出严格而明确的定义,然后再深入浅出地展开对理论部分的论述。在第 7～11 章每章围绕一个典型问题,对各种方法的应用实例加以描述,并进行适当比较。这样的安排不仅给读者一个清晰的概念,而且引导读者对各种方法的特点进行深入的比较,从而学会针对实际问题的特点选择有效的解决方法。

本书从第 1 章到第 8 章以及第 10 章,第 11 章由李军翻译,第 9 章由边肇祺翻译,边肇祺并对全书进行了校正。书中难免有翻译不确切的地方,望请读者批评指正。

感谢两位作者在本书的翻译过程中给予的帮助。清华大学出版社为本书的出版做了大量工作,译者在此一并致谢。

译　者

1998 年 11 月

前　言

最优化在本质上是一门交叉学科。它对多个学科产生了重大影响，并已成为不同领域中很多工作都不可或缺的工具。传统最优化方法已经十分完善且有多部优秀教材出版，然而一些已被证明在解决全局最优化问题上行之有效的新方法又涌现了出来，例如神经网络、模拟退火、随机机、均场理论和遗传算法等。

本书为高级最优化技术的最新发展，特别是启发式搜索、神经网络、模拟退火、随机机、均场理论和遗传算法，提供一个技术性的描述，并着重强调它们的数学基础、实现方法以及实际应用。这一教材适用于电类、计算机类等专业一年级研究生的教学，也可作为工程技术人员、科研人员及其他专业人员的参考书。本书是过去五年中我们所授几门课程的结晶，也凝聚了我们在这一领域多项交叉学科研究的成果。在过去的十年里，上述高级最优化技术受到越来越多的重视，但是新书却很少出版。绝大多数新的最优化理论及其应用都散见于期刊杂志、技术报告和会议文集。这使得初学入门者难于进行系统学习。我们希望这本书能使读者对这些方法综合了解，从而填补科学文献上的这一空缺。

本书在内容安排上采用了每一章都相对独立的模块式结构，其中第 2～6 章每章覆盖了一项最优化技术的理论，而第 7～11 章则侧重于不同范畴里最优化技术的实际应用。每章既可单独学习，又可作为一个大的综合性题目来研究。

我们深深得益于新泽西州理工学院那些在过去的五年中学习了我们的研究生课程“高级最优化技术”和“人工神经网络”的研究生们，尤其是 G. Wang，J. Chen，D. Liu，A. Agrawal，Y. Yu，Z. Zhang，A. Arulumbalam，S. Balasekar，J. Li 和 N. Sezgin。他们这些年来的反馈和评论帮助我们形成了本书现在的样式。我们非常感谢 L. Fitton 对于编辑此书的帮助以及 Kluwer 学术出版社的 S. Rumsey 在手稿输入方面所提供的支持。最后，我们衷心地表达对 Kluwer 学术出版社 A. Greene 的谢忱，感谢他对我们完成此书的不断鼓励。

Nirwan Ansari
Edwin Hou

目　录

前言 …… Ⅵ

第 1 章　引言 …… 1
1.1　计算复杂度 …… 1
1.1.1　算法分析 …… 1
1.1.2　NP 完全问题 …… 3
1.2　最优化技术综述 …… 3
1.2.1　启发式搜索 …… 3
1.2.2　霍普费尔德神经网络 …… 4
1.2.3　模拟退火与随机机 …… 4
1.2.4　均场退火 …… 4
1.2.5　遗传算法 …… 5
1.3　本书的结构 …… 5
1.4　习题 …… 5

第 2 章　启发式搜索方法 …… 6
2.1　图搜索算法 …… 6
2.2　启发函数 …… 10
2.3　A′搜索算法 …… 12
2.4　习题 …… 13

第 3 章　霍普费尔德神经网络 …… 15
3.1　离散霍氏网 …… 16
3.2　连续霍氏网 …… 18
3.3　按内容联想的存储器 …… 19
3.4　组合最优化 …… 20
3.4.1　旅行商问题 …… 21
3.4.2　二分图问题 …… 23
3.4.3　N 皇后问题 …… 24
3.5　习题 …… 25

第 4 章　模拟退火和随机机 …… 27

4.1 统计力学和麦绰泼里斯算法 …… 27
4.2 模拟退火 …… 30
4.2.1 有限时间实现 …… 31
4.2.2 一个例子：TSP …… 33
4.3 随机机 …… 35
4.3.1 波尔兹曼机 …… 35
4.3.2 高斯机 …… 37
4.3.3 柯西机 …… 38
4.4 习题 …… 40

第5章 均场退火 …… 42
5.1 均场近似 …… 42
5.2 鞍点展开 …… 43
5.3 稳定性 …… 45
5.4 均场网的参数 …… 45
5.5 二分图示例 …… 47
5.6 习题 …… 48

第6章 遗传算法 …… 49
6.1 简单遗传操作 …… 49
6.1.1 繁殖 …… 49
6.1.2 交叉 …… 51
6.1.3 突变 …… 51
6.2 一个示例 …… 52
6.3 为什么遗传算法会奏效？ …… 54
6.4 其它遗传操作 …… 56
6.5 习题 …… 57

第7章 旅行商问题 …… 58
7.1 为什么霍氏网经常不能生成有效解答？ …… 58
7.1.1 霍氏网的动力学 …… 60
7.1.2 拉格朗日参数的另一种表达方式 …… 62
7.1.3 霍普费尔德的10城市问题 …… 63
7.2 应用启发式搜索算法求解 TSP …… 67
7.3 应用模拟退火算法求解 TSP …… 69
7.4 应用遗传算法求解 TSP …… 70
7.5 本征值分析概述 …… 71
7.6 连接矩阵的 λ_1 的推导 …… 72

7.7 习题 …… 73

第 8 章 电信 …… 74
8.1 卫星广播调度 …… 74
8.1.1 SBS 问题的神经网络表达 …… 74
8.1.2 SBS 问题的均场公式 …… 76
8.1.3 算法的参数 …… 77
8.1.4 临界温度 T_c …… 78
8.1.5 一个示例 …… 80
8.2 集成 TDMA 通信系统的数据吞吐量最大化 …… 81
8.2.1 多路复用方案与数据吞吐量 …… 81
8.2.2 数据吞吐量的最大化 …… 82
8.3 总结 …… 86
8.4 习题 …… 86

第 9 章 点模式匹配 …… 87
9.1 问题的表述 …… 87
9.2 模拟退火框架 …… 89
9.2.1 编码方案 …… 89
9.2.2 能量函数 …… 90
9.2.3 扰动规则 …… 90
9.2.4 接受规则 …… 90
9.2.5 冷却流程 …… 90
9.2.6 停止准则 …… 91
9.3 进化程序设计 …… 91
9.3.1 解空间的表示 …… 91
9.3.2 群体空间：规模、初始化及其利用 …… 91
9.3.3 适度函数 …… 91
9.3.4 繁殖 …… 91
9.3.5 遗传算子 …… 92
9.3.6 模拟结果 …… 94
9.4 总结 …… 96

第 10 章 多处理器调度 …… 97
10.1 模型与定义 …… 97
10.2 均场退火 …… 98
10.2.1 MSP 霍普费尔德能量函数 …… 98
10.2.2 均场近似 …… 100

10.2.3 MSP 的均场公式 …… 100
10.2.4 数值解法和仿真 …… 100
10.3 遗传算法 …… 102
10.3.1 符号串表示 …… 103
10.3.2 起始群体 …… 103
10.3.3 适度函数 …… 104
10.3.4 遗传操作 …… 104
10.3.5 完整的算法 …… 106
10.3.6 仿真结果 …… 107
10.4 习题 …… 108

第 11 章 作业调度 …… 109
11.1 调度的分类 …… 110
11.2 JSP 的遗传算法 …… 111
11.2.1 JSP 的编码方式 …… 111
11.2.2 调度的生成 …… 112
11.2.3 遗传操作 …… 114
11.3 仿真结果 …… 115
11.4 习题 …… 116

参考文献 …… 117

各词术语中英文对照表(以汉语拼音排序) …… 127

第1章 引 言

许多科学和工程问题都可以归结为有约束条件的最优化问题，并可用如下数学公式加以表达：

$$\min_{x\in S} f(x) \text{ 基于 } g(x) \tag{1.1}$$

其中 S 是解域，f 是价值函数，g 是约束条件集合。各种最优化问题可以根据 S，f 和 g 的特点加以分类。如果 f 和 g 都是线性函数，则(1.1)式所描述的是一个易于求解的线性最优化问题。否则，(1.1)式便成为较难解决的非线性最优化问题。线性规划是一类典型的线性最优化问题，其约束条件的形式为 $g(x)\geqslant 0$ 或 $g(x)=0$。线性规划问题可以用很简单的算法[125]求解，并能在有限步骤内求得最优解。然而，在工程和其它领域中遇到的问题有很多都属于"难于求解"的一类问题，无法使用确定性算法求解。例如旅行商问题(TSP)和各种调度问题等。

包括神经网络、模拟退火、随机机、均场理论和遗传算法在内的各种最优化新方法的发现和发展，使得更有效地求解那些可以良好定义的难题成为可能。本书就以这些新的最优化方法和它们的应用为主题。

1.1 计算复杂度

在很多情况下，一个最优化问题可以用许多方法加以解决，而每种方法又能够采取多种算法予以实现。由于所有这些算法全都可能给出同一最优解，人们需要根据求得最优解的效率对它们进行分类。当然，一个最优化问题也有可能根本难以解决，因而不存在有效的寻求最优解的方法。计算复杂度理论就是用于研究算法有效性和问题难度的手段[57][94][110]。这一节主要介绍有关计算复杂度理论的一些基本概念。

1.1.1 算法分析

为便于有效地对算法进行真实比较，我们假定编译效率、程序语言以及计算机运算速度等因素对于有关算法来说都是完全相同的。

算法所用时间或空间随问题变大的增长率是衡量算法有效性的一个重要尺度。问题的"大小"是以输入数据量来计算的，例如旅行商问题中城市的数目。算法占用的时间和空间，亦即算法的时间复杂度和空间复杂度，则取决于算法用于求解问题所需的计算机运行时间和存储空间。研究一个算法随问题的规模增长时的极限性能，即渐进时间/空间复杂度，也是很有意义的。

我们可以定义一个以数据量为输入，时间/空间占据量为输出的函数，用来度量算法的量化时间/空间复杂度。例如，需要求解的问题是将一组整数由小到大排序。解决手段

之一是利用冒泡排序算法[10]比较相邻两数并进行适当交换。冒泡排序算法在最坏情况下的时间复杂度可以表示为

$$f(n)=\frac{n(n-1)}{2}=\frac{n^2}{2}-\frac{n}{2} \tag{1.2}$$

其中 n 是被排序整数的个数。

下表所列正是冒泡排序算法在不同输入数据量下的时间复杂度，它显示出每当输入量增加一个数量级时，冒泡排序算法的时间复杂度大致增加两个数量级。换句话说，时间复杂度以 n 的二次指数增长，对应于(1.2)式函数中的二次项 $n^2/2$。

n	1	10	100	1000	10000	10^5
$f(n)$	0	45	4950	499500	5.0×10^7	5.0×10^9

由于算法的渐进复杂度是我们关注的对象，余项 $-n/2$ 可以忽略不计，而二次项中的常数因子 1/2 也可省去，因此，冒泡排序算法的时间复杂度为 n^2 数量级，或记为 $O(n^2)$。

定义 1.1 若 f 和 g 是定义在正整数域上的正函数，$f,g:I^+\to R^+$，则

(1) $f=O(g)$，如果存在正常数 c 和 N，对所有的 $n\geqslant N$ 都使得 $f(n)\leqslant c\cdot g(n)$。

(2) $f=\Omega(g)$，如果存在正常数 c 和 N，对所有的 $n\geqslant N$ 都使得 $f(n)\geqslant c\cdot g(n)$。

(3) $f=\Theta(g)$，如果存在正常数 c,d 和 N，对所有的 $n\geqslant N$ 都使得 $d\cdot g(n)\leqslant f(n)\leqslant c\cdot g(n)$。

例如，(1.2)式中有

$$\frac{n(n-1)}{2}\leqslant n^2,\text{其中 } n\geqslant 1$$

因此可有 $c=1$ 和 $N=1$，使得 $f=O(n^2)$。

在很多情况下，算法的复杂度为下列函数之一：

$$O(\log n),O(n),O(n\log n),O(n^2),O(2^n),O(n^{\log n}),O(n!),O(n^n)$$

要注意的是，上述函数是按复杂度增大排列的。

表 1.1 时间复杂度函数的比较

时间复杂度函数	输入量 n				
	10	20	30	40	100
n	10 ns	20 ns	30 ns	40 ns	100 ns
$n\log n$	10 ns	26.0 ns	44.3 ns	64.1 ns	200 ns
n^2	100 ns	400 ns	900 ns	1.6 μs	10 μs
2^n	1.0 μs	1.0 ms	1.1 s	18.3 min	4.0 世纪
$n!$	3.6 ms	77.1 年	8.4×10^{13} 世纪	2.6×10^{29} 世纪	3.0×10^{139} 世纪

表 1.1 在假定所用计算机每秒可以执行 10 亿次运算的前提下，比较了几个方程在不

同输入量条件下的时间复杂度。很明显，如果一个算法的复杂度是指数函数（例如 2^n 或 $n!$），它在大输入量情况下是无望求解的。当一个算法的时间复杂度方程可以用多项式方程来划界时，我们称之为多项式时间算法；否则，它是一个指数时间算法。

1.1.2 NP 完全问题

如上所述，多项式时间算法比指数时间算法易解得多。不过总是有一些最优化问题无法用多项式时间算法求解。这些问题被称为难解问题。事先了解一个问题是否为难解问题自然会有很多好处。NP 完全理论的建立就是为了确定问题的难解度，并检验各个问题在难易程度上的相互关系。这一 NP 完全理论的基础是由库克(Cook)[46]确立的。

(1) 如果一个问题 P 可以被一个多项式时间算法求解，而另一个问题 Q 可以在多项式时间内被变换为 P，则一定存在一个多项式时间算法可以求解 Q。其中在多项式时间内将一个问题转换到另一个问题的过程称为多项式时间约化。

(2) 决策问题(答案为"是"或"否"的一类问题)中可在多项式时间内用非确定性计算机① 解出的特定类型被称为 NP。

(3) NP 中的任一问题都可以被多项式约化为 NP 中的一个特定决策问题，即满意度问题。因此，如果可以找到一个多项式时间复杂度算法来解决这一满意度问题，则 NP 中的任一问题都可在多项式时间内得解。另一方面，如果 NP 中任一问题是难解问题，则此一满意度问题也是难解问题。满意度问题可被认为是 NP 中"最难"的问题。

(4) 存在一组使 NP 中的其它问题以多项式方式约化于它们的"最难"问题。

有很多优化算法，诸如旅行商问题、多处理器任务调度问题等，都被发现可以多项式约化为 NP 中最难的问题。这些问题构成了 NP 完全问题类。在文献[57]中可以查到它们。究竟 NP 完全问题是否难解仍然是个谜。虽然很多线索表明 NP 完全问题确为难解，但至今未见严格的证明或证伪。不过，如果可以表明一个问题属于 NP 完全问题类，它多半是难于求解的。

1.2 最优化技术综述

这一节简要介绍各种最近发展起来的最优化新方法。

1.2.1 启发式搜索

启发式搜索最初是作为人工智能中问题求解程序的搜索器而被开发出来的[117]。在多数情况下，待解问题可以用状态空间加以表达，并用状态空间或图搜索算法求解。宽度优先和深度优先等图搜索算法的应用范围有限。不过，与待解问题相关的(启发)信息一般来说是可获取并运用于搜索过程。这使得最佳优先算法通常会有较好的表现。A* 算法[73]适于求解状态空间中从起始节点到终止节点的最小代价路径。它依据代价函数来选择下一个被搜索或扩展的节点。代价函数计算出从起始节点到当前节点路径的代价，并用启发知

① 非确定性计算机是一类计算模型，例如非确定性图灵(Turing)机。

识估计出到终止节点的余下路径的代价。在满足"可容性"的条件下，如果存在最小代价路径，A*一定可以找到。

1.2.2 霍普费尔德神经网络

自从霍普费尔德(Hopfield)揭示了他所提出的计算网络的运算能力[84]以来，大量的工作被投入霍普费尔德神经网络(简称霍氏网)的分析和应用。虽然最初的网络结构有它的弱点，霍普费尔德开创性的成果仍旧在某种程度上打通了一条神经网络的复兴之路。霍氏网的收敛性和稳定性已经被证明。加以适当改进后，霍氏网可用于求解旅行商问题等许多有约束条件的最优化问题。

将一个有约束条件的最优化问题抽象为可由霍氏网解决的形式，需要

(1) 将代价函数和约束条件转换为霍普费尔德能量函数；

(2) 确定拉格朗日(Lagrange)参数。

上述步骤往往是成功应用霍氏网的关键。许多研究人员未能复现霍普费尔德求解旅行商问题的结果，毛病经常就出在这里。在后续章节中，我们将讨论霍氏网的稳定性及其应用。

1.2.3 模拟退火与随机机

模拟退火算法最初由科克派特里克(Kirkpatrick)、小哥拉特(Gelatt Jr.)和瓦克奇(Vecchi)发明[98]。它模仿了液体结晶的过程：

(1) 在高温下，粒子能量较高，可以自由运动和重新排序；

(2) 随着温度的下降，粒子能量减弱，运动减少；

(3) 粒子最终进入平衡态，固化为具有最小能量的晶体。

模拟退火算法需要两个主要操作：一个是热静力学操作，用于安排降温过程；另一个是随机张弛操作，用于搜索在特定温度下的平衡态。模拟退火算法的长处在于它具有跳离局部最优解的能力。在给定温度下，模拟退火算法不但进行局部搜索，而且可以在一定概率下"爬山"到代价更高的解答以防止搜索进程陷入局部最小解。应用模拟退火算法以及各种不同的统计特性于霍氏网，可以得到波尔兹曼(Boltzmann)、高斯(Gaussian)、柯西(Cauchy)等随机机。类似于霍氏网的是，能量函数的拉格朗日参数对随机机的性能有重要影响。

1.2.4 均场退火

为了能以比随机机中的随机张弛操作更快的速度达到热平衡态，神经元的值可以由它们的平均值来加以近似。这时，在每一温度下，对神经元施加的是一个确定性的而不是随机性的操作。与随机机相比，这一网络只需较少的迭代次数便可在每一温度下达到热平衡态。在退火过程中，这一近似方法被称为均场退火。它在性能和计算复杂度二者之间作了一个很好的折衷。

1.2.5 遗传算法

遗传算法是一种模拟自然选择和遗传的随机搜索算法。它由贺兰德(Holland)提出[83],最初用于研究自然系统的适应过程和设计具有自适应性能的软件。遗传算法的基本形式要求有一个染色体表示用来对参数空间编码、一个适度函数用于估价所有的染色体、一组遗传操作用于产生新的染色体和一组概率用于控制遗传操作。遗传算法已被非常成功地应用于解决许多最优化问题并越来越流行。

1.3 本书的结构

本书分为两部分,其中第2~6章介绍5种先进优化算法的背景和理论,第7~14章描述这些算法在不同工程领域里的实际应用。具体而言,第2章讨论以A*算法为主的启发式搜索方法,包括图表示、图搜索算法、启发式算法、A*算法和可容性。第3章描述霍普费尔德神经网络,并讨论如何将一个有约束条件的最优化问题转化为一个可用霍氏网求解的无约束条件的最优化问题。第4章从最初的麦绰泼里斯(Metropolis)算法入手,沿着模拟退火算法的发展途径叙述了用模拟退火取代霍氏网中的确定性规则从而推导出随机机的各种方法。第5章描述如何利用神经网络中神经元的热均值来近似模拟退火的随机过程。这一称为均场退火的近似方法提供了寻找最优解的有效方法。第6章深入探讨遗传算法,包括遗传算法的简介、遗传操作及其分析等。第7章讨论旅行商问题,并给出分别用启发式搜索、模拟退火算法、霍氏网和遗传算法求解的方法。第8章以均场退火在卫星广播调度和集成TDMA通信系统数据吞吐量最大化中的应用为例,描述优化算法在远程通信中的应用。第9章讨论计算机视觉中十分重要的点模式匹配问题,并给出基于模拟退火和遗传算法的求解方法。第10章讨论多处理机系统中的任务调度问题及其基于均场退火和遗传算法的两种不同求解方法。第11章讨论制造系统中的作业调度问题。这一章还要讨论这一问题基于遗传算法的求解方法,并与传统方法加以比较。

本书采用模块式结构,第一部分中每章都可单独学习,然后可从第二部分有关应用的章节中获取更多的技术细节。

1.4 习题

1.1 证明:若 $f_1=O(g_1)$ 且 $f_2=O(g_2)$,则

$$f_1+f_2=O(g_1+g_2)。$$

1.2 证明:若

$$\lim_{n\to\infty}\frac{f(n)}{g(n)}=a,\ 0<a<\infty,$$

则 $f=O(g)$。

第2章　启发式搜索方法

启发式搜索方法依靠任务无关信息来简化搜索进程。在很多情况下，问题求解可视为系统化地构造或查找解答的过程。因此，在人工智能领域中，开发出了用于计算机问题求解的各种不同搜索算法[117]。尽管在某些情况下并非十分有效，迷宫、定理证明、国际象棋等问题都可以由计算机程序给出解答。一般来说，搜索过程包括检查搜索空间、估价可能有解的各种不同路径、记录已经搜索到的各个路径。

为简化搜索并减少搜索过程中时有出现的大量可选路径，从与待解问题有关的信息中所得到的启发知识或"经验法则"可用来确定搜索的方向。精心选择的启发信息可在很大程度上加大找出解答的机会，且在同时减少求解所需的时间。这一章统观启发式搜索方法并着重介绍 A* 算法。

在如图 2.1 所示的 3×3 九宫格棋盘上，摆放着 8 个将牌，上面不重复地刻着从 1 到 8 这 8 个数码。棋盘上余下一个空格使得其相邻将牌可以移动到它上面。设法找到一个将牌移动序列，用最少次数的移动将给定的起始布局转换为给定的终止布局，这就是 8 数码问题。实际上，记录空格的移动比记录将牌的移动方便得多。

$$\begin{bmatrix}1&2&3\\8&4&5\\7&&6\end{bmatrix}\xrightarrow{\text{右移}}\begin{bmatrix}1&2&3\\8&4&5\\7&6&\end{bmatrix}\xrightarrow{\text{上移}}\begin{bmatrix}1&2&3\\8&4&\\7&6&5\end{bmatrix}\xrightarrow{\text{左移}}\begin{bmatrix}1&2&3\\8&&4\\7&6&5\end{bmatrix}$$

起始布局　　n_1　　n_2　　终止布局

图 2.1　8 数码问题

在求取最小移动次数时，可用对各个布局的评价作为代价函数，从而把 8 数码问题变成一个最小代价问题。代价函数可被分为两项：一项是至今已经付出的代价；另一项则是基于启发信息估计出的到达目标尚需付出的代价。在代价函数中引入适当选择的启发信息(见 2.2 节)，计算机搜索程序能够有效地解决 8 数码问题。

2.1　图搜索算法

图是构造搜索空间、开展搜索过程的便利工具。

定义 2.1　图 $G=(V,E)$ 是一个二元组，其中 $V=\{v_1,v_2,\cdots,v_n\}$ 是节点的有限集，$E=\{(v_i,v_j)\subseteq V\times V\}$ 是连结 V 中节点的连线集。

一个图可为有向图亦可为无向图。有向图的连线是有方向性的，因而从 v_i 到 v_j 的连线 (v_i,v_j) 与从 v_j 到 v_i 的连线 (v_j,v_i) 是不同的。但在无向图中，连线 (v_i,v_j) 与 (v_j,v_i) 是相同的。图 2.2 是有向图 $G=(\{v_1,v_2,v_3,v_4,v_5\},\{(v_1,v_2),(v_1,v_4),(v_2,v_5),(v_3,v_2),(v_3,v_4),(v_4,v_3)\}$ 的图示。

定义 2.2 在有向图 $G=(V,E)$ 中，若 $v_i,v_j\in V$ 且 $(v_i,v_j)\in E$，亦即存在连线 (v_i,v_j)，则节点 v_j 是节点 v_i 的子节点或称后辈节点。同样，v_i 是 v_j 的父节点或称前辈节点。

定义 2.3 树是一类除根节点外的每个节点都只有一个父节点的图。

如果我们从图 2.2 所示有向图中去掉两根连线 (v_3,v_2) 和 (v_3,v_4)，则所余下的就是一个树。在搜索过程中，树被用来记录已展开的路径。

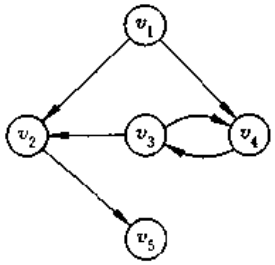


图 2.2 有向图示例

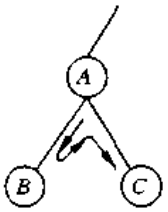


图 2.3 回溯示例

多数情况下，有待考察的图是一个有向图，通常还有唯一的一个起始节点。搜索过程不断生成后辈节点并把它们加入图中。这一过程被称为节点展开。搜索持续进行，直到它找到属于终止节点集的任一节点。以图的形式记录搜索过程的一个重要原因是为了便于回溯。回溯使得搜索过程得以废弃任何失败的路径，从而重新开拓一条崭新的路径。以图 2.3 所示搜索图为例，如果搜索过程到达节点 B 但它却似乎无助于求解，则搜索过程可以回溯到父节点 A 并从节点 C 继续。

一个路径是否有助于求解通常决定于代价函数。它对一个节点的评价基于至今已经花费的代价和其通往目标节点的可能性。代价函数可与图中的连线一起被用来表达搜索过程中从一个节点到另一个节点的代价。

定义 2.4 $cost(v_i,v_j)$ 是节点 v_i 与节点 v_j 间连线的代价。

定义 2.5 节点序列 $v_1\rightarrow v_2\rightarrow\cdots\rightarrow v_k$ 是从节点 v_1 到节点 v_k 的长度为 k 的路径。其中对于 $j=1,2,\cdots,k-1$，有 v_{j+1} 为 v_j 的子节点。

一条路径的代价也就是从起始节点经由如下公式列出的中间节点到达目标节点的路径上所有连线代价之和，即

$$cost(n_1\rightarrow\cdots\rightarrow n_k)=\sum_{j=1}^{k-1}cost(n_j,n_{j+1}) \tag{2.1}$$

因此，要评价节点 n，我们可以建立一个代价函数 $f(n)$，并使其包含两个组成部分：其一是从起始节点到 n 的实际代价；其二是从 n 到终止节点的代价估值。下面给出的基于 $f(n)$ 的图搜索过程 Graph-Search 试图找到一个从起始节点到终止节点的最小代价路径。

Graph-Search 算法

输入：起始节点 s 和一组终止节点 $\{t_i\}$。

输出：图 G 和树 T。

(1) 初始化。以起始节点 s 构造图 G 和集合 $OPEN$，$G\leftarrow\{s\}$，$OPEN\leftarrow\{s\}$，且令集合

CLOSE 为空集。

(2) LOOP：若 *OPEN* 为空集，算法以失败结束。

(3) 从 *OPEN* 中取出第一个节点 n，并使 $OPEN \leftarrow OPEN-\{n\}$，$CLOSED \leftarrow CLOSED \cup \{n\}$。

(4) 若 $n \in \{t_i\}$，算法以成功结束。

(5) 展开 n 并令 M 为 n 的子节点且不是其前辈节点的节点集。将 M 加入 G。

(6) 将 M 中既不在 *OPEN* 中也不在 *CLOSED* 中的节点放入 *OPEN*。

(7) 给每个 M 中既不在 *OPEN* 中也不在 *CLOSED* 中的节点设置一个指向其父节点的逆向指针。

(8) 调整 M 中那些同时也在 *OPEN* 或 *CLOSED* 中的节点的逆向指针。

(9) 调整 M 和 *CLOSED* 中每个节点的后继节点的逆向指针。

(10) 依据评价函数值重新安排 *OPEN* 中的节点。

(11) 回到 LOOP。

上述算法中所用的逆向指针就构成了解树 T，而回溯则是通过将节点分别放入 *OPEN* 和 *CLOSED* 两集合中而实现的。如果一个节点尚未被展开，亦即它的子节点还未被生成，则将它放入 *OPEN*；否则，一个已被展开的节点将被放入 *CLOSED*。图 2.4 示范出一个执行 Graph-Search 算法历经 3 次循环的例子。

循环 1

步骤(1)：最初 $OPEN=\{s\}$，*CLOSED* 为空集。见图 2.4(a)。

步骤(3)：s 被展开，*OPEN* 为空集，$CLOSED=\{s\}$。

步骤(5)：假定展开 s 生成 3 个子节点，$M=\{a,b,c\}$。见图 2.4(b)。

步骤(6)：$OPEN=\{a,b,c\}$。

步骤(7)：设置逆向指针。见图 2.4(c)。

循环 2

步骤(3)：假定 a 被选来扩展，$OPEN=\{b,c\}$，$CLOSED=\{s,a\}$。

步骤(5)：假定扩展 a 生成三个子节点，$M=\{b,d,e\}$，见图 2.4(d)。

步骤(6)：$OPEN=\{b,c,d,e\}$。

步骤(7)：设置逆向指针。见图 2.4(e)。注意 b 在此之前已有逆向指针指向 s，这种情况下不作调整。

循环 3

步骤(3)：假定 b 被选来扩展，$OPEN=\{c,d,e\}$，$CLOSED=\{s,a,b\}$。

步骤(5)：假定扩展 b 生成一个子节点，$M=\{e\}$。见图 2.4(f)。

步骤(6)：$OPEN=\{c,d,e\}$。

步骤(7)：设置逆向指针。见图 2.4(g)。注意 e 在此之前已有反向指针指向 a，这种情况下要调整为指向 b。

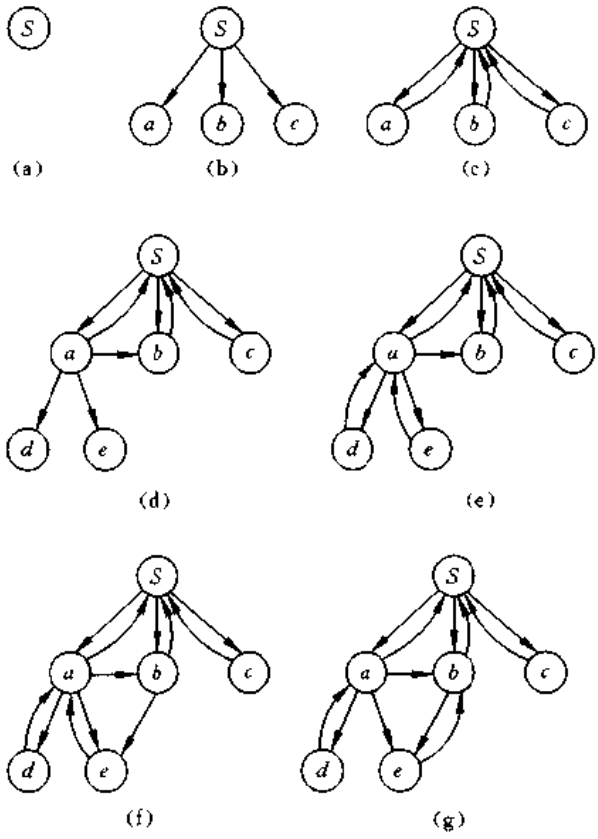


图 2.4　搜索图

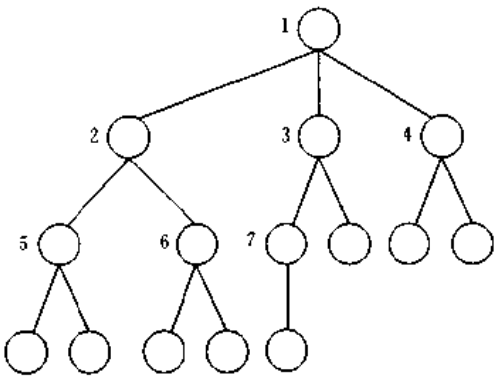


图 2.5　宽度优先搜索

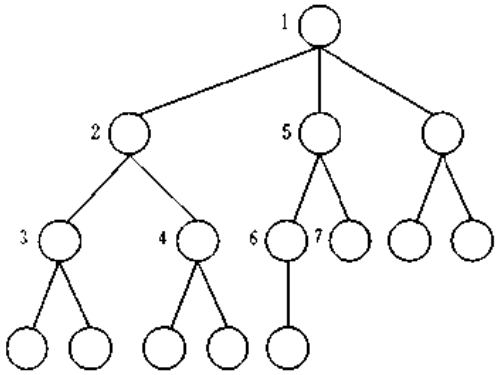


图 2.6　深度优先搜索(深度限制=2)

搜索的成功取决于选择下一个被扩展节点的选取，也受 Graph-Search 算法中步骤(10)的控制。这一步骤依据代价函数值来决定 *OPEN* 中节点的顺序。如果代价函数只考虑从起始节点到当前节点的代价，这一搜索称为无信息搜索。这一情况通常意味着无法获取从当前节点到终止节点可能代价的有关附加信息。另外，如果连线的代价是一个常数，例如 1，则代价函数可被定义为搜索树的深度。在此前提下，如果 *OPEN* 中的节点是按代价函数值由小到大排序，就得到宽度优先搜索。宽度优先搜索遍历搜索整个搜索空间。如

果 $OPEN$ 中的节点是按代价函数值由大到小排序，就得到深度优先搜索。图 2.5 和 2.6 显示出宽度优先搜索与深度优先搜索的不同。

2.2 启发函数

启发式方法，或更一般地，经验规则，是指引搜索方向以有效求取到达目标节点路径的原则或规则。一般来说，所用的启发知识在很大程度上取决于待解问题和节点表示。例如，可用于求解前述 8 数码问题的一个很好的启发信息就是不在最终位置上的离家将牌个数，记为 $Misplaced(n)$。对于图 2.1 所示初始将牌布局，将牌 4，5，6 离家，因而 $Misplaced(n)=3$。

在图搜索中，启发知识的数值表示为从当前节点到目标节点的代价。它构成了如下代价函数的一部分：

$$f(n)=g(n)+h(n) \tag{2.2}$$

其中，$g(n)$是从起始节点到当前节点 n 代价的最小估值，$h(n)$为从当前节点 n 到终止节点代价的估值。显然，$g(n)$可用至今已被展开的从起始节点到当前节点 n 的所用路径中的最小代价来近似。然而，从 n 到任一目标节点的路径都是未知的。为此，无法精确确定 $h(n)$而只能对其加以估算。在此我们规定目标节点的启发函数值为零，即对于 $n\in\{t_i\}$，$h(n)=0$。

在无法获取启发知识亦即对于所有节点 n 都有着 $h(n)=0$ 的情况下，启发式搜索退化为宽度优先搜索。当其启发函数具有 $f(n)=g(n)+h(n)$这一特殊形式时，Graph-Search 算法成为 A 算法。

A 算法

输入：起始节点 s 和一组终止节点$\{t_i\}$。

输出：图 G 和树 T。

(1) 初始化。以起始节点 s 构造图 G 和集合 $OPEN$，$G\leftarrow\{s\}$，$OPEN\leftarrow\{s\}$，且令集合 $CLOSE$ 为空集。

(2) LOOP：若 $OPEN$ 为空集，算法以失败结束。

(3) 从 $OPEN$ 中取出具有最小 $f(n)$值的节点 n，并使 $OPEN\leftarrow OPEN\{n\}$，$CLOSED\leftarrow CLOSED\cup\{n\}$。

(4) 若 $n\in\{t_i\}$，算法以成功结束。

(5) 展开 n 且令 M 为 n 的子节点且不为其前辈节点的节点集。将 M 加入 G。

(6) 对每一个节点 $m\in M$：

① 如果 $m\notin OPEN$ 且 $m\notin CLOSED$，将 m 放入 $OPEN$。估计 $h(m)$并计算 $f(m)=g(m)+h(m)$，其中 $g(m)=g(n)+cost(n,m)$。

② 如果 $m\in OPEN$ 或 $m\in CLOSED$，将其逆向指针调整到给出最小 $g(m)$值的路径。

③ 如果 m 的逆向指针被调整且 m 在 $CLOSED$ 中，将其重新加入 $OPEN$。

(7) 回到 LOOP。

函数 $g(n)$,$h(n)$和 $f(n)$在最小代价路径上的值可基于下列定义得到。

定义 2.6 $g^*(n)$是从 s 到 n 的所有路径中的最小代价。

定义 2.7 $h^*(n)$是从 n 到一个目标节点的所有路径中的最小代价。

由 $g^*(n)$和 $h^*(n)$的定义可得

$$f^*(n) = g^*(n) + h^*(n) \tag{2.3}$$

其中 $f^*(n)$是从起始节点经过节点 n 到达一个目标节点的所有路径中的最小代价。

下面用图 2.7 给出的实例示范 g^* 和 h^* 的计算方法。有向图 $G=(V,E)$中,

$V=\{a,b,c,d,e,i,j,k,l\}$

$E=\{(s,a),(s,b),(s,c),(a,d),(b,d),(b,e),(c,e),$
$(d,i),(d,j),(d,k),(e,j),(e,k),(e,l)\}$

连线的代价由函数 $cost$ 给出,且起始节点为 s,目标节点为 $\{j,k\}$。

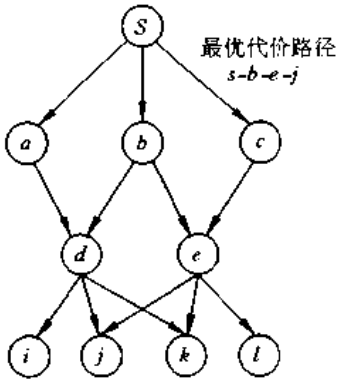


图 2.7 计算 g^*,h^* 和 f^*

计算 g^*,h^* 和 f^* 的过程可归纳如下:

(1) 从起始节点开始计算 g^*。显然,$g^*(s)=0$。

(2) 对其它任一节点 n,计算从 s 到 n 的每条路径的代价,而 $g^*(n)$是其中的最小值。

① 从 s 到 a 只有一条路径,所以 $g^*(a)=g(s\to a)=cost(s,a)=1$。

② 从 s 到 d 存在两条路径,即 $s\to a\to d$ 和 $s\to b\to d$。$g(s\to a\to d)=cost(s,a)+cost(a,d)=1-4=5$ 而 $g(s\to a\to d)=cost(s,b)+cost(b,d)=2+2=4$,所以 $g^*(d)=\min\{5,4\}=4$。

③ 从 s 到 j 存在 4 条路径,它们的代价分别为

$$g(s\to a\to d\to j)=1+4+3=8$$
$$g(s\to b\to d\to j)=2+2+3=7$$
$$g(s\to b\to e\to j)=2+1+1=4$$
$$g(s\to c\to e\to j)=3+3+1=7$$

因此,$g^*(j)=\min\{8,7,4,7\}=4$。

(3) 从目标节点开始计算 h^*,可得 $h^*(j)=h^*(k)=0$。

(4) 对其它任一节点 n,计算从 n 到 j 或 k 的每条路径的代价,而 $h^*(n)$是其中的最小值。例如,由 b 通往终止节点的路径及其代价可为

① $b\to d\to j$,$h(b\to d\to j)=cost(b,d)+cost(d,j)=2+3=5$;

② $b\to d\to k$,$h(b\to d\to k)=cost(b,d)+cost(d,k)=2+1=3$;

③ $b\to e\to j$,$h(b\to e\to j)=cost(b,e)+cost(e,j)=1+1=2$;

④ $b\to e\to k$,$h(b\to e\to k)=cost(b,e)+cost(e,k)=1+3=4$。

因此,$h^*(d)=\min\{5,3,2,4\}=2$。

一旦得到 $g^*(n)$和 $h^*(n)$,$f^*(n)$可由 $f^*(n)=g^*(n)+h^*(n)$求得。

对于图 2.7 所示有向图,$s\to b\to e\to j$ 是从 s 到目标节点之一 j 的最小代价路径,且代

价为 4。值得注意的是，在最小代价路径上的每个节点的 f^* 值都等于最小代价。当然，在实际求解过程中，图的结构在我们展开它之前是未知的。因此，g^* 只能基于至今已被展开的路径计算，而 h^* 则根本无法得知。

2.3 A*搜索算法

定义 2.8 A 算法在启发函数 $h(n)$ 满足

$$h(n) \leqslant h^*(n), \text{对所有节点 } n \tag{2.4}$$

的条件下被称为 A* 算法。

这也就是说 A* 算法的启发函数值永远低估了实际最小代价。A* 算法的重要意义在于，当 A* 算法结束时，它一定已经找到了一条从起始节点到目标节点的最小代价路径。关于 A* 算法的完整理论推导可参见文献[26]、[118]和[127]。这一节综述其中一些主要结论。

结论 2.1 若存在从起始节点到目标节点的路径，A* 算法一定会终止。

结论 2.2 A* 算法是可容(许)的。也就是说，如果存在一条从起始节点到目标节点的路径，A* 算法以发现最优解而终止。

结论 2.1 表明，如果存在从起始节点到目标节点的解答路径，A* 算法一定能找到它。结论 2.2 更进一步指出，A* 算法找到的解答路径一定具有从起始节点到目标节点的最小代价。

作为一个例子，在这里我们用 A* 算法解决以图 2.8 中起始节点 s 和终止节点 g 所给出的 8 数码问题。

$$s=\begin{bmatrix}2 & 8 & 3\\ 1 & 6 & 4\\ 7 & & 5\end{bmatrix} \qquad g=\begin{bmatrix}1 & 2 & 3\\ 8 & & 4\\ 7 & 6 & 5\end{bmatrix}$$

图 2.8 8 数码问题的起始布局和目标布局

在从一个布局转换为另一个布局时，付出的代价是一步移动。我们要找出的是从 s 到 g 的最少移动步数走法。正如在前一节中所讨论过的，我们以 $Misplaced(n)$ 亦即不在目标布局给出的位置上的离家将牌数为启发函数。图 2.9 示出了 A* 算法展开的节点及相应的 $f(n)$ 计算值。在展开了 6 个节点后，到达终止节点。

图 2.9 中，九宫格左侧的黑体数码表示节点展开的顺序，右侧的数码则是该节点的 f 值。最小代价路径由加重的连线表示。

虽然 A* 算法保证可以发现最优解，但它的性能与启发函数的选择有很大关系。如果启发函数严重低估实际的 h^*，算法退化为宽度优先搜索。如果问题规模较大，对用于保存搜索图和解树的资源要求也就相当高。

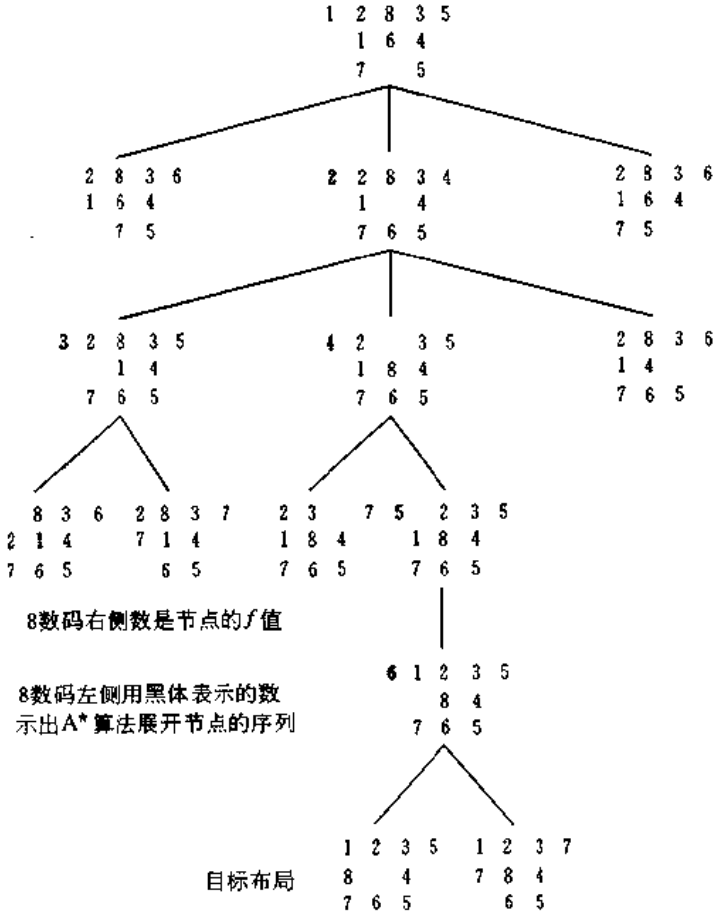


图 2.9　8 数码问题的结果

2.4　习题

2.1　考虑第 2.1 节中的 8 数码问题。给定起始节点为 $\begin{bmatrix}1 & 2 & 3\\8 & & 5\\7 & 6 & 4\end{bmatrix}$ 且终止节点为 $\begin{bmatrix}1 & 2 & 3\\8 & & 4\\7 & 6 & 5\end{bmatrix}$，用 A* 算法及下列各启发函数构造搜索图。

(1) $h(n)=0$。

(2) $h(n)=Misplaced(n)$，其中 $Misplaced(n)$ 为离家将牌数。

(3) $h(n)=seq(n)$，其中 $seq(n)$ 对居于正中位置的将牌为 1，而对于那些位于四周、其数码却又不按顺时针由小到大排序(跟在其后将牌数码不是比它刚好大 1)的将牌为 2。

2.2　计算下图 g^*，h^*，f^* 的值。

$$V=\{s,a,b,c,d,e,i,j,k,l,m\}$$

$cost(s,a) = 1$　　$cost(s,b) = 2$　　$cost(s,c) = 2$

$cost(a,d) = 3$　　$cost(a,i) = 4$　　$cost(a,j) = 7$

$cost(b,d) = 1$　　$cost(b,e) = 2$　　$cost(b,i) = 1$

$cost(c,i) = 3$　　$cost(c,m) = 5$　　$cost(d,k) = 2$

$cost(e,j) = 1$　　$cost(i,l) = 1$　　$cost(i,m) = 2$

2.3 考虑一个具有如下起始布局的积木游戏：

B	B	B	W	W	W	

其中有三个黑色将牌(B)，三个白色将牌(W)和一个空格。将牌可按如下规则移动：

(1) 将牌可以代价 1 移到相邻的空格上去。

(2) 将牌可越过最多两个其它将牌而跳到空格上去，且代价为其跳过的将牌数。

这一游戏的目标是使所有白色将牌都在任一黑色将牌的左侧(不管空格的位置在哪里)。为这个问题选择一个启发函数并给出搜索图产生过程。

2.4 8 皇后问题是要找出一个在 8×8 国际象棋盘上摆放 8 个皇后而又使它们互不威胁的布局(见第 3 章)。

(1) 设计一个表示 8 皇后问题布局的方案。

(2) 为 8 皇后问题定义一个可行的启发函数。

第3章　霍普费尔德神经网络

霍普费尔德对于神经网络的复兴作出了重大贡献。他的成果展示了模拟神经网络所具有的重要计算能力[84][85]。霍普费尔德网络可用于信息存取的联想式存储器，也可用于求解组合优化问题。它的网络输出反馈到网络前面层的输入，因而属于循环式神经网络[75]。霍普费尔德网络有一些不同的形式，图 3.1 是其基本结构。图中 $I_1, I_2, \cdots, I_N$ 是外部对网络的输入；$s_1, s_2, \cdots, s_N$ 是神经元对网络外部的输出；$u_1, u_2, \cdots, u_N$ 是网络中相应神经元的输入总量(激活能)；w_{ij}则是从第 j 个神经元的输出与第 i 个神经元的输入之间的连接权，且 $w_{ji}=w_{ij}, w_{ii}=0$；而 $\zeta(\cdot)$则是非线性的硬限函数或 Sigmoid 函数。

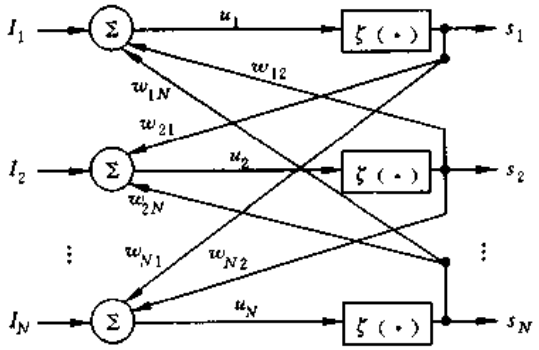


图 3.1　霍氏网的基本结构

图 3.2 示出了上述非线性函数的各种选择，其中限幅函数的定义如下：

二值硬限器：

$$\zeta(u_i) = \mathrm{sgn}_{T_i}(u_i) = \begin{cases} 1 & u_i > T_i \\ 0 & u_i < T_i \end{cases} \tag{3.1}$$

双极硬限器：

$$\zeta(u_i) = \mathrm{sgn}_{T_i}(u_i) = \begin{cases} 1 & u_i > T_i \\ -1 & u_i < T_i \end{cases} \tag{3.2}$$

而 Sigmoid 函数的定义则为

二值 Sigmoid：

$$\zeta(u_i) = \mathrm{sgm}_{T_i,\beta}(u_i) = \frac{1}{2}\left[1 + \tanh\left(\frac{u_i - T_i}{\beta}\right)\right] \tag{3.3}$$

双极 Sigmoid：

$$\zeta(u_i) = \mathrm{sgm}_{T_i,\beta}(u_i) = \tanh\left(\frac{u_i - T_i}{\beta}\right) \tag{3.4}$$

当选用硬限函数时，霍氏网成为二值或二极网络，亦即离散霍氏网；而当选用 Sigmoid 函

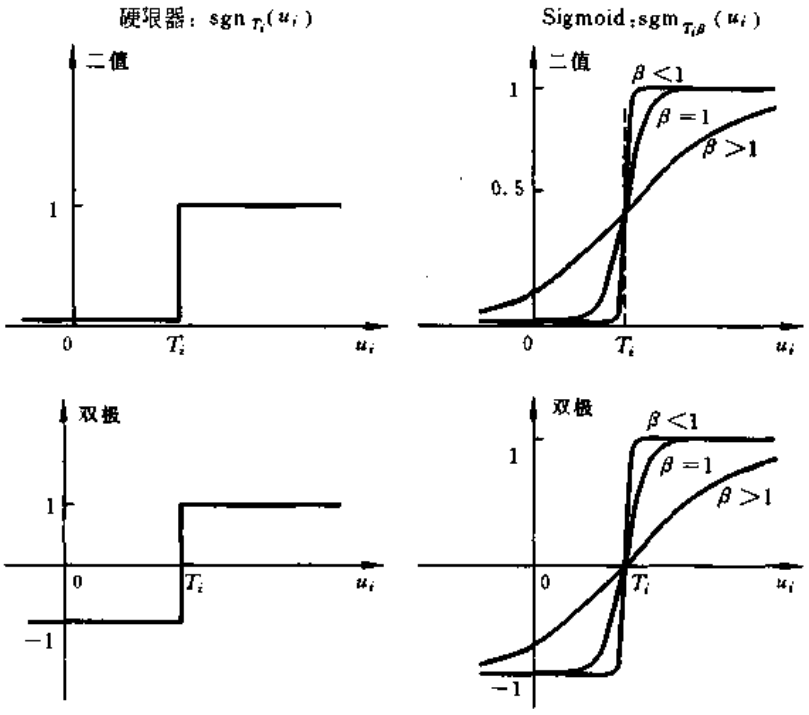


图 3.2　可供选择的各种非线性函数

数时，霍氏网是一个连续值网络。

3.1　离散霍氏网

“离散”一词在神经网络术语中通常是指神经元采用二值或双极的情况，但广义上也包括其它阶跃取值的神经元，例如三值神经元。以硬限函数取代图 3.1 中的非线性函数，即可得一离散霍氏网。在神经网络中，神经元更新既可同步进行亦可异步进行。在同步进行时，网络中所有神经元的更新同时进行，也就是

$$S^+ = \mathrm{sgn}_T\{WS^- + I\} \tag{3.5}$$

其中

$$T = \begin{bmatrix} T_1 \\ T_2 \\ \vdots \\ T_N \end{bmatrix},\ W = \begin{bmatrix} 0 & w_{12} & \cdots & w_{1N} \\ w_{21} & 0 & \cdots & w_{2N} \\ \vdots & \vdots & & \vdots \\ w_{N1} & w_{N2} & \cdots & 0 \end{bmatrix},\ S = \begin{bmatrix} s_1 \\ s_2 \\ \vdots \\ s_N \end{bmatrix},\ \text{且 } I = \begin{bmatrix} I_1 \\ I_2 \\ \vdots \\ I_N \end{bmatrix} \tag{3.6}$$

式中变量的上角标“+”表示该变量更新后的值；类似的，上角标“−”表示其更新前的值[①]。(3.6)式中权值矩阵 W 是对称的。当神经元的不断更新使得 $S^+ = S^-$ 时，网络收敛。不过，同步更新也可能将网络带到一个具有两个互补状态的动态平衡。在异步进行时，在同一时刻只有一个神经元更新而且这一神经元在网络中每个神经元都更新之前不会再次

① 本书在需要时即以这两个上角标表示变量值的相对更新(时标)。

更新。异步更新一定能使网络收敛。在异步更新时神经元的更新顺序可以是随机的，亦即

$$s_i^+ = \mathrm{sgn}_{T_i}(u_i) = \mathrm{sgn}_{T_i}\Big(\sum_{j\neq i} w_{ij}s_j^- + I_i\Big) \tag{3.7}$$

注意，当 $u_i=0$ 时 $s_i^+ = s_i^-$。霍氏网的分析类似于控制理论中非线性动力系统的分析。在动力系统分析中，其稳定性是至关重要的。这对霍氏网也不例外。

定义 3.1　若神经元 i 的值在后续更新中不再改变，则称其为稳定的。

定义 3.2　若霍氏网中所有神经元都是稳定的，则称其为稳定的。

霍氏网状态的特性与一个李雅普诺夫(Lyapunov)函数[①]有很大关系[89]。在神经网络术语中，这一李雅普诺夫函数被称为霍普费尔德能量函数。要想说明霍氏网是稳定的，只要表明其能量函数确为李雅普诺夫函数即可。(3.8)式就是霍普费尔德给出的霍氏网能量函数。

$$E(\boldsymbol{S}) = -\frac{1}{2}\sum_i\sum_{j\neq i} s_i s_j w_{ij} - \sum_i s_i I_i = -\frac{1}{2}\boldsymbol{S}^{\mathrm{T}}\boldsymbol{W}\boldsymbol{S} - \boldsymbol{I}^{\mathrm{T}}\boldsymbol{S} \tag{3.8}$$

我们可以容易地表明上式是满足李雅普诺夫函数的三个条件的：

(1) 从 $\dfrac{\partial}{\partial s_i}E(\boldsymbol{S}) = -\sum_{j\neq i} w_{ij}s_j - I_i$ 可以看出[②] $E(\boldsymbol{S})$对于所有 $\boldsymbol{S}$ 的分量是连续的。

(2) 严格地讲，(3.8)式并不满足李雅普诺夫函数的第二个条件。不过对于神经元有界的神经网络的稳定性而言，这一条件可以退化为只要求该函数是有界的。它与要求函数随时间递减的第三个条件合在一起，使得网络的动态过程最终停止于函数的下界处，达到稳定。因为 $\boldsymbol{W}$ 和 $\boldsymbol{I}$ 都是确定值向量，且 $\boldsymbol{I}$ 是有界的，E 有其下界。

一个这样的下界可以是　$E_{\min} = -\dfrac{1}{2}\sum_i\sum_j |w_{ij}| - \sum_i |I_i|$。

(3) 我们通过证明 s_i 的任何变化都使 E 下降来说明第三个条件也满足。在此以 ΔE 表示由霍氏网状态变化 Δs_i 引起的能量变化。由

$$\frac{\partial}{\partial s_i}E(\boldsymbol{S}) = -\sum_{j\neq i} w_{ij}s_j - I_i = -u_i \tag{3.9}$$

可得

$$\Delta E = \sum_i \Delta s_i \frac{\partial}{\partial s_i}E(\boldsymbol{S}) = -\sum_i \Delta s_i u_i \tag{3.10}$$

不失一般性，引入二值硬限函数且令其阈值为 0。则有

$$\Delta s_i = \mathrm{sgn}(u_i) - s_i = \begin{cases} 1 - s_i \geqslant 0 & u_i > 0 \\ 0 - s_i \leqslant 0 & u_i \leqslant 0 \\ 0 & u_i = 0 \end{cases} \tag{3.11}$$

因此，$\Delta s_i u_i \geqslant 0, \forall\, i$。故而 $\Delta E \leqslant 0$，霍普费尔德能量函数对于网络状态的任何变化都是下

---

① 如果存在一个平衡态 x^* 使得如下三个条件成立，函数 $f(x), x\in R^n$ 是一个李雅普诺夫函数。
(1) $f(x)$对于所有 x 是连续的。
(2) $f(x)$是正定的，亦即 $f(x^*)=0$ 且对 $x\neq x^*$ 有 $f(x)>0$。
(3) $f(x)$对时间的导数 $\dot{f}(x)$是半负定的，亦即该导数随时间递减。

② 若 $f: A\to R^n$ 在 $c\in A$ 可导，则存在严格正数 δ 和 K 使得当 $\|x-c\|<\delta$ 时有 $\|f(x)-f(c)\|\leqslant K\|x-c\|$。由此即可推论 f 是连续的。

降的。

在异步更新时，霍普费尔德能量函数作为一个李雅普诺夫函数单调下降直到进入稳定状态。在此状态下，无论网络状态还是能量函数都不再变化。证明霍普费尔德能量函数为李雅普诺夫函数只说明平衡态或稳定点的存在，例如联想存储器的一个回想模式，或者组合优化问题的一个解答。它并不保证这一回想模式或问题解答的最优性。

3.2 连续霍氏网

“连续”一词在霍氏网中指的是在图 3.1 用 Sigmoid 非线性时，神经元取[0,1]或[−1,1]间的连续值。另外，如图 3.3 所示在神经元的净输入与其它神经元的输出之间，积分器被用来引入时间延迟从而使激活 u_i 落后于刺激 s_i。事实上，图 3.3 只是连续霍氏网的一个功能性表示，而其原来的实现则是一个基于生物神经网络并由放大器、连线、电阻、电容等构成的反馈线路。在此，变量以类似于霍氏网的方式定义。从图 3.3 即可看出神经元变量的动态进程是以下列形式出现的：

$$s_i = \mathrm{sgm}_{T_i,\beta}(u_i) \tag{3.12}$$

$$\frac{\mathrm{d}}{\mathrm{d}t}u_i = \sum_{j\neq i} w_{ij}s_j + I_i \tag{3.13}$$

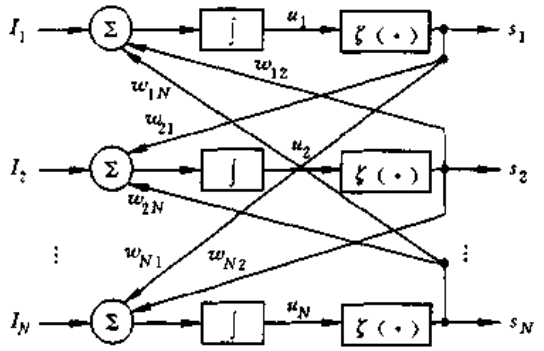


图 3.3 连续霍氏网的结构

与离散霍氏网相似的是，连续霍氏网也需要一个适当的霍普费尔德能量函数来保证其收敛于稳定状态。这只需将(3.8)式中的 s_i 由离散变为连续即可得到。注意，由此可得

$$\frac{\mathrm{d}}{\mathrm{d}t}u_i = -\frac{\mathrm{d}}{\mathrm{d}s_i}E \tag{3.14}$$

要想证明上述收敛条件成立，只需证明这一霍普费尔德神经网能量函数确为一个李雅普诺夫函数即可。与离散霍氏网的情况类似，证明该函数满足李雅普诺夫函数的前两个条件是很容易的事。第三个条件的满足则可用如下推导来说明：

$$\frac{\mathrm{d}}{\mathrm{d}t}E = \sum_i \frac{\mathrm{d}s_i}{\mathrm{d}t}\frac{\mathrm{d}E}{\mathrm{d}s_i}$$

$$= -\sum_i \frac{\mathrm{d}s_i}{\mathrm{d}t}\frac{\mathrm{d}u_i}{\mathrm{d}s_i}$$

$$=-\sum_i \frac{\mathrm{d}s_i}{\mathrm{d}u_i}\left(\frac{\mathrm{d}u_i}{\mathrm{d}s_i}\right)^2 \tag{3.15}$$

注意到 $\mathrm{sgm}_{\gamma,\beta}(u_i)$单调增长，亦即$\frac{\mathrm{d}s_i}{\mathrm{d}u_i}\geqslant 0$。所以，$\frac{\mathrm{d}}{\mathrm{d}t}E\leqslant 0$，也就是说 E 的时间导数是半负定的。

下面概述实现霍氏网神经元之间相互作用的迭代过程。

(1) 初始化。

(2) 在每个周期(扫描)[①] 重复下列步骤：

① 随机抽取一个在此周期中尚未更新的神经元。

② 计算

$$u_i^+ = u_i^- + \Delta t\left[-\frac{\mathrm{d}}{\mathrm{d}s_i^-}E\right]= u_i^- + \Delta t\left[\sum_{j\neq i} w_{ij}s_i^- + I_i\right] \tag{3.16}$$

其中 Δt 一般是一个很小的值。

③ $s_i^- = \zeta(u_i^+)$。

(3) 若到达稳定状态，停止；否则转回第(2)步，进入下一个周期。

最初的连续霍氏网[86]所用的是二极 Sigmoid 非线性，且其神经元间的相互作用和能量函数也稍有不同，即

$$C_i\frac{\mathrm{d}}{\mathrm{d}t}u_i = \sum_{j\neq i} w_{ij}s_j + I_i - \frac{u_i}{R_i} \tag{3.17}$$

$$E = -\frac{1}{2}\sum_i\sum_{j\neq i} s_i s_j w_{ij} - \sum_i s_i I_i + \sum_i \frac{1}{R_i}\int_0^{s_i}\mathrm{sgm}^{-1}(v)\mathrm{d}v \tag{3.18}$$

其中 sgm^{-1}代表 Sigmoid 函数的反函数，R_i 是来自一个放大器的输入电阻以及与此反馈线路中的放大器相连的其它电阻，C_i 则是该放大器及其相连导线的总电容。(3.18)式的最后一项在 Sigmoid 函数值大增益下由于接近限幅器可被忽略不计。可以证明公式(3.18)是一个李雅普诺夫函数而且一定收敛于稳定状态。

3.3 按内容联想的存储器

离散霍氏网最初是为设计联想式存储器(CAM)提出的。存储器是用于存储(写入)和恢复(读出)信息(数据)的介质。在传统数字计算机中，随机存储器(RAM)的信息存取是依靠地址(标签)来进行的。对应于此，CAM 中通常用向量表示的信息则由该向量本身、与其相似向量或另一向量来存取。用相似向量存取的方式被称为自相关，而用另一向量存取则为互相关。

作为一个 CAM 的霍氏网，表示信息的向量(模式)集合 $S^{(1)}, S^{(2)}, \cdots, S^{(p)}$ 以权值矩阵 W 的形式存储，且

① 在整个网络中对每个神经元进行一次更新即构成了一个周期或一次扫描。除非另有说明，一次扫描与一个周期是相同的。

$$w_{ij}=\begin{cases}(1-\delta_{ij})\sum_{k=1}^{p}(2s_i^{(k)}-1)(2s_j^{(k)}-1) & \text{对于二值限幅}\\ (1-\delta_{ij})\sum_{k=1}^{p}s_i^{(k)}s_j^{(k)} & \text{对于双极限幅}\end{cases} \tag{3.19}$$

其中：

w_{ij}是 $\boldsymbol{W}$ 中第(i,j)个元素；

$s_i^{(k)}$ 是 $\boldsymbol{S}^{(k)}$中的第 i 个元素；

δ_{ij}是克罗奈科尔(Kronecker)增量函数。

数据更新以(3.7)式给出的异步方式进行，但不计外部输入。如果一个与存储器矩阵中 $\boldsymbol{S}^{(k)}$最相近的向量输入网络，$\boldsymbol{S}^{(k)}$将是网络的输出。实际上，$\boldsymbol{S}^{(1)},\boldsymbol{S}^{(2)},\cdots,\boldsymbol{S}^{(p)}$是网络的稳定状态，并满足下列条件

$$s_i^{(k)}=\operatorname{sgn}\Big\{\sum_{j\neq i}w_{ij}s_j^{(k)}\Big\}\text{或}\ \boldsymbol{S}^{(k)}=\operatorname{sgn}\{\boldsymbol{W}\boldsymbol{S}^{(k)}\} \tag{3.20}$$

例 3.1 令 $\boldsymbol{S}^{(1)}=[1\quad 1\quad -1]^{\mathrm{T}},\boldsymbol{S}^{(2)}=[-1\quad -1\quad 1]^{\mathrm{T}}$ 为存储于一个 CAM 中的两个信息向量，其中上角标 T 表示向量的转置。由此，可据(3.19)式构造出权值矩阵 $\boldsymbol{W}$ 如下：

$$\boldsymbol{W}=\begin{bmatrix}0 & 2 & -2\\ 2 & 0 & -2\\ -2 & -2 & 0\end{bmatrix} \tag{3.21}$$

注意 $\boldsymbol{S}^{(1)}=\operatorname{sgn}\{\boldsymbol{W}\boldsymbol{S}^{(1)}\}$和 $\boldsymbol{S}^{(2)}=\operatorname{sgn}\{\boldsymbol{W}\boldsymbol{S}^{(2)}\}$是网络的两个稳定状态。如图 3.4 所示，其余 6 个网络状态中的任何一个都可在一次扫描中收敛于这两个状态中的一个。

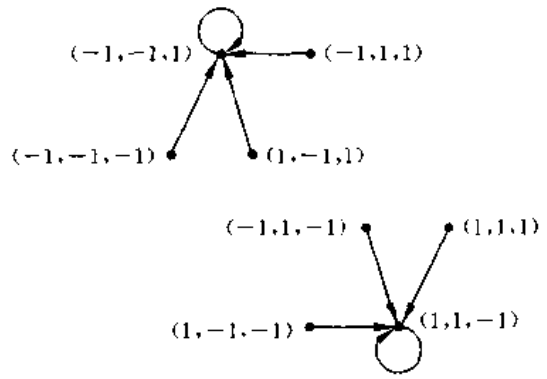


图 3.4 例 3.1 中 CAM 的状态转换

霍氏网的容量是一个很重要的参量，特别是将其用作CAM 时。据此可以确定网络中可以存入多少信息，或是网络可以分辨多少模式。霍氏网的尺度性质不好。也就是说，它的容量增长小于线性增长[8][107]。读者可以参考有关文献进一步了解增大容量的方法，例如使用三值神经元[24][25]，以及应用子网[36]。

3.4 组合最优化

霍氏网稳定状态的确定是运用迭代式的更新方式使霍普费尔德能量函数最小化而得

到的。只要一个最优化问题的约束条件和代价可被转化为霍氏网中适当的霍普费尔德能量函数，从而将这一问题映射到霍氏网的框架之中，霍氏网可以很自然地用于解决组合优化问题。下面我们通过为几个著名的问题构造霍普费尔德能量函数来展示霍氏网解决优化问题的能力。

3.4.1 旅行商问题

旅行商问题(TSP)可以如下表述：给定一组 N 个城市和它们两两之间的直达距离，找出一个闭合的旅程，使每个城市刚好经过一次且总的旅行距离最短。N 城市 TSP 的解答由访问这 N 个城市的一个次序表给出。要把这个问题映射到霍氏网的框架中去，只需用一个旅程表作为网络的终止状态。在霍普费尔德与坦客(Tank)[86]所采用的表示方法中，一个旅程的城市次序由一组神经元的终止状态代表。例如，对一个 N 城市问题，网络需要 N^2 个神经元，其中每个神经元代表一个城市在旅程表中的位置。因为有 N 个城市，每个城市又有 N 个可能的位置，所以要有 $N\times N$ 个神经元。一个途经 6 个城市(A,B,C,D,E,F)的 TSP 路径可被表示如下：

	1	2	3	4	5	6
A	0	1	0	0	0	0
B	0	0	0	1	0	0
C	0	0	0	0	1	0
D	1	0	0	0	0	0
E	0	0	1	0	0	0
F	0	0	0	0	0	1

在这个例子中，每一个城市都有 6 种可能的位置(从第一个被访问到最后一个被访问)，因此要用到 36 个神经元。在上面给出的网络状态(路径)中，D 市是第一个被访问的，A 市第二，…，而 F 市最后。这一路径总的旅程距离为 $d_{DA}+d_{AE}+d_{EB}+d_{BC}+d_{CF}+d_{FD}$，其中 d_{XY} 表示从 X 市到 Y 市的直达距离。总之，TSP 可被方便地用一个正方矩阵表示，矩阵中每个元素代表一个城市和它在旅程表上的位置。

霍普费尔德和坦客为 TSP 设计了如下能量函数以表示网络的状态：

$$\begin{aligned} E=&\frac{A}{2}\sum_X\sum_i\sum_{j\neq i}s_{Xi}s_{Xj}+\frac{B}{2}\sum_i\sum_X\sum_{Y\neq X}s_{Xi}s_{Xj}\\ &+\frac{C}{2}\Big(\sum_X\sum_i s_{Xi}-\tilde{N}\Big)^2\\ &+\frac{D}{2}\sum_X\sum_{Y\neq X}\sum_i d_{XY}s_{Xi}(s_{Y,i+1}+s_{Y,i-1}) \end{aligned} \tag{3.22}$$

其中 A,B,C 和 D 都是正常数[①](或将约束的优化问题转化为无约束的优化问题的拉格朗日参数)。下角标的取值范围由 N 限定，其中 X 或 Y 是一个城市的名称，而 i 则是该城市

① 注意不要将这些常数混同于城市名。

在一个旅程表中的位置。参数 $\tilde{N}$ 通常选用比城市数 N 稍大一些的值。(3.22)式的前三项对应于 TSP 的约束条件，而最后一项则对应于 TSP 的代价函数。假定网络到达了一个稳定状态，且在这一状态下所有神经元的取值均为"1"或"0"。当且仅当矩阵中每一行最多只有一个神经元处于激活状态(输出为 1)下，第一项才为零。换句话说，这一项只在每个城市被访问的次数不能多于一次的约束条件得到满足时才为零。第二项当且仅当每一列最多只有一个神经元处于激活状态时为零。也就是说，这一项只在旅程表中每个位置上不能有多于一个城市的约束条件得到满足时才为零。即使前两项都为零，也未必能得到合乎要求的解答，因为有可能所有神经元都处于抑制状态(输出 0)。为此，第三项被引入，并促使稳定状态至少给出符合 TSP 要求的完整旅程。这三项与最后一项合在一起，倾向于旅行距离短的旅程。一个满足约束条件的网络状态的总能量直接与旅行距离成正比。这使得具有最小能量的状态与最短路径相对应。

定义

$$w_{Xi,Yj} = -A\delta_{XY}(1-\delta_{ij}) - B\delta_{ij}(1-\delta_{XY}) - C - Dd_{XY}(\delta_{j,i+1}+\delta_{j,i-1}) \tag{3.23}$$

TSP 的霍普费尔德能量函数，(3.22)式可被表述如下：

$$E = -\frac{1}{2}\sum_X\sum_Y\sum_i\sum_j w_{Xi,Yj}s_{Xi}s_{Yj} - C\sum_X\sum_i \tilde{N}s_{Xi} + \frac{C}{2}\tilde{N}^2 \tag{3.24}$$

注意上式具有与(3.8)式给出的原始霍普费尔德能量函数相似的形式，但却又有下列不同之处：

(1) 因为网络是二维的，每个变量都有两个下标，而且求和符号也相应增加一倍。

(2) 增加了一项 $\frac{C}{2}\tilde{N}^2$。因为这是个常数项，它对优化过程没有影响。可以证明(3.24)式事实上也是一个李雅普诺夫函数。

(3) 若忽略上述常数项，则外来输入(激活)为 $I_{Xi}=C\tilde{N}$。

TSP 在二值 Sigmoid 非线性和零阈值条件下的连续霍氏网实现步骤可归纳如下：

(1) 初始化网络中的神经元，如 $s_{Xi}=0.5+v$，其中 v 是一个很小的随机数。

(2) 在每次扫描[①] 时重复下列步骤：

① 随机抽取一个在此次扫描中尚未更新的神经元。

② 计算

$$\begin{aligned} u_{Xi}^{+} &= u_{Xi}^{-} + \Delta t\left[-\frac{\mathrm{d}}{\mathrm{d}s_{Xi}^{-}}E\right] \\ &= u_{Xi}^{-} + \Delta t\left[\sum_Y\sum_j w_{Xi,Yj}S_{Yj}^{-} + c\tilde{N}\right] \end{aligned} \tag{3.25}$$

其中 Δt 一般是一个很小的值。

③ $s_{Xi}^{+}=\mathrm{sgm}_{0,\beta}(u_{Xi}^{+})$。

(3) 若到达稳定状态，停止；否则转回第(2)步，进入下一次扫描。

---

① 霍普费尔德和坦客[86]采用了稍许不同但却更加复杂的初始化程序。一个附加项 $-\Delta t u_{Xi}^{-}$ 被引入(3.25)式，但在 Δt 很小时可忽略。

3.4.2 二分图问题

给定一个有一组 N 个(偶数)节点和一组两两节点之间连线的一般图,二分图问题的目的是要将节点分为相等的两组,且令跨越这两组之间的连线最少。这一问题的解答可被应用于 VLSI(超大规模集成电路)的布线设计。例如,图 3.5 给出了一个图的两种不同的二分方式,一个有两条跨越连线(此为最小值),另一个则有七条。

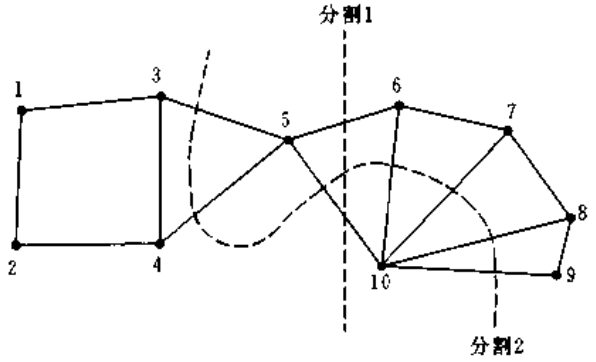


图 3.5 二分图示例

图的连接方式可由如下连接性矩阵表示:

$$\boldsymbol{C}=\begin{bmatrix}0&1&1&0&0&0&0&0&0&0\\1&0&0&1&0&0&0&0&0&0\\1&0&0&1&1&0&0&0&0&0\\0&1&1&0&1&0&0&0&0&0\\0&0&1&1&0&1&0&0&0&1\\0&0&0&0&1&0&1&0&0&1\\0&0&0&0&0&1&0&1&0&1\\0&0&0&0&0&0&1&0&1&1\\0&0&0&0&0&0&0&1&0&1\\0&0&0&0&1&1&1&1&1&0\end{bmatrix} \tag{3.26}$$

其中 $[C]_{ij}=c_{ij}=\begin{cases}1 & \text{若第 } i \text{ 个节点与第 } j \text{ 个节点相连}\\0 & \text{否则}\end{cases}$

注意 $\boldsymbol{C}$ 是一个对称矩阵。记两个分区为 C_A 和 C_B,且定义一个在节点 i 处的神经元为

$$s_i=\begin{cases}1 & \text{若 } i\in C_A\\-1 & \text{若 } i\in C_B\end{cases} \tag{3.27}$$

二分图问题由此可被转化为下列最小化问题:

$$\text{在 } \sum_i s_i=0 \text{ 前提下,最小化 } -\sum_i\sum_{j\neq i}c_{ij}s_is_j \tag{3.28}$$

其中第一项是代价项,且为所有不同节点对的代价之和。最小化代价项相当于试图把每一个节点对的两个节点都放在同一个分区里从而避免出现跨越分区的连线,而约束条件则迫使分区具有相同的大小。

为将这一问题映射到霍氏网的框架中去，我们需要定义一个适当的霍普费尔德能量函数把最小代价和约束条件结合到一起。该函数可证明是李雅普诺夫函数，即

$$E=-\frac{1}{2}\sum_i\sum_{j\neq i}s_is_jc_{ij}+\frac{a}{2}\Big\{\sum_i s_i\Big\}^2 \tag{3.29}$$

$$=\frac{Na}{2}-\frac{1}{2}\sum_i\sum_{j\neq i}s_is_jw_{ij} \tag{3.30}$$

其中 N 是节点数，a 是一个常数(拉格朗日参数)，且 $w_{ij}=c_{ij}-a$。因此，每个神经元的净输入为

$$u_i^+=-\frac{\mathrm{d}}{\mathrm{d}s_i^-}E=\sum_{j\neq i}w_{ij}s_j^- \tag{3.31}$$

而更新规则为

$$s_i^+=\mathrm{sgn}\Big(\sum_{j\neq i}w_{ij}s_j^-\Big) \tag{3.32}$$

3.4.3 N 皇后问题

N 皇后问题即在一个 $N\times N$ 的国际象棋棋盘上放置 N 个国际象棋皇后并使它们互不构成威胁。换句话说，棋盘上不能有多于一个皇后位于一条水平、垂直、或对角(45°或−45°)线上。图 3.6 是一个 8×8 棋盘上符合要求的 8 皇后布局。为将 N 皇后问题的约束条件映射到霍氏网的框架中，可将棋盘上的每个位置用一个神经元来表示。当一个棋盘上的位置被放上一个皇后时，相应的神经元“兴奋”；否则神经元处于“抑制”状态。因此，需要一个具有 $N\times N$ 个神经元的霍氏网来解决这一问题。

记神经元为 s_{ij}，其中下角标 i 和 j 分别表示该神经元在棋盘上所处的行和列。鉴于每一行、列或对角线上都只能有一个皇后，这一问题的约束条件可以映射为如下能量函数：

$$\begin{aligned}E=&\frac{A}{2}\sum_i\Big(\sum_j s_{ij}-1\Big)^2+\frac{B}{2}\sum_j\Big(\sum_i s_{ij}-1\Big)^2\\&+\frac{A+B}{2}\sum_i\sum_j s_{ij}(1-s_{ij})\\&+\frac{C}{2}\sum_i\sum_j\sum_{k\neq 0}s_{ij}(s_{i+k,j+k}+s_{i+k,j-k})\end{aligned} \tag{3.33}$$

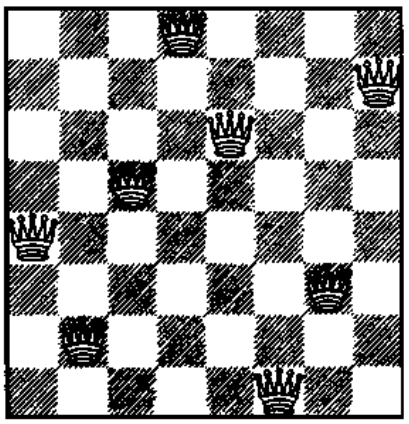


图 3.6　8 皇后问题的一个正确摆法

其中，A，B 和 C 是拉格朗日参数。式中第一项在每行只有一个皇后时为零；第二项在每列只有一个皇后时为零；第三项使每个神经元收敛到“1”或“0”；第四项在每个对角线只有一皇后时为零。可以证明(3.33)式是一个李雅普诺夫函数。因此网络的神经元按下式迭代更新：

$$\begin{aligned}u_{ij}^+&=u_{ij}^-+\Delta t\Big[-\frac{\mathrm{d}}{\mathrm{d}s_{ij}^-}E\Big]\\&=u_{ij}^--\Delta t\Big[A\Big(\sum s_{ij}^--1\Big)+B\Big(\sum_i s_{ij}^--1\Big)\end{aligned}$$

$$+\frac{A+B}{2}(1-2s_{ij}^{-})+C\sum_{k\neq 0}(s_{i+k,j+k}^{-}+s_{i+k,j-k}^{-})\Big] \tag{3.34}$$

$$s_{ij}=\mathrm{sgm}_{\alpha,\beta}(u_{ij}^{+}) \tag{3.35}$$

其中 Δt 通常是一个很小的常数。

3.5 习题

3.1 证明(3.18),(3.24),(3.30)和(3.33)式是李雅普诺夫函数。

3.2 演示例 3.1 中 CAM 的状态转换(提示：首先异步地更新那些不会生成含糊的"0"状况的神经元)。

3.3 讨论给例 3.1 中 CAM 增添附加信息向量 $S^{(3)}=[1\quad -1\quad 1]^{\mathrm{T}}$ 会造成的影响。

3.4 证明(3.22)式与(3.24)式等价。

3.5 编写一个计算机程序，应用霍氏网求解下面连接性矩阵给出的二分图问题。在不同的初始条件下模拟运行 1000 次，观察解答的统计特性。

$$C=\begin{bmatrix}
0&1&0&0&0&0&0&0&1&0&0&0&0&1&0&0&0&0&0&0\\
1&0&1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&1&0&1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&1&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&1&1&0&0&0&1&0&1&1&1&0&0&0&0&0&0&0&0&0\\
0&0&1&1&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&1&0&1&1&0&1&0&0&1&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&1&1&0&0&0&1&0&0&0&0&0&0&0&0&0\\
1&0&0&0&1&0&0&0&0&1&0&0&1&1&0&0&0&0&0&0\\
0&0&0&0&1&0&0&0&1&0&1&1&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&1&1&0&0&0&0&1&1&0\\
0&0&0&0&0&0&0&0&0&1&1&0&0&0&0&1&1&1&0&1\\
0&0&0&0&0&0&0&0&1&0&0&0&0&0&1&1&0&0&0&0\\
1&0&0&0&0&0&0&0&1&0&0&0&0&0&1&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&1&1&0&1&1&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&1&1&0&1&0&1&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&1&0&0&1&1&0&0&0&1\\
0&0&0&0&0&0&0&0&0&0&1&1&0&0&0&0&0&0&1&1\\
0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&1&0&1\\
0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&1&1&1&0&
\end{bmatrix}$$

3.6 用霍氏网求解 20 皇后问题并模拟运行 1000 次。

3.7 重复霍普费尔德的 10 城市 TSP 实验[86]。城市的分布由下列二维坐标给出。

城市	x 坐标	y 坐标
A	0.4000	0.4439
B	0.2439	0.1463
C	0.1707	0.2293
D	0.2293	0.7610
E	0.5171	0.9414
F	0.8732	0.6536
G	0.6878	0.5219
H	0.8488	0.3609
I	0.6683	0.2536
J	0.6195	0.2634

以不同的初始条件模拟运行 1 亿次，并画出出现次数对应于合乎要求旅程的旅行距离的直方图[①]。霍普费尔德在他的实验中用到了下列参数：$A=B=500, C=200, D=500, \tilde{N}=15$，且二值 Sigmoid 函数中 $T_i=0, \beta=0.02$。

3.8 完成从(3.29)式到(3.30)式的推导。

① 如在获取合乎条件的解答上遇到困难，请参考第 7 章。霍氏网在求解 TSP 问题时的一大弱点就是不合乎要求的解答占很大百分比。

第4章 模拟退火和随机机

自从科克派特里克、小哥拉特和瓦克奇(Kirkpatrick,Gelatt Jr. and Vecchi)在前人对于统计力学的研究[108]基础上发表了他们开创性的论文[98]以来,模拟退火算法被赞誉为解决许多高难度组合最优化问题的"救星",并已被应用于诸如超大规模集成电路(VISI)的计算机辅助设计[100][146][164]、图象处理和计算机视觉[18][159][170]、电信(文献[15,37,51,55,139,165,166,167,168])、经济[63][171]以及其它众多工程和科学领域(文献[104,115,133,152,157])。在将退火的概念运用于神经网络的过程中,一系列被称为随机机的新型计算方法又涌现了出来。这一章对模拟退火和各种随机机的派生进行综述,并重点介绍模拟退火的基本特点。读者可从文献[2,58,59,123,163]等找到更多详细的分析。

4.1 统计力学和麦绰泼里斯算法

模拟退火是根据液态或固态材料中粒子的统计力学与复杂组合最优化问题的求解过程的相似之处而提出来的。统计力学论述材料中相互作用粒子的特性。材料中粒子结构的不同对应于不同的能量水平。如果用粒子结构或其相应能量来定义材料的状态,麦绰泼里斯(Metropolis)算法[108]可以给出一个简单的数学模型,用于描述材料在温度 T 下从具有能量 $E(i)$ 的状态 i 进入具有能量 $E(j)$ 的状态 j 的机制:

- 若 $E(j)\leqslant E(i)$,状态转换被接收。
- 若 $E(j)>E(i)$,则状态转换以如下概率被接收:

$$e^{\frac{E(i)-E(j)}{KT}} \tag{4.1}$$

其中 K 是物理学中的波尔兹曼常数,T 是材料的温度。

在一个特定温度下,如果进行足够多次数的转换,将能达到热平衡。此时材料处于状态 i 的概率可由波尔兹曼分布表述

$$\pi_i(T)=P_T(s=i)=\frac{e^{-\frac{E(i)}{KT}}}{Z_T} \tag{4.2}$$

其中 s 是一个用于表示材料当前状态的随机变量,$Z_T=\sum_{j\in S}e^{-\frac{E(j)}{KT}}$ 被称为划分函数,它对在状态空间 S 上的所有可能状态求和。显而易见,

$$\lim_{T\to\infty}\pi_i(T)=\lim_{T\to\infty}\frac{e^{-\frac{E(i)}{KT}}}{\sum_{j\in S}e^{-\frac{E(j)}{KT}}}=\frac{1}{|S|} \tag{4.3}$$

这表明所有状态在高温下具有相同的概率。而随着温度的下降,

$$\begin{aligned}\lim_{T\to 0}\pi_i(T) &= \lim_{T\to 0}\frac{e^{-\frac{E(i)-E_{\min}}{KT}}}{\sum_{j\in S}e^{-\frac{E(j)-E_{\min}}{KT}}}\\ &= \lim_{T\to 0}\frac{e^{-\frac{E(i)-E_{\min}}{KT}}}{\sum_{j\in S_{\min}}e^{-\frac{E(j)-E_{\min}}{KT}}+\sum_{j\notin S_{\min}}e^{-\frac{E(j)-E_{\min}}{KT}}}\\ &= \lim_{T\to 0}\frac{e^{-\frac{E(i)-E_{\min}}{KT}}}{\sum_{j\in S_{\min}}e^{-\frac{E(j)-E_{\min}}{KT}}}\\ &= \begin{cases}\dfrac{1}{|S_{\min}|} & \text{如果 } i\in S_{\min}\\ 0 & \text{其它}\end{cases}\end{aligned} \tag{4.4}$$

其中 $E_{\min}=\min_{j\in S}E(j)$ 且 $S_{\min}=\{i:E(i)=E_{\min}\}$。从上式可见，当温度降至很低时，材料倾向于进入具有最小能量的状态。

在统计力学中，固体的晶格结构通常处于低能状态。退火作为一个获取热槽中固体低能状态的热处理过程，经常用于晶体生长。这一过程首先将热槽中的固体加热，使其溶化为液体，然后逐步降温。在每个温度下，所有粒子随机排列直至达到热平衡。如果冷却过程足够缓慢，从而确保在每个温度下都达到热平衡，则当系统冻结时($T\to 0$)低能晶体固态形成。相反，热槽温度迅速下降的淬火过程会使固化的结果为具有非晶结构或亚稳非晶结构(有缺陷的晶体)的玻璃状物质。

熵的概念有些深奥，但毕竟是理解热过程的一个非常重要的热力学工具。材料在热平衡状态下熵的定义为

$$H(T)=-\sum_{i\in S}\pi_i(T)\cdot\ln\pi_i(T) \tag{4.5}$$

因此，材料在极限温度下的熵为

$$\lim_{T\to\infty}H(T)=-\sum_{i\in S}\frac{1}{|S|}\ln\frac{1}{|S|}=\ln|S| \tag{4.6}$$

$$\lim_{T\to 0}H(T)=-\sum_{i\in S_{\min}}\frac{1}{|S_{\min}|}\ln\frac{1}{|S_{\min}|}=\ln|S_{\min}| \tag{4.7}$$

且有

$$\begin{aligned}\frac{\partial H(T)}{\partial T} &= -\sum_{i\in S}[\ln\pi_i(T)+1]\cdot\frac{\partial\pi_i(T)}{\partial T}\\ &= -\sum_{i\in S}\left[-\frac{E(i)}{KT}-\ln Z_T+1\right]\cdot\frac{\partial\pi_i(T)}{\partial T}\end{aligned} \tag{4.8}$$

如果我们定义平均能量为

$$\overline{E}_T=\sum_{i\in S}\pi_i(T)\cdot E(i) \tag{4.9}$$

方差为

$$\sigma_T^2 = \sum_{i\in S}(E(i) - \overline{E}_T)^2 = \overline{E_T^2} - \overline{E}_T^2 \tag{4.10}$$

则

$$\begin{aligned}
\frac{\partial \pi(T)_i}{\partial T} &= \frac{\partial}{\partial T}\left\{\frac{e^{\frac{-E(i)}{KT}}}{Z_T}\right\} \\
&= \frac{E(i)}{KT^2}\frac{e^{\frac{-E(i)}{KT}}}{Z_T} - \frac{1}{Z_T^2}e^{\frac{-E(i)}{KT}}\frac{\partial}{\partial T}Z_T \\
&= \frac{E(i)}{KT^2}\pi_i(T) - \frac{\pi_i(T)}{KT^2}\frac{1}{Z_T}\sum_{i\in S}e^{\frac{-E(i)}{KT}}E(i) \qquad (4.11) \\
&= \frac{\pi_i(T)}{KT^2}[E(i) - \overline{E}_T] \\
\frac{\partial H(T)}{\partial T} &= \sum_{i\in S}\left[\frac{E(i)}{KT} + \ln Z - 1\right]\cdot\frac{\pi_i(T)}{KT^2}[E(i) - \overline{E}_T] \\
&= \frac{1}{K^2T^3}\sum_{i\in S}E(i)\pi_i(T)[E(i) - \overline{E}_T] \\
&\quad + \frac{(\ln Z - 1)}{KT^2}\sum_{i\in S}\pi_i(T)[E(i) - \overline{E}_T] \qquad (4.12) \\
&= \frac{1}{K^2T^3}\left[\sum_{i\in S}E^2(i)\cdot\pi_i(T) - \overline{E}_T\sum_{i\in S}E(i)\cdot\pi_i(T)\right] \\
&= \frac{\delta_T^2}{K^2T^3}
\end{aligned}$$

由 $T>0$ 和 $\sigma_T^2\geqslant 0$，可知 $\frac{\partial H(T)}{\partial T}\geqslant 0$。可从(4.3)式、(4.4)式和(4.12)式看出熵随温度下降而单调递减(见图 4.1)。在统计力学中，熵被用来衡量物理系统的有序性：熵越大，系统越无序。如果温度缓慢下降并使材料在每个温度下都松弛到热平衡，材料的熵在退火过程中会单调递减，使得材料进入有序的(晶态)结构。

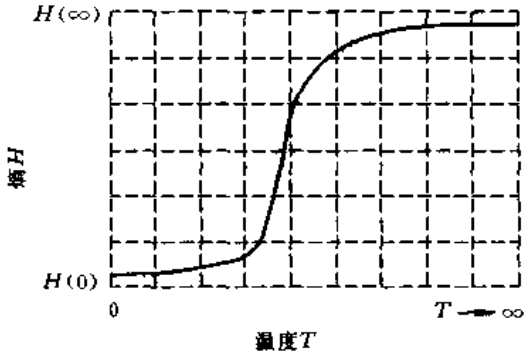


图 4.1 材料的熵 $H(T)$ 与温度 T 的关系

4.2 模拟退火

表 4.1 给出了麦绰泼里斯算法与模拟退火的类比。麦绰泼里斯算法描述了液体结晶的过程：在高温下，粒子能量较高，可以自由运动和重新排序；在低温度下，粒子能量减弱，迁移率减小，并最终进入平衡态，固化为具有最小能量的晶体。模拟退火算法搜索组合优化问题的最优解或最佳组合。假定有待解决的问题是寻找一个多变量代价函数的最小值，可采用一种简单的迭代算法——局域搜索来求解最小代价。局域搜索从一个给定的初始解答出发，随机地生成新的解答。如果这一新解的代价小于当前解答的代价，则用它取代当前解答，否则舍弃这一新解。局域搜索算法不断地随机生成新解并重复上述步骤，直至求得最小代价解。不幸的是，局域搜索可能陷于局部最小值而不能自拔。为避免搜索过程在局部最小处受阻，模拟退火允许产生“爬山运动”，即转移到代价较高的解答。

表 4.1 麦绰泼里斯算法与模拟退火的类比

麦绰泼里斯算法	模拟退火
· 热退火过程的数学模型	· 组合优化中局域搜索的推广
· 物理系统中的一个状态	· 最优化问题的一个解答
· 状态的能量	· 解答的代价
· 粒子的迁移率	· 解答的接受率
· 温度	· 控制参数

考虑一个定义在某个有限集 S 上代价函数为 $f:s\to R^+$ 的组合优化问题，并定义有限集 S，其中 $s\in S$ 是待求问题的解或构形。对每一个构形 $s\in S$ 都存在一个由对 s 的微小扰动而生成的邻域集合 $N(s)\subseteq S$。

在模拟退火中，接受从一个状态 $s(k)$ 到另一个状态 $s'(k)$ 的概率由麦绰泼里斯准则[108]决定：

$$P[s(k),s'(k)]=P_r\{s(k)\to s'(k)\}=\mathrm{e}^{-\frac{[f(s'(k))-f(s(k))]^+}{T}} \tag{4.13}$$

其中 $s'(k)$ 是通过一个扰动机制由 $N(s)$ 生成的新解，k 表示第 k 次迭代，而 $[x]^+=\max\{0,x\}$。相应地

$$P_r\{s(k+1)=s'(k)\}=\begin{cases}1 & \text{若 } f(s'(k))<f(s(k))\\ \mathrm{e}^{-\frac{f(s'(k))-f(s(k))}{T}} & \text{否则}\end{cases} \tag{4.14}$$

从(4.14)式给出的麦绰泼里斯准则可以看出，在一个给定温度 T 下，模拟退火的操作类似于局域搜索但却允许搜索过程按一定概率从一个较低代价构形“爬山”到一个较高代价构形，从而避免搜索过程陷于局部最小值。随机点过程 $\{s(k):k\geqslant 0\}$ 可用离散时间齐次马尔科夫(Markov)链[2]来分析。一步转移矩阵为

$$P(i,j)=P_r[s(k+1)=j|s(k)=i]$$
$$=\begin{cases}0 & \text{若 } j\notin N(i) \text{ 且 } j\neq i\\ G(i,j)\min\{1,e^{-\frac{f(j)-f(i)}{T}}\} & \text{若 } j\in N(i) \text{ 且 } j\neq i\\ 1-\sum\limits_{i'\neq i}G(i,i')\min\{1,e^{-\frac{f(j)-f(i)}{T}}\} & \text{若 } j\neq i\end{cases} \tag{4.15}$$

其中 $G(i,j)$ 是从构形 i 生成构形 j 的概率。

如果生成任一组合 i 的概率在其邻域集合 $N(i)$ 中均匀分布，且构形的转移基于(4.15)式，则相应的马尔科夫链可被证明是不可约的、非周期性的、循环的[2]。在这些条件下，经过有限次的转移，组合 i 的静平衡分布 $\pi_i(T)$ 由波尔兹曼分布给出，即

$$\begin{aligned}\pi_i(T)&=\lim_{k\to\infty}P_r\{s(k)=i|T\}\\&=\lim_{k\to\infty}P_r\{s(k)=i|s(0)=j,\forall\ j,T\}\\&=\frac{e^{-\frac{f(i)}{T}}}{\sum\limits_{j\in S}e^{-\frac{f(j)}{T}}}\end{aligned} \tag{4.16}$$

与统计力学((4.4)式)的分析类似，构形 i 在过程冻结时的静平衡分布变为

$$\pi_i^*=\lim_{T\to\infty}\pi_i(T)=\begin{cases}\dfrac{1}{|S_{\min}|} & \text{如果 } i\in S_{\min}\\ 0 & \text{否则}\end{cases} \tag{4.17}$$

而且

$$\lim_{T\to 0}[\lim_{k\to\infty}P(s(k)\in S_{\min})]=\lim_{T\to 0}\sum_{i\in S}\pi_i(T)=\sum_{i\in S_{\min}}\pi_i^*=1 \tag{4.18}$$

该式表明模拟退火算法渐进收敛于具有最小代价的构形。也就是说，如果温度是缓慢下降的，且在每个温度下都有足够数目的转移，则最小代价的全局最优构形(解答)以概率 1 被找到。关于收敛性的各种证明可见文献[59,60,61,70]，其中文献[70]给出了最严格的充要条件。

4.2.1 有限时间实现

模拟退火算法有两个主要操作：一个是称为冷却流程的热静力学操作，用于设定温度下降幅度(算法的一个参数)；另一个用于在每个温度下搜索最优解的随机松弛过程。本章前述分析只是说明模拟退火具有逐渐达到最优解的能力，但却是以搜索过程经历无限次转换为前提的。在实际应用中，模拟退火必须在有限时间内实现，否则它就显示不出相对于简单随机搜索的优势。

在有限时间条件下实现模拟退火算法需要有：一个起始温度、一个控制温度下降的函数、一个决定在每个温度下状态转移参数的准则、一个终止温度以及一个终止模拟退火的准则。

科克派特里克等人[98]在解决计算机芯片的布线问题时采用了下列指数冷却流程：

(1) 给定一个高起始温度 $T_0=10$。相应于材料溶化为液态的物理状况，一个高的温度确保接受率接近 1。一个温度下的接受率定义为被接受的状态转移数与在给定温度下所有提出的状态转移数之比。在高温下，事实上所有提出的转移都被接受。随着温度的下

降，状态转移的接受率越来越小。状态转移最终随着系统冻结于一个很低温度时停止。起始温度的获取可从一个较低温度出发，逐渐升温直至接受率接近 1。

(2) 降温函数为 $T_{k+1}=\alpha T_k$，其中 α 通常接近 1，且 T_k 代表第 k 次递减时的温度。科克派特里克等用的是 $\alpha=0.9$。

(3) 在每个温度下的马尔科夫链的长度就是达到 ε-准平衡态时被生成的状态转移个数，其中 ε 是一个特定的正常数。换句话说，当给定温度下马尔科夫链的长度足以使已得解答的概率分布以 ε(在范数意义上)接近于相应的静态分布。直观上，这一条件可被定量化为要求在每个温度下被接受的转移数超过特定阈值。科克派特里克等以 50000 次被接受的转移作为每个温度下的阈值。同时，ε 越大，要求的马尔科夫链越小。注意，在高温下几乎所有生成的转移都被接受，因而 ε-准平衡态迅速达到。也就是说，高温下解答的概率分布接近均匀，与(4.3)式给出的静态分布相同。这就是为什么退火应从起始温度 T_0 出发。当温度降到 T_0 以下时，接受率从 1 急剧下降。在降温函数的降温梯度与马尔科夫链的长度之间，存在这一个取舍的问题。降温梯度越大，达到准平衡态所需的马尔科夫链越长。在温度趋于零时，基本上没有可被接受的状态转移，因而马尔科夫链可能会非常的长。在实际应用中，为马尔科夫链的长度设有一个常数上限。

(4) 当温度足够接近于零，或连续一组马尔科夫链上最后一个解答的代价不再发生变化时，退火过程终止。科克派特里克等所用的终止条件则是在最后连续三个温度下都达不到所希望的接受数。

阿特斯(Aarts)和拉合文(Laarhoven)提出了另一个冷却流程[2][4]。该流程使得模拟退火算法的运行可在多项式时间内完成。

(1) 起始温度 T_0 由下述过程得到：

令 n_1 为代价减小的转移的数目，

n_2 为代价增大的转移的数目，

$\bar{\Delta}^{(+)}$为 n_2 个代价增大转移的平均代价差。

将接受率近似为

$$A \approx \frac{n_1 + n_2 e^{-\frac{\bar{\Delta}^{(+)}}{T}}}{n_1 + n_2} \tag{4.19}$$

可得

$$T \approx \frac{\bar{\Delta}^{(+)}}{\log_e\left(\frac{n_2}{n_2 A - n_1(1-A)}\right)} \tag{4.20}$$

① 设置 $T_0=0$。

② 生成 n 组解答，并以(4.20)式更新 T。这里 $n=n_1+n_2$。

③ 基于②，以(4.19)式更新 A。

④ 重复②和③直到 A 超过一个接近于 1 的阈值。

(2) 按下式降温

$$T_{k+1} = \frac{T_k}{1 + \frac{T_k \log_e(1+\delta)}{3\sigma_k}} \tag{4.21}$$

其中 δ 被称为距离参数，σ_k 是在温度 T_k 处生成的所有解答(状态)的标准差。在实际应用中，标准差可由样本差来近似。δ 越大，降温幅度也就越大；反之亦然。

(3) 在每个温度下马尔科夫链的长度可被视为在这一温度下起始解答邻域的大小，而这一邻域的大小又取决于生成解答的机制。以 N 城市 TSP 为例，如果使用稍后将要介绍的对换机制，邻域的大小为 $(N-1)(N-2)$。

(4) 退火过程在温度 T 到达终止温度 T_s 时停下。T_s 满足下列条件：

$$\frac{T_s}{<f>_\infty}\left.\frac{\partial <f>_T}{\partial T}\right|_{T=T_s} < \varepsilon_s \tag{4.22}$$

其中 ε_s 是称为停止参数的较小正常数，而 $<f>_T$ 则为温度 T 下生成的解答的代价均值(代价期望值)。在实际应用中，代价的期望值通常由样本均值来近似。

可以证明，在应用上述冷却流程的条件下，到达停止准则时的马尔科夫的链数以 $O(\log_e|S|)$ 为上限，算法计算复杂度为 $O(\tau L\log_e|S|)$，其中 τ 是计算一次转移所需的时间，L 则是每个马尔科夫链的长度(假定在任一温度下的马尔科夫链长度不变)。

大量用快速动态流程(文献[22,68,92,155])或并行计算方法(文献[5,23,33,93,169])加速退火过程的研究工作已经发表，一些新近涌现出来技术也被用于与模拟退火相比较和结合，例如进化算法和 tabu 搜索等(文献[29,101,106,172,174])。

4.2.2 一个例子：TSP

用模拟退火解决组合优化问题需要有：

- 简明的问题表示：在解答空间上所有可能解有良好定义的代价函数。
- 从一个解答到另一个解答的扰动和转移机制。
- 冷却流程。

下面以旅行商问题为例给出模拟退火求解过程的框架。这一 TSP 已在第 3 章定义过，在此只简述如下：旅行商要访问 N 个城市并回到起始城市，令 $D=[d_{XY}]$ 为距离矩阵，且 d_{XY} 为城市 X 和城市 Y 之间的距离，其中 $X,Y=1,2,\cdots,N$。将此 TSP 映射到用模拟退火解决的框架中去要经过以下步骤：

(1) TSP 的解答空间定义为

$$S = \{N \text{ 城市的所有循环排列}, \Xi = (\xi(1),\cdots,\xi(N))\} \tag{4.23}$$

其中 $\xi(k)$ 表示从城市 k 出发访问的下一个城市。旅程的总代价定义为

$$f(\Xi) = \sum_{X=1}^{N} d_{X,\xi(X)} \tag{4.24}$$

(2) 有很多类型生成机制可用于 TSP。这里采用二交换机制[102]作为示例。任选两个城市 X 和 Y，二交换机制通过反转 X 和 Y 之间访问城市的顺序而获取新的旅程(见图 4.2)。给定一个旅程

$$(\xi(1),\cdots,\xi^{-1}(X),X,\xi(X),\cdots,\xi^{-1}(Y),Y,\xi(Y),\cdots,\xi(N)) \tag{4.25}$$

施加交换机制于城市 X 和 Y，可得新的旅程

$$(\xi(1),\cdots,\xi^{-1}(X),X,\xi^{-1}(Y),\cdots,\xi(X),Y,\xi(Y),\cdots,\xi(N)) \tag{4.26}$$

注意(4.25)式中从城市 $\xi(X)$ 到城市 $\xi^{-1}(Y)$ 的一段路径被反转为(4.26)式中从城市

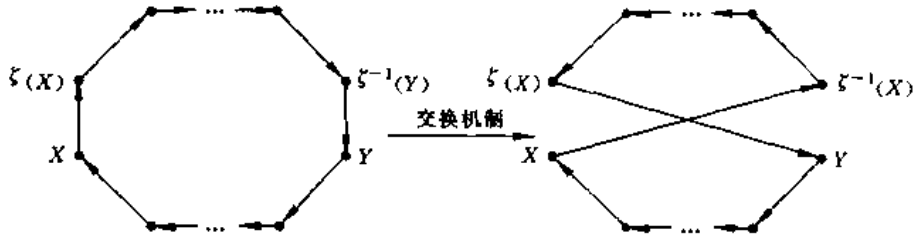


图 4.2 交换机制示例

$\xi^{-1}(Y)$到城市 $\xi(X)$的路径，但这段路径的距离并没有变化。所以，旅程代价的变化为

$$\Delta f = d_{X,\xi^{-1}(Y)} + d_{\xi(X),Y} - d_{X,\xi(X)} - d_{\xi^{-1}(Y),Y} \tag{4.27}$$

这一转换机制符合麦绰泼里斯准则，因此

$$\text{接受新旅程的概率} = \begin{cases} 1 & \text{如果 } \Delta f \leqslant 0 \\ e^{-\frac{\Delta f}{T}} & \text{否则} \end{cases} \tag{4.28}$$

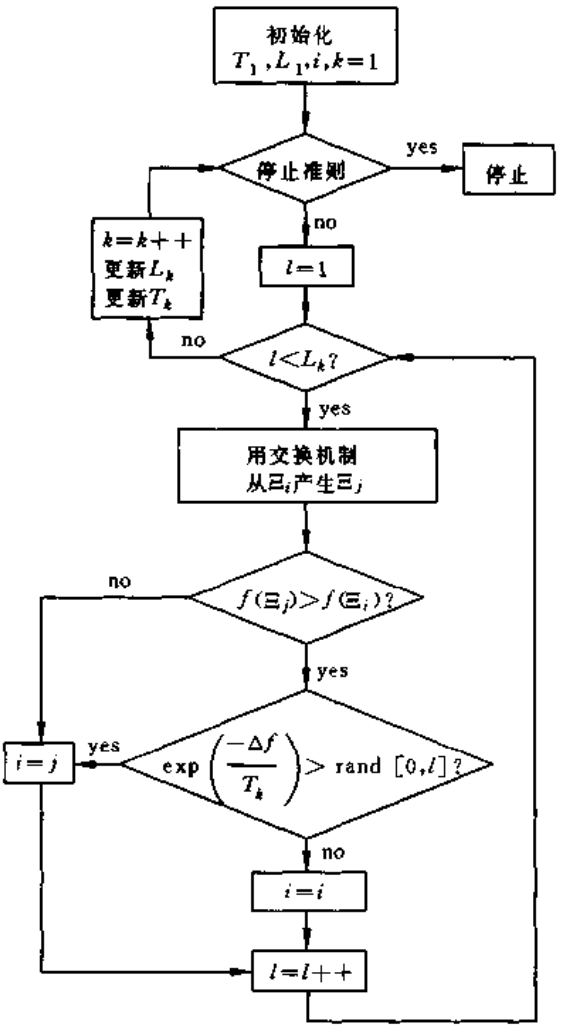


图 4.3 应用模拟退火求解 TSP 的总结

(3) 可以采用诸如 4.2.1 节介绍的各种冷却流程。图 4.3 所示流程图总结了用模拟退火求解 TSP 的步骤。图中

- i 和 j 标记旅程，k 标记温度(算法参数)，而 l 则为在每个温度下已生成旅程的个数。
- T_k 和 L_k 分别为第 k 个马尔科夫链的温度和允许长度。
- Ξ_i 是当前旅程，而 Ξ_j 是新生成的旅程。
- $f(\Xi_i)$是旅程 Ξ_i 的代价。
- $\Delta f=f(\Xi)-f(\Xi_i)$
- rand[0,1]是来自一个在 0 和 1 之间均匀分布的随机数发生器的值。

4.3 随机机

随机霍氏网是一个最基本的波尔兹曼机。以基于热动力学中的艾星(Ising)模型的转移概率代替麦绰泼里斯准则，可将模拟退火应用于离散霍氏网就得到波尔兹曼机。在随机机系列中，高斯机是最通用的。用它可得到波尔兹曼机和柯西(Cauchy)机。

4.3.1 波尔兹曼机

考虑图 3.1 所示的双极霍氏网，回想

$$E(S)=-\frac{1}{2}\sum_i\sum_{j\neq i}s_is_jw_{ij}-\sum_i s_iI_i \tag{4.29}$$

翻转一个神经元的状态，从 s_i 到 $-s_i$，所引起的能量变化为

$$\begin{aligned}\Delta E(i)&=E(s_1,s_2,\cdots,-s_i,\cdots,s_N)-E(s_1,s_2,\cdots,s_i,\cdots,s_N)\\&=\Delta s_i\frac{\partial}{\partial S_i}E\\&=\Delta s_i\Big(-\sum_{j\neq i}w_{ij}s_i-I_i\Big)\\&=2s_iu_i\end{aligned} \tag{4.30}$$

依据艾星模型，接受这一从 s_i 到 $-s_i$ 转移的概率为

$$P(s_i\to -s_i)=\frac{1}{1+e^{\frac{\Delta E(i)}{T}}}=\frac{1}{1+e^{\frac{2s_iu_i}{T}}} \tag{4.31}$$

其中 T 是退火过程的温度。图 4.4 示出了(4.31)式所给出的接受概率。可见转移到低能状态的概率比转移到高能状态的要大。在高温下状态转移极易实现。而在温度到达零点时，转移到一个具有较高能量状态成为不可能发生的事情。因此，神经元 i 状态翻转的概率为

$$\text{从 }-1\text{ 到 }1:\ \frac{1}{1+e^{-\frac{2u_i}{T}}}=P(u_i) \tag{4.32}$$

$$\text{从 }1\text{ 到 }-1:\ \frac{1}{1+e^{\frac{2u_i}{T}}}=1-P(u_i) \tag{4.33}$$

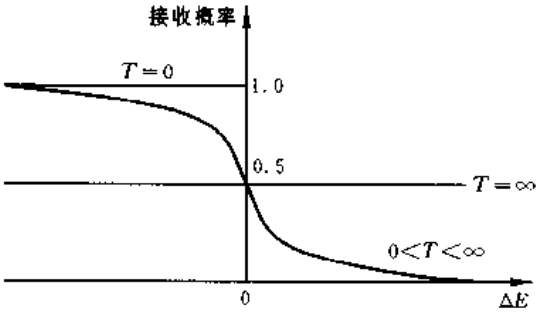


图 4.4 波尔兹曼机在不同温度下的接受概率

用下式给出的随机激活函数取代图 3.1 中确定性的非线性函数，可以方便地获取随机霍氏网：

$$\zeta(u_i)=\begin{cases}1 & \text{以概率 } P(u_i)\\ -1 & \text{以概率 } 1-P(u_i)\end{cases} \tag{4.34}$$

用 u_i 代替 $2u_i$，(4.30)～(4.34)式可以用类似的方式运用于二值网。可以证明，尽管转移概率不同，稳态分布没有变化，且仍为波尔兹曼分布((4.2)式)。证明过程与 4.2 节给出的类似。随机霍氏网(波尔兹曼机)可泛化为神经元任意互连的网络，而这些神经元又可进一步划分为可见的和隐含的两类。所有随机霍氏网的神经元都是可见的。隐含的神经元与网络的输入或输出没有直接联系。可见神经元又可划分为输出神经元和输入神经元两类。在霍氏网被用作联想式存储器时，权值由训练获得。类似地，波尔兹曼机应用于联想式存储器或有教师的学习等问题时，它的权值也可由训练得到。

组合优化问题不但可被映射到霍氏网的框架上加以解决，它们也可被表述为二值规划问题[3]。回顾 TSP 并定义下列变量：

$$S\triangleq\{s_{Xi}\in\{0,1\}\mid X,i=1,2,\cdots,N\}, \tag{4.35}$$

其中

$$s_{Xi}=\begin{cases}1 & \text{如果城市 } X \text{ 在旅程中处于第 } i \text{ 个位置}\\ 0 & \text{其它}\end{cases} \tag{4.36}$$

由此，TSP 可被表述为求取使下列代价函数最小的旅程。

$$f(S)\mid_{S\in S}=\sum_X\sum_Y\sum_i\sum_j\tilde{w}_{XiYj}s_{Xi}s_{Yj} \tag{4.37}$$

其约束条件为

$$\sum_X s_{Xi}=1\ \text{且}\ \sum_i s_{Xi}=1,\forall\ X,i=1,2,\cdots,N \tag{4.38}$$

其中

$$\tilde{w}_{XiYj}=\begin{cases}d_{XY} & \text{若 } j=(i+1)\bmod N\\ 0 & \text{其它}\end{cases} \tag{4.39}$$

在波尔兹曼机上实现上面表述时，每个变量都用一个神经元实现，并通过引入约束重新定义权值。其目的是找到波尔兹曼机的一个构形(状态)，使如下一致性函数(consensus function)取得最大值。

$$C(S)|_{S\in S}=\sum_{X}\sum_{Y}\sum_{i}\sum_{j}w_{X_iY_j}s_{X_i}s_{Y_j} \tag{4.40}$$

其中

$$w_{X_iY_j}\begin{cases}>\max\{d_{XA}+d_{XB}|A\neq B\} & \text{若 } X=Y \text{ 且 } i=j\\ =-d_{XY} & \text{若 } X\neq Y \text{ 且 } j=(i+1)\bmod N\\ <-\min\{w_{X_iX_i},w_{Y_jY_j}\} & \text{若}(X=Y \text{ 且 } i\neq j) \text{ 或}(X\neq Y \text{ 且 } i=j)\end{cases} \tag{4.41}$$

这个一致性函数与代价函数十分相像，不过(4.38)式给出的代价函数约束条件已被吸收到(4.41)式给出的一致性函数的权值中去了。代价函数的最小化因而也等价于一致性函数的最大化。可以证明波尔兹曼机的一致性函数是可行的。也就是说，一致性函数的所有局部最大值都对应于有效的旅程，满足(4.38)式。这一函数也是保序的，亦即

$$f(S^{(k)})>f(S^{(l)})\Rightarrow C(S^{(k)})<C(S^{(l)}) \text{ 对于 } S^{(k)},S^{(l)}\in S \text{ 和 } k\neq l \tag{4.42}$$

与随机霍氏网使能量最小不同的是，这一波尔兹曼机使一致性最大。因此，在对这一模型采用模拟退火时，接受一个从 $S^{(-)}$ 到 $S^{(+)}$ 的状态转移的概率为

$$\text{接受概率}=\frac{1}{1+\mathrm{e}^{-\frac{\Delta C}{T}}} \tag{4.43}$$

其中 $\Delta C=C(S^{(+)})-C(S^{(-)})$。当 $T\to 0$ 时，基本上不可能转换到一个一致性较小的状态。接受概率看上去是图 4.4 中曲线沿 y 轴的翻转。

4.3.2 高斯机

与波尔兹曼机的推导相似，高斯机[12]可由连续霍氏网(图 3.3)获得。高斯机的动态特性与连续霍氏网的类似，只不过在每个神经元处加入了一个随机变量，从而有

$$s_i=\mathrm{sgm}_{T_i,\beta}(u_i) \tag{4.44}$$

$$\frac{\mathrm{d}}{\mathrm{d}t}u_i=\sum_{j\neq i}w_{ij}s_j+I_i+\eta \tag{4.45}$$

其中 η 是一个均值为零的高斯随机变量。它的标准差为 $\sigma=cT$，其中 c 是一个常数，T 是温度(与模拟退火有关的控制参数)。加入随机扰动的目的是使网络能够跳出局部最小值。

实现高斯机的步骤可以归纳如下：

(1) 初始化。

① 设 $s_i=v$，其中 v 是一个分布于 $-\frac{1}{N}$ 与 $\frac{1}{N}$ 之间的小随机数。

② 设 T_0 为起始温度。

③ 定义退火流程。阿其雅玛(Akiyama)等采用了如下降温函数：

$$T_k=\frac{T_0}{1+\frac{k}{c_T}} \tag{4.46}$$

其中 T_k 是第 k 个温度，而 C_T 是一个常数。

(2) 在每个温度处重复下列步骤直至达到热平衡[①]:

① 随机选择一个在此次扫描中尚未被更新的神经元。

② 计算

$$u_i^+ = u_i^- + \Delta t\Big[\sum_{j\neq i} w_{ij}s_j^- + I_i + \eta\Big] \tag{4.47}$$

其中 Δt 通常是一个很小的数值。

③ $s_i^+ = \mathrm{sgm}_{\beta_k}(u_i^-)$。阿其雅玛等[12]采用了

$$\beta_k = \frac{\beta_0}{1 + \dfrac{k}{c_\beta}} \tag{4.48}$$

其中 β_0 是 β 的初始值,c_β 是个常数。注意,当 $k\to\infty$ 时,Sigmoid 函数的值接近硬限器。开始时神经元的值聚集在 0 附近,随着退火过程的推进,它们逐渐接近并稳定在 1 或 -1 附近。这一使 Sigmoid 函数变陡的机制被称为函数的锐化。

(3) 降低温度,并重复步骤(2)直到温度到达零点。

上述的 u_i^+ 是一个随机过程。如果采用瞬时激活和硬限器(即除去图 3.3 中的积分器并使用硬限函数),则高斯机的动态过程为

$$u_i = \sum_{j\neq i} w_{ij}s_j + I_i + \eta \text{ 且 } s_i = \mathrm{sgn}(u_i) \tag{4.49}$$

因此可得

$$\begin{aligned} P(s_i = 1) = P(u_i \geqslant 1) &= \int_0^\infty f_{u_i}(\lambda)\mathrm{d}\lambda \\ &= \int_0^\infty \frac{1}{\sqrt{2\pi}\sigma_{u_i}} \mathrm{e}^{\left(-\frac{(\lambda-\bar{u}_i)^2}{2\sigma_{u_i}^2}\right)}\mathrm{d}\lambda \\ &= \int_{-\infty}^{\frac{\bar{u}_i}{\sigma_{u_i}}} \frac{1}{\sqrt{2\pi}} \mathrm{e}^{\left(-\frac{\lambda^2}{2}\right)}\mathrm{d}\lambda \\ &= \Phi\left(\frac{\bar{u}_i}{\sigma_{u_i}}\right) \end{aligned} \tag{4.50}$$

其中 $f_{u_i}(\cdot)$ 是 u_i 的概率分布函数,$\bar{u}_i$ 和 σ_{u_i} 分别是 u_i 的均值和标准差,$\Phi(\cdot)$ 是累积高斯分布函数。注意 $\Phi(\cdot)$ 具有类似于 Sigmoid 函数的特性,因此选择一个适当的 c 可使高斯机接近波尔兹曼机(见习题 4.5)。

4.3.3 柯西机

柯西机,亦称快速模拟退火[155][156],也可容易地由高斯机得到。只需将高斯机中的高斯随机变量(白噪声)用柯西随机变量(有色噪声)取代即可。这一替换增加了网络接受一个代价变大转移的可能性,因而增强了它跳出局部最小值的能力。另外,应用下述快速冷却流程使得柯西机有可能比另两个随机机更快地收敛:

① 在实际应用中,可以限定扫描次数。

$$T_k = \frac{T_0}{1+k} \tag{4.51}$$

其中 T_k 为第 k 次降温或扫描时的温度，T_0 是起始温度。

如果应用瞬时激活和硬限函数，则柯西机的动态过程类似于高斯机，只是下式中的 η 是一个柯西随机变量：

$$u_i = \sum_{j \neq i} w_{ij} s_j + I_i + \eta \text{ 且 } s_i = \operatorname{sgn}(u_i) \tag{4.52}$$

其中 η 的概率密度函数为[48]

$$f_\eta(\eta) = \frac{1}{\pi} \frac{v}{v^2 + u^2} \tag{4.53}$$

这里，$v>0$ 是一个常数。众所周知，柯西随机变量的均值和方差是不确定的。所以神经元 s_i 被激活的概率为

$$\begin{aligned} P(s_i = 1) = P(u_i \geqslant 0) &= \int_0^\infty f_{u_i}(\lambda)\mathrm{d}\lambda \\ &= \int_0^\infty \frac{1}{\pi} \frac{v}{v^2 + (\lambda - \sum_{j \neq i} w_{ij} s_j - I_i)} \mathrm{d}\lambda \\ &= \frac{1}{2} + \frac{1}{\pi} \arctan\left(\frac{\sum_{j \neq i} w_{ij} s_j + I_i}{v} \right) \end{aligned} \tag{4.54}$$

其中 $f_{u_i}(\cdot)$ 是柯西随机变量 u_i 的概率分布函数。如果设定 v 等于退火过程的温度参数 T，(4.54)式与柯西机在温度 T 处有一个神经元被激活的概率相同。换句话说，在每个温度处，(4.54)式可用图 4.5 中两个等价途径中的任意一个实现。图中 rand[0,1]是来自一

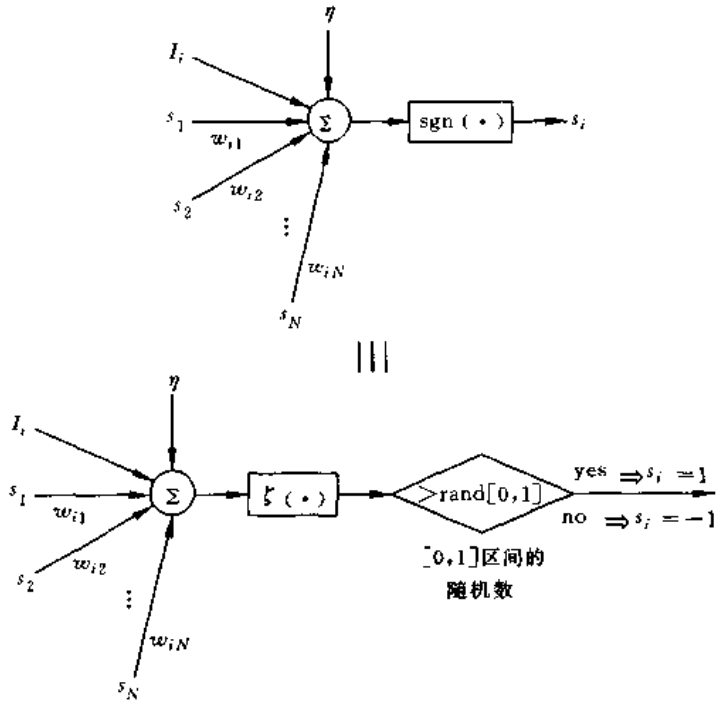


图 4.5 柯西机动态过程的实现

个在 0 和 1 之间均匀分布的随机数发生器的值，且

$$\zeta(x) = \frac{1}{2} + \frac{1}{\pi}\arctan\left(\frac{x}{T}\right) \tag{4.55}$$

4.4 习题

4.1 用模拟退火算法求解一个 TSP 的最小旅程距离。TSP 的距离矩阵如下：

$$\boldsymbol{D} = \begin{bmatrix} 0 & 1 & 2 & 3 & 4 & 5 \\ 1 & 0 & 6 & 1.5 & 2.5 & 0.5 \\ 2 & 6 & 0 & 0.5 & 0.2 & 1.2 \\ 3 & 1.5 & 0.5 & 0 & 0.4 & 4.4 \\ 4 & 2.5 & 0.2 & 0.4 & 0 & 5.1 \\ 5 & 0.5 & 1.2 & 4.4 & 5.1 & 0 \end{bmatrix}$$

采用二交换机制，以(4.24)式定义代价函数，定义从旅程 Ξ_i 到 Ξ_j 的转移的接受率为

$$e^{-\frac{[f(\Xi_j)-f(\Xi_i)]^+}{T}}$$

转移机制通过比较接受概率与一个来自均匀分布的随机变量的数值而实现。假定 $T_S=10$，起始旅程为 $\Xi_1=(2,1,4,3,5,6)$。二交换机制中用到的城市 X 和 Y 以及为最初 5 个可能转移而生成的随机数由下表给出。填入表中空项，亦即找出新旅程、新旅程的代价和每个旅程的接受率，并决定是否接受转移。

建议旅程	X	Y	随机数	代价 $f(\Xi_i)$	接受概率	接受转移?
$\Xi_1=(2,1,4,3,5,6)$	—	—	—		—	—
Ξ_2	1	5	0.2			
Ξ_3	1	6	0.7			
Ξ_4	2	3	0.5			
Ξ_5	6	5	0.6			
Ξ_6	3	4	0.4			

4.2 考虑一个组合优化问题，它的解答空间大小为 $|S|=6$，等价函数 $f(i)=i$，其中 $i=1,2,\cdots,6$。应用模拟退火算法，并假定从一个解答生成另一个解答的概率在整个解答空间上均匀分布。进一步假定从一个解答到另一个解答的转换遵守麦绰波里斯准则。找出模拟退火算法在 $T=5$ 时的马尔科夫链转移矩阵。

4.3 分别应用模拟退火、波尔兹曼机、高斯机和柯西机求解第 3 章中给出的霍普费尔德 10 城市 TSP。

4.4 推导(4.50)式中的 $\bar{u}_i$ 和 σ_{u_i}。

4.5 选择一个适当的 c 使得(4.50)式尽可能地接近(4.32)式。(4.32)式中 $u_i = \sum_{j \neq i} w_{ij} s_j + I_i$，而(4.50)式中 $u_i = \sum_{j \neq i} w_{ij} s_j + I_i + \eta$。提示：尝试 $c = \sqrt{2/\pi}$。

4.6 找出(4.52)式定义 u_i 的概率密度函数。

4.7 证明(4.54)式。

第5章 均场退火

第4章已经证明，如果在每个温度下状态转移的数目足够大，模拟退火算法渐近收敛于具有最小代价的解答。将模拟退火应用于霍普费尔德神经元网络而得到的随机机也展示出防止网络陷于局部最小值的优点。这是由于模拟退火算法使得随机机的状态通过扰动而演变。然而，随机机的这一长处是以变量的随机松弛过程所要求的大量运算为代价的。求取随机机在每个温度下的平衡态的随机松弛过程十分冗长。用统计物理中常用的均场近似[62][158]来取代随机神经元，可望得到一个更快达到热平衡的松弛过程。这里，随机机的随机性二值神经元被确定性连续值神经元替代，而模拟退火的随机更新过程则被一组用于更新的确定性等式替代。虽然这一近似方法不能保证找到全局最小值，但却可以用小得多的计算量找到接近最优解的很好近似。均场近似的概念并不新鲜，首先将这一概念引入神经网络以求解最优化问题的是皮特森(C. Peterson)和他的同事们(文献[128，129，130，131，132])。从那以来，已有很多关于均场退火及其实现和应用的文献发表，如[15，19，32，45，80，97，119，120，144，149，162，165，166，167，176，177]。

5.1 均场近似

类似于模拟退火，均场退火也有两个概念不同的操作：一个热静力学操作用于安排降温过程和一个确定性(而不是随机性)松弛过程用于搜索解答的均值。

可用几种不同的方法推导出均场退火的确定性松弛过程。其中最简单的途径就是用均场近似取代霍氏网中由(4.34)式给出的激活函数，亦即用随机变量均值的函数来代替随机变量函数的均值。记第 i 个神经元状态 s_i 的均值为 $v_i=<s_i>$。因此，从(4.34)式可知状态 s_i 基于激活能量 u_i 的均值可由下式计算出

$$\begin{aligned} v_i =< s_i >&= (+1)P(u_i) + (-1)(1 - P(u_i)) \\ &= 2P(u_i) - 1 \\ &= \frac{2}{1 + \exp\left(\frac{-2u_i}{T}\right)} - 1 \\ &= \frac{1 - \exp\left(\frac{-2u_i}{T}\right)}{1 + \exp\left(\frac{-2u_i}{T}\right)} \\ &= \tanh\left(\frac{u_i}{T}\right) = \tanh\left(\frac{\sum_{j\neq i} w_{ij}s_j + I_i}{T}\right) \end{aligned} \tag{5.1}$$

注意式中激活能量 u_i 是一个随机变量，它的值随着与第 i 个神经元的输入相连的其它神

经元的随机变化而变化。对于 $i=1,2,\cdots,N$，(5.1)式是一组有 N 个随机变量 s_i 的非线性等式。从这一组等式求解 s_i 即使不是不可能的，也是一个极为艰巨的任务。这也是为什么要应用均场退火的原因。如果外部输入 I_i 不是随机的，激活能量 u_i 的平均波动为

$$< u_i >=< \sum_{j\neq i} w_{ij}s_j + I_i >= \sum_{j\neq i} w_{ij} < s_j > + I_i \tag{5.2}$$

在(5.1)式中的激活能量被其均值所取代后，可得如下均场等式：

$$< s_i >= \tanh\left(\frac{< u_i >}{T}\right) = \tanh\left(\frac{\sum_{j\neq i} w_{ij} < s_j > + I_i}{T}\right) \tag{5.3}$$

因为未知变量$<s_i>$是确定性的，(5.3)式对于 $i=1,2,\cdots,N$ 所得的不等式组可以用迭代法很容易地求解。将原来的一组可怕的非线性随机方程转换为较易处理的非线性确定性等式正是应用均场近似的本意。上述均场等式亦可用鞍点展开方式导出。

5.2 鞍点展开

回想随机霍氏网在状态 S 处的稳态概率分布符合如下波尔兹曼分布：

$$\pi_T(S) = \frac{\exp\left(-\frac{E(S)}{T}\right)}{Z_T} \tag{5.4}$$

其中：S 是状态空间；

$E(S)$是网络在状态 S 的能量；

T 仍是温度的控制参数；

Z_T 是下式给出的划分函数：

$$Z_T = \sum_{S\in S} \exp\left(-\frac{E(S)}{T}\right) \tag{5.5}$$

对于一个大型的最优化问题，划分函数 Z_T 的精确计算是很费功夫的。取而代之，可用鞍点扩展[132]近似来解决这一问题。注意狄拉克(Dirac)脉冲函数 $\delta(\cdot)$可被表示为

$$\delta(x) = \frac{1}{2\pi\sqrt{-1}}\int_I e^{xy}dy \tag{5.6}$$

其中的积分是沿虚轴进行的。利用狄拉克脉冲函数的移位特性

$$f(x_0) = \int_R f(x)\delta(x - x_0)dx \tag{5.7}$$

划分函数可由下式计算

$$\begin{aligned} Z_T &= \sum_{S\in S} \exp\left(-\frac{E(S)}{T}\right) \\ &= C\sum_{S\in S}\int_R\int_I e^{-\frac{E(v)}{T}} \cdot e^{u(S-v)}dudv \\ &= C\int_R\int_I e^{-\frac{E(v)}{T}-u\cdot v} \cdot \sum_{S\in S} e^{u\cdot S}dudv \\ &= C\int_R\int_I e^{-\frac{E(v)}{T}-u\cdot v+\ln\sum_{S\in S}e^{u\cdot S}}dudv \end{aligned}$$

$$= C\int_R\int_I e^{-E_{\text{eff}}(\boldsymbol{u},\boldsymbol{v})}d\boldsymbol{u}d\boldsymbol{v} \tag{5.8}$$

其中

$$E_{\text{eff}}(\boldsymbol{u},\boldsymbol{v}) = \frac{E(\boldsymbol{v})}{T} + \boldsymbol{u}\cdot\boldsymbol{v} - \ln\sum_{S\in S} e^{\boldsymbol{u}\cdot S} \tag{5.9}$$

在统计力学中被称为有效能量，而 C 是一个复常数。虽然很难找出上述积分方程的闭解，但若对其在有效能量的一个鞍点(一个临界点)处用泰勒(Taylor)级数展开，可以证明这一积分主要由鞍点处的能量值决定。这一数值方法被称为鞍点展开。鞍点可在 $E'_{\text{eff}}(\boldsymbol{u},\boldsymbol{v})=0$ 的根中找到，因而满足

$$\frac{\partial E_{\text{eff}}}{\partial \boldsymbol{u}} = \boldsymbol{v} - \frac{\sum_{S\in s} \boldsymbol{S}\cdot e^{\boldsymbol{u}\cdot S}}{\sum_{S\in S} e^{\boldsymbol{u}\cdot S}} = 0 \tag{5.10}$$

和

$$\frac{\partial E_{\text{eff}}(\boldsymbol{u},\boldsymbol{v})}{\partial \boldsymbol{v}} = \frac{1}{T}\frac{\partial E(\boldsymbol{v})}{\partial \boldsymbol{v}} + u = 0 \tag{5.11}$$

因此

$$\boldsymbol{v} = <S_T> = \frac{\sum_{S\in S} S\cdot e^{\boldsymbol{u}\cdot S}}{\sum_{S\in s} e^{\boldsymbol{u}\cdot S}}$$

$$u = -\frac{1}{T}\frac{\partial E(\boldsymbol{v})}{\partial \boldsymbol{v}} \tag{5.12}$$

其中 $<S_T>$ 是温度 T 处 S 的热均值。

在统计物理中，$\boldsymbol{u}$ 和 $\boldsymbol{v}$ 被称为均场向量。如果状态 $S=[s_1,s_2,\cdots,s_n]^{\mathrm{T}}$ 用双极值序列表示，亦即 $s_i\in\{-1,1\}^n$，则 $\boldsymbol{v}=[v_1,v_2,\cdots,v_n]^{\mathrm{T}}$ 且

$$v_i = \frac{\sum_{s_i\in\{-1,1\}} s_i\cdot e^{u_i s_i}}{\sum_{s_i\in\{-1,1\}} e^{u_i s_i}} = \frac{e^{u_i}-e^{-u_i}}{e^{u_i}+e^{-u_i}} = \tanh(u_i) \tag{5.13}$$

其中 $\boldsymbol{u}=[u_1,u_2,\cdots,u_n]^{\mathrm{T}}$ 且 $u_i=-\frac{1}{T}\frac{\partial E(\boldsymbol{v})}{\partial v_i}$。因此，可以得到如下与(5.3)式相同的均场等式：

$$v_i = \tanh(u_i) = \tanh\left(-\frac{1}{T}\frac{\partial E(\boldsymbol{v})}{\partial v_i}\right) \tag{5.14}$$

在 3.1 节中霍氏网的能量曾由(3.8)式表示。它可用均场变量重写如下：

$$E(\boldsymbol{v}) = -\frac{1}{2}\sum_i\sum_j w_{ij}v_iv_j - \sum_i v_iI_i \tag{5.15}$$

其中 $v_i\in[-1,1]$。网络的稳定状态对应于超立方体 $\{-1,1\}^n$ 的 2^n 个角，亦即(5.16)式定义的能量函数的局部最小值。在这种情况下，均场等式变为

$$v_i = <s_i> = \tanh\left(\frac{\sum_j w_{ij}v_j + I_i}{T}\right) \tag{5.16}$$

因此，均场变量 v_i 可以用如下迭代松弛过程求出：

$$v_i^+ = \tanh\left(\frac{\sum_j w_{ij}v_j^- + I_i}{T}\right) \tag{5.17}$$

5.3 稳定性

图 5.1 所示为(5.17)式的一个实现方案。它与图 3.3 所示连续霍氏网等价，只是用 v_i 和 $\tanh(\cdot)$分别取代了 s_i 和 $\zeta(\cdot)$，并在每个积分器中都加入了控制参数$\frac{1}{T}$用来计算 v_i^- 与 v_i^+ 之间的时差。

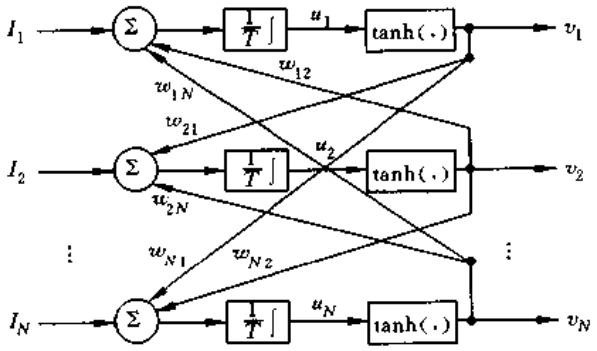


图 5.1 均场网络

均场网络的稳定性可由证明 $E(\boldsymbol{v})$是一个李雅普诺夫函数而类似地建立。正如连续霍氏网的情形那样，$E(\boldsymbol{v})$符合第 3 章中所描述的李雅普诺夫函数所需满足的条件 1 和 2。条件 3 可以如下证明：

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t}E(\boldsymbol{v}) &= \sum_i \frac{\mathrm{d}E(\boldsymbol{v})}{\mathrm{d}v_i}\frac{\mathrm{d}v_i}{\mathrm{d}t} \\ &= \sum\left(-\frac{\mathrm{d}u_i}{\mathrm{d}t}\right)\frac{\mathrm{d}v_i}{\mathrm{d}u_i}\frac{\mathrm{d}u_i}{\mathrm{d}t} \\ &= -\sum_i \frac{\mathrm{d}v_i}{\mathrm{d}u_i}\left(\frac{\mathrm{d}u_i}{\mathrm{d}t}\right)^2\end{aligned} \tag{5.18}$$

因为 $v_i=\tanh\left(\frac{1}{T}u_i\right)$是一个单调递增函数，$\frac{\mathrm{d}v_i}{\mathrm{d}u_i}\geqslant 0$。因此，$\frac{\mathrm{d}}{\mathrm{d}t}E(\boldsymbol{v})\leqslant 0$。这表示 $E(\boldsymbol{v})$对时间的导数是负半定的。由此可知，在每个给定的 T 下，均场网络一定可以收敛到一个平衡态。

5.4 均场网的参数

运行一个均场网需要事先确定如下参数以便取得较好的结果：

(1) 拉格朗日参数　一个组合最优化问题可用能量函数(或代价函数)表征为如下形式：

$$\text{能量} = w_1 \cdot \text{“代价”} + w_2 \cdot \text{“约束”} \tag{5.19}$$

其中 w_1 和 $w_2 \in R^+$ 是拉格朗日参数[①]。注意如下两个极端情况：

① 如果 $w_1 \to \infty$，(5.19)式以其第一项(代价)为主，松弛过程会给出不符合约束条件的无效解答。

② 如果 $w_1 = 0$，松弛过程可能终止于一个虽然有效但却远非全局最优的解答。

因此，根据具体问题适当确定的拉格朗日参数将在很大程度上影响解答的质量。

(2) 临界温度　临界温度定义为在平均意义上每个神经元开始显著移向两个极值 -1 和 1 时的温度。从(5.17)式显见，在高温下神经元的所有均值在 0 附近随机分布。因此，在过高温度下开始松弛过程是计算资源的一种浪费。另一方面，如果起始温度过低，松弛过程变得像统计力学中的淬火一样，神经元均值的演变将会产生很差的结果。所以，松弛过程开始时的临界温度的确定是至关重要的。在拉格朗日参数和临界温度之间通常存在某种联系。

(3) 每一温度下的状态转移　因为在每个温度下一定收敛于平衡态，每一温度下的迭代次数可由网络状态在连续两次迭代之间的相似性来确定。

(4) 终止温度或停止准则　通常，当温度到达 0 时，退火过程终止。如果进一步的松弛过程不会产生更好的解答，可以定义一个适当的停止准则以提早终止退火过程，从而节省计算资源。

(5) 降温函数　降温函数安排温度下降的流程。温度的快速下降有可能使松弛过程产出很差的结果，而缓慢下降又会减低计算效率。

最后四个参数构成了均场退火的冷却流程。这些参数的设定可用类似于第 4 章中所讨论的方式。如何更好地确定这些参数至今仍是一个挑战性的研究题目。

均场退火的步骤可以归纳如下：

(1) 根据问题设定参数

- 构造能量函数；
- 确定拉格朗日参数；
- 确定临界温度或起始温度；
- 确定在每个温度下允许的转移或迭代次数；
- 定义降温函数；
- 定义停止准则。

(2) 初始化 $v_i = \text{rand}(-\delta, \delta), i = 1, 2, \cdots, N$。其中 $\text{rand}(-\delta, \delta)$ 随机生成分布于 $-\delta$ 和 δ 之间且 $|\delta| \ll 1$ 的数值。

(3) 重复下列操作直至满足停止准则

- 在每个温度下展开松弛过程直至达到平衡态：

$$v_i^+ = \tanh\left(\frac{\sum_j w_{ij} v_j^- + I_i}{T}\right)$$

① 在把一个有约束的最优化问题变换为一个无约束最优化问题的意义上。

· 降温。

5.5 二分图示例

二分图问题已在 3.4.2 节介绍过。用均场变量改写能量函数，(3.30)式变为

$$E(\boldsymbol{v}) = \frac{N\alpha}{2} - \frac{1}{2}\sum_i \sum_{j\neq i} w_{ij} v_i v_j \tag{5.20}$$

其中 N 是节点数，α 是一个拉格朗日参数，而 $w_{ij}=c_{ij}-\alpha$。因此

$$\frac{\mathrm{d}}{\mathrm{d}v_i^-} E(\boldsymbol{v}) = -\sum_{j\neq i} w_{ij} v_j^- \tag{5.21}$$

且每个温度下的松弛过程变为

$$v_i^+ = \tanh\left(-\frac{1}{T}\frac{\mathrm{d}E(\boldsymbol{v})}{\mathrm{d}v_i}\right) = \tanh\left(\frac{1}{T}\sum_j w_{ij} v_j^-\right) \tag{5.22}$$

解决二分图问题的均场退火算法[①] 可归纳如下：

(1) 根据问题设定参数：

· 构造(5.20)式给出的能量函数；

· 确定拉格朗日参数 α；

· 确定临界温度或起始温度 T_c；

· 在每个温度下，当

$$\frac{1}{N}\sum_i |v_i^+ - v_i^-| < \varepsilon_1 \tag{5.23}$$

时转移(迭代)停止。也就是说，算法在两次迭代间神经元取值平均差小于 ε_1 时达到热平衡。

· 定义降温函数 $T_{k+L}=\beta T_k$，其中 β 一般接近于 1，且 T_k 是在第 k 次降温的温度。

· 定义停止准则

所有神经元的取值无一例外地都在$[-1,-\delta_1]$或$[\delta_1,1]$的范围内；

$\frac{1}{N}\sum_i |v_i| \geqslant \varepsilon_2$，即全部神经元中的 $100\varepsilon_2\%$ 几乎达到其稳态值$\{-1,1\}$。

(2) 初始化 $v_i=\mathrm{rand}(-\delta,\delta)$，$i=1,2,\cdots,N$。其中 $\mathrm{rand}(-\delta,\delta)$随机生成分布于 $-\delta$ 和 δ 之间且 $|\delta|\ll 1$ 的数值。

(3) 重复下列操作直至满足停止准则

· 在每个温度下展开松弛过程直至达到平衡态：

$$v_i^+ = \tanh\left(\frac{1}{T}\sum_j w_{ij} v_j^-\right)$$

· 降温。

---

① 有些参数可以凭经验设定，例如，$\alpha=1$，$\beta=0.9$，$T_c=1$，$\varepsilon_1=10^{-3}$，$\varepsilon_2=0.9$，$\delta=0.2$，$\delta_1=0.8$。

5.6 习题

5.1 在均场等式的实际应用中，可能只需要适当选择一个参数 T。研究这一情况并为 3.4.1 节所介绍的 TSP 选择一个适当的 T。这会使均场退火退化为一个简单的霍氏网吗？

5.2 对于一个同时应用均场近似方法和鞍点扩展方法的二值网络，推导出它的均场等式。

5.3 求出 $<s_i s_j>$。

5.4 用均场退火求解习题 3.5 中的二分图问题，并试验相关参数不同取值对解答的影响。

5.5 用均场退火重复习题 3.6 给出的 20 皇后问题。

5.6 用均场退火重复习题 3.7 描述的霍普费尔德实验。

第6章 遗传算法

遗传算法(GA)是模拟自然选择和遗传的随机搜索算法。密执安大学约翰·郝兰德(John Holland)提出这一算法的最初目的是研究自然系统的自适应行为并设计具有自适应功能的软件系统。近来,遗传算法作为问题求解和最优化的有效工具,引起越来越多的注意(文献[30][64][67][143])。

遗传算法是一种迭代算法。它在每一次迭代时都拥有一组解答。这组解答最初是随机生成的。在每次迭代时又有一组新的解答由模拟进化和继承的遗传操作生成。每个解答都由一个目标函数给予评价,而且这一过程不断重复,直至达到某种形式上的收敛。新的一组解答不但可以有选择地保留一些目标函数值高的旧的解答,而且可以包括一些经由其它解答结合而得的新的解答。

遗传算法的术语借鉴于自然遗传学。一个解称为一个符号串或染色体。染色体由决定其特性的基因构成,而基因又可以有称为等位基因的不同取值。目标函数称为适度函数,而一组染色体称为群体。遗传算法的一次迭代称为一代。

最简单的遗传算法包括如下组成部分:

- 一个对参数空间编码的符号串表示;
- 一个评价符号串的适度函数;
- 一组产生成新的符号串的遗传操作;
- 一组控制遗传操作的概率值。

典型的遗传算法步骤有:

(1) 初始化 随机生成一个符号串群体。

(2) 基于适度函数对符号串进行评价。

(3) 应用一组遗传操作生成一个新的符号串群体。

(4) 重复步骤(2)和(3)直到解答收敛。

遗传算法成功的关键在于符号串表示和遗传操作的设计。下一节介绍一种具有三个简单操作的遗传算法。

6.1 简单遗传操作

遗传操作通过模拟进化和继承过程而生成符号串(新的或旧的)。繁殖、交叉和突变是三个简单遗传操作,它们在实际应用中给出了很好的结果。

6.1.1 繁殖

在大自然中,生命力最强的物种征服弱小的物种以确保其生存。运用这一适者生存的

法则，繁殖操作在旧的群体中“随机”选择符号串生成一个新的群体。选择并不是完全随机的，它基于一个符号串相对于整个群体的适度值。

假定一个群体有 6 个符号串，而且它们的适度值如下：

符号串(i)	适度值(f_i)	$f_i/\sum_i f_i$
A	25	0.5
B	10	0.2
C	6	0.12
C	6	0.12
D	2	0.04
E	1	0.02
总数	50	1.0

注意，一个群体中的每个符号串不必是唯一的。$f_i/\sum_i f_i$ 被视为符号串 f_i 在下一代中存活的概率。这意味着具有较高适度值的符号串会有较大的存活机会。另外，在整个算法运行过程中，一个群体的符号串数目是一个常数。繁殖操作生成的是一个同样大小的群体。这意味着适度值较大的符号串最终会在群体中成为多数。

实现上述选择过程的一种方法是偏置轮盘。每个符号串在轮盘上占有一格，而格的大小则与符号串的适度值成正比。在选择一个新的符号串时，先转动轮盘，待轮盘停下，落在标记处的格所对应的符号串被选中。

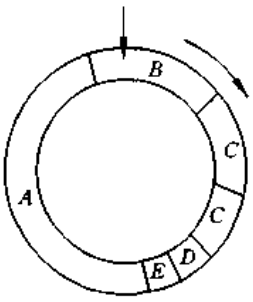


图 6.1　偏置轮盘

图 6.1勾画出了上例中的轮盘。轮盘转动6次生成一代新的群体，且符号串的期望组

符号串(i)	存活率(p_i)	期望次数($6p_i$)
A	0.5	3
B	0.2	1.2
C	0.24	1.44
D	0.04	0.24
E	0.02	0.12

合为基于期望次数[①]，新的群体可能是$\{A,A,A,B,C,C\}$。很明显，如果繁殖操作被重复运用，适度值较高的符号串最终会在整个群体中占据主导地位。由于繁殖操作并不生成新的符号串，我们需要其它操作以探究新的解答。

6.1.2 交叉

交叉操作利用了来自不同符号串的基因经由交配而混合以产生新符号串的概念。由于基因表达了符号串的特性，如果不同符号串的"好的"特性得以结合，所得符号串可能会有更好的特性。

假定一个符号串的基因被排成一条直线，则两个符号串的交叉可按如下步骤进行(见图 6.2)：

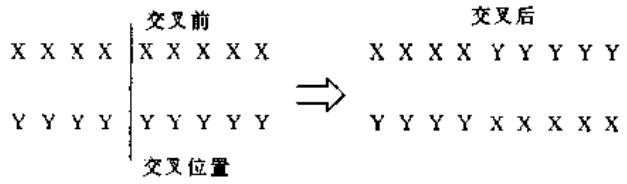


图 6.2 简单交叉操作

(1) 随机选择一个将每个符号串断开为两部分的点(截点)。

(2) 交换符号串的后一部分。

两个具有其父母双方基因成分的符号串由此生成。这一交叉操作是交换信息、生成新解的简单而有效的方法。需要注意的是，如果整个群体只有一种符号串，交叉操作不会生成任何新的符号串。可以利用突变操作来避免这种情况的发生。

6.1.3 突变

突变操作随机选取符号串中的一个基因并将其改变为一个不同的等位基因以生成一个新的符号串(见图 6.3)。它将可变性引入群体，从而提供逃脱局部最小值的手段。注意，一个仅应用突变操作的算法等同于随机搜索。

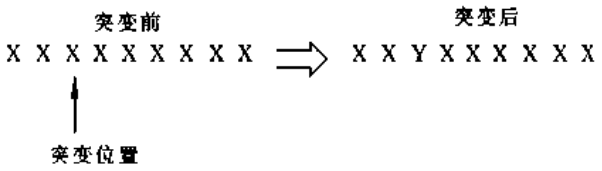


图 6.3 突变操作

因为突变操作可以有很强的破坏性，并非总要用到它，而是由一个突变概率(p_m)来控制。与此类似的，交叉操作的运用也由交叉概率(p_c)来控制。

一个简单的遗传算法可归纳如下：

---

① 选中某一符号串次数的期望值。

(1) 生成一个具有 m 个符号串的起始群体。

(2) 重复步骤(3)直至解答的适度值收敛。

(3) 生成一个新的 m 符号串群体。步骤如下：

① 应用繁殖操作两次，亦即用轮盘选出两个符号串。

② 如果 p_c>rand[0,1]，则应用交叉操作于这一对符号串。

③ 如果 p_m>rand[0,1]，则应用突变操作于这一对符号串。

④ 将生成的两个符号串加入新的群体。

下节用一个例子来说明遗传算法和遗传操作的用法。

6.2 一个示例

以求解下列函数的最大值为例。

$$f(x)=0.4-\mathrm{sinc}(4x)+1.1\mathrm{sinc}(4x+2)+0.8\mathrm{sinc}(6x-2)+0.7\mathrm{sinc}(6x-4),\ x\in[-2,2] \tag{6.1}$$

其中

$$\mathrm{sinc}(x)=\begin{cases}1 & x=0\\ \dfrac{\sin(\pi x)}{\pi x} & x\neq 0\end{cases} \tag{6.2}$$

图 6.4 给出了函数 $f(x)$在变量 x 取值[−2,2]时的曲线。f 的最大值 1.501564 对应于 $x=-0.507179$。

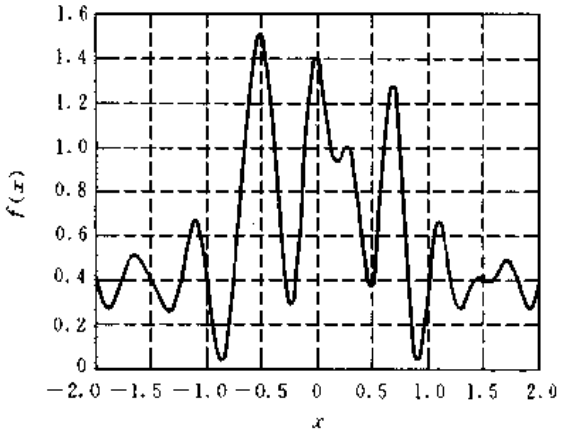


图 6.4 函数 $f(x)$在 $x\in[-2,2]$上的曲线

在运用遗传算法求解 f 的最大值时，可以将 x 的值用一个二值形式表达为二值向量。一个 16 位的二值数提供的分辨率是每位$(2-(-2))/(2^{16}-1)=0.000061$。(6.3)式将变量域[−2,2]离散化为二值数[0,65535]。

$$x=-2+\frac{4b}{65535} \tag{6.3}$$

其中 b 是[0,65535]中的一个二值数。例如，

$$v_1 = (1\,0\,1\,1\,1\,0\,1\,1\,1\,1\,1\,0\,0\,1\,1\,0) \Rightarrow x = 2.93506$$

$$v_2 = (0\,0\,1\,1\,0\,0\,1\,1\,1\,1\,1\,1\,1\,1\,1\,0) \Rightarrow x = -1.87610$$

在运用遗传算法求解(6.1)式给出的函数的最大值时，用到了繁殖、交叉和突变等三个遗传操作。

繁殖操作由上节给出的偏置轮盘实现。交叉操作将两个二值向量混合在一起，并生成两个新的二值向量。在这里我们采用最简单的交叉形式，即随机选取两相邻位之间作为截点，交换两向量在截点后的尾部以获取两个新的向量。

例如，若选取截点如下(以 ‖ 表示)：

$$v_1 = (1\,0\,1\,1\,1\,0\,1\,1\,1\,1 \,\|\, 0\,0\,1\,1\,0)$$

$$v_2 = (0\,0\,1\,1\,0\,0\,1\,1\,1\,1 \,\|\, 1\,1\,1\,1\,0)$$

则两个新的二值向量为

$$v_1 = (1\,0\,1\,1\,1\,0\,1\,1\,1\,1 \,\|\, 1\,1\,1\,1\,0)$$

$$v_2 = (0\,0\,1\,1\,0\,0\,1\,1\,1\,1 \,\|\, 0\,0\,1\,1\,0)$$

突变操作十分直截了当。给定一个二值向量，随机选取一位并将其反置即可。例如，若 v_1 中带下划线的一位被选中，则突变后的新向量为 v_1'

$$v_1 = (1\,0\,1\,1\,1\,0\,1\,1\,1\,\underline{1}\,0\,0\,1\,1\,0)$$

$$v_1' = (1\,0\,1\,1\,1\,0\,1\,1\,1\,0\,0\,0\,1\,1\,0)$$

在求解中用到了下列参数：

- 群体大小=30；
- 交叉概率=0.3；
- 突变概率=0.01。

从不同的起始群体出发，运用遗传算法100次。算法找到最优解的成功率为80%。图6.5给出了一次成功运用的收敛过程。注意，经过400次迭代，一个群体大小为30的遗传算法在终止时已经评价了400×30个二值向量(有些向量会重复出现)。

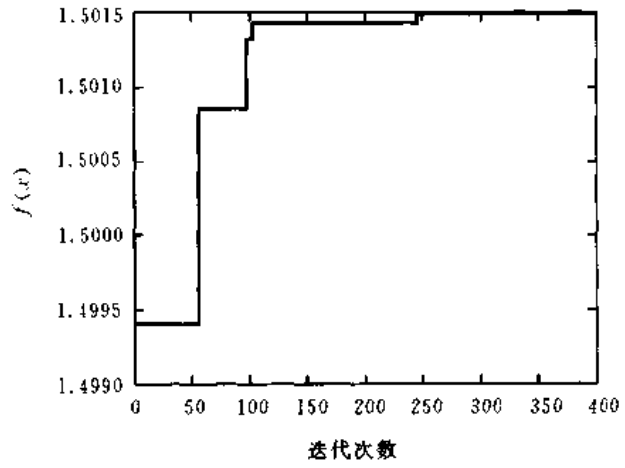


图 6.5　一次成功运用的收敛过程

6.3 为什么遗传算法会奏效?

以二值符号串表达的遗传算法的理论基础可参见文献[64][83]。分析二值符号串的重要工具是相似性模板。它揭示出存在于二值符号串之间的相似性。

在前面的叙述中,二值符号串中每个基因的取值只能是0或1。模板是二值符号串的一种扩展,其中基因的取值可为0,1或"*"。符号"*"表示一种随意的情况,也就是说基因的取值既可为0亦可为1。例如,(*,0,1,*,0,0)、(*,*,*,*,*,*)和(1,0,0,1,0,1)都是6位二值符号串的模板。如果一个基因的取值不是"*",我们称其为确定位;否则称为不定位。

当一个模板和一个符号串的每个基因都匹配("*"与0和1都匹配)时,称这对模板和符号串匹配。因此,模板(*,0,1,*,0,0)与4个符号串(0,0,1,0,0,0)、(0,0,1,1,0,0)、(1,0,1,0,0,0)和(1,0,1,1,0,0)都匹配。同理,模板(*,*,*,*,*,*)与所有64个6位二值符号串匹配,而模板(1,0,0,1,0,1)仅与符号串(1,0,0,1,0,1)匹配。

对于n位符号串,一共存在3^n个模板,且每个符号串可与2^n个模板相匹配。在一个拥有m个符号串的群体中,可与这m个符号串相匹配的模板数目在2^n到$m2^n$之间。

模板可以两个值来刻画:阶和定长。模板S的阶是其中0和1的出现次数,以$o(S)$表示。例如,$o((*,0,1,*,0,0))=4$,$o((*,*,*,*,*,*))=0$,而$o((1,0,0,1,0,1))=6$。模板S的定长是其中第一个和最后一个确定位之间的距离,以$\delta(S)$表示。例如,$\delta((*,0,1,*,0,0))=4$,$\delta((*,*,*,*,*,*))=0$,而$\delta((1,0,0,1,0,1))=5$。

繁殖、交叉和突变等三个遗传操作对一个符号串群体的影响可以利用每一代中能与一个模板匹配的符号串个数的期望值来加以考察。在此,采用下列符号:

- n是一个二值符号串的位数。
- $P(t)$是在t时刻的符号串群体。
- m是$P(t)$中符号串的个数,亦即群体大小。
- $f(s)$是符号串$s\in P(t)$的适度值。
- $F(t)=\sum_i f(s_i)$,$s_i\in P(t)$是$P(t)$中所有符号串的总适度值。
- $\overline{F(t)}=\sum_i f(s_i)/m$是$P(t)$中所有符号串的平均适度值。
- $\overline{F(S)}=\sum_i f(s_i)$[①],$s_i$与$S$匹配,是$P(t)$中所有与模板$S$匹配的符号串的平均适度值或模板$S$的平均适度值。
- $\zeta(S,t)$是$P(t)$中与模板S匹配的符号串个数的期望值。
- p_c是交叉概率。
- p_m是突变概率。

对于繁殖,一个$P(t)$中的符号串s_i存活到下一代的概率等于$f(s_i)/F(t)$。因此,对于一个拥有m个符号串的群体,$P(t+1)$中与模板S匹配的符号串个数的期望值为

① 译者注:此表达式应除以$P(t)$中与S匹配的s_i的数量。

$$\zeta(S,t+1)=m\zeta(S,t)\frac{\overline{F(S)}}{F(t)} \tag{6.4}$$

$$=\zeta(S,t)\frac{\overline{F(S)}}{\overline{F(t)}} \tag{6.5}$$

这就是说，与一个特定模板相匹配的符号串个数随着该模板的平均适度值与群体的平均适度值的比率而改变。若比率大于1，与“高于平均”模板匹配的符号串个数增长；若比率小于1，与“低于平均”模板匹配的符号串个数减少。如果比率长期保持大于1，则与“高于平均”模板匹配的符号串个数呈指数增长(见习题6.5)。与此类似，如果比率长期保持小于1，则与“低于平均”模板匹配的符号串个数呈指数减少。

交叉和突变操作通过改变符号串中基因的取值来生成新符号串。对于一个能与交叉或突变操作前后的符号串都匹配的模板来说，它的确定位在该操作实施后应保持不变。

交叉将两个符号串同时截为两段并交换它们的尾部。当截点处于两个确定位之间时，这使得模板很有可能被破坏[①]。既然截点可被选在$n-1$个可能位置中的任意一个，一个模板S被破坏的概率是$\delta(S)/(n-1)$。与交叉概率p_c一并考虑，模板S被破坏的概率是$p_c\delta(S)/(n-1)$。因此，模板S存活的概率是

$$1-p_c\frac{\delta(S)}{n-1} \tag{6.6}$$

对于与一个模板S相匹配的符号串个数的期望值，交叉的效果为

$$\zeta(S,t-1)\geqslant\zeta(S,t)\left(1-p_c\frac{\delta(S)}{n-1}\right) \tag{6.7}$$

突变只改换符号串的一位。如果被变更的是一个不定位，则模板可以存活。由此可见，模板S在突变操作后仍存活的概率为

$$(1-p_m)^{o(S)} \tag{6.8}$$

对于$p_m\ll1$，(6.8)式可用$1-o(S)p_m$来近似。

将繁殖、交叉和突变的影响结合在一起，我们得到

$$\begin{aligned}\zeta(S,t+1)&\geqslant\zeta(S,t)\frac{\overline{F(S)}}{\overline{F(t)}}\left[1-p_c\frac{\delta(S)}{n-1}\right][1-p_mo(S)]\\&\geqslant\zeta(S,t)\frac{\overline{F(S)}}{\overline{F(t)}}\left[1-p_c\frac{\delta(S)}{n-1}-p_mo(S)\right]\end{aligned} \tag{6.9}$$

这就引出了下列定理。它指出：与一个“高于平均”模板相匹配的字符串个数的期望值呈指数增长；交叉和突变对短的和低阶的模板没有影响。

定理6.1　遗传算法的基本定理或模板定理　短的、低阶的、高于平均的模板在后代中获得呈指数增长的扩散。

短的、低阶的、高于平均的模板被称为结构单元并为遗传算法的寻求对象。结构单元之所以能在后代中获得呈指数增长的扩散从而得到最优解，其原因可参考统计决策理论中k个武装盗匪问题[83]。结合不同符号串中“好的”基因以获取更好基因的思想在直觉上是令人感兴趣的，并可作为假定。

---

① 如果新的尾部具有与旧的尾部完全相同的确定位，则此模板将存活。

假定 6.1　结构单元假定　遗传算法通过结构单元的并列来寻求接近最优的性能。

虽然这一假定尚未被证明，大量应用的实验结果都支持这一假定的有效性。有兴趣的读者可以在文献[64]中找到有关资料。

6.4　其它遗传操作

对于许多问题，二值符号串不能被用来表示解空间。例如，TSP 的一个旅程可以很自然地表示为一个 n 城市的排列，但二值符号串的交叉和突变操作不能用于其上。

一组称为重排操作的新的操作可用来处理这类表示问题[50][65]。它包括三种操作：部分匹配交叉(PMX)、有序交叉(OX)、循环交叉(CX)。

一种排列的表示要求每个等位基因都要在符号串中出现但只出现一次。PMX 操作[65]要求选取两个截点以便确定一个匹配段。例如，假定旅程 1 和旅程 2 是两个 10 城市 TSP 中的旅程，其中截点以 ‖ 表示：

$$\text{旅程 } 1 = (J\ H\ D \parallel E\ F\ I \parallel G\ C\ B\ A)$$

$$\text{旅程 } 2 = (H\ G\ E \parallel B\ C\ J \parallel I\ A\ D\ F)$$

根据旅程 1 和旅程 2 中两个截点之间的中间段给出的映射关系可得

$$\text{旅程 } 1' = (I\ H\ D \parallel B\ C\ J \parallel G\ F\ E\ A)$$

$$\text{旅程 } 2' = (H\ G\ B \parallel E\ F\ I \parallel J\ A\ D\ C)$$

这是两个新的合法旅程。它们都包含有旧旅程的成分。

CX 操作[122]通过找到两个符号串构成的循环而生成新的符号串。利用上例中 PMX 操作用到的两个旅程，我们可以构成如下排列：

$$\begin{pmatrix} J\ H\ D\ E\ F\ I\ G\ C\ B\ A \\ H\ G\ E\ B\ C\ J\ I\ A\ D\ F \end{pmatrix}$$

第一个新旅程通过首先选取旅程 1 中的第一个城市而构成

$$\text{旅程 } 1' = (J\ ?\ ?\ ?\ ?\ ?\ ?\ ?\ ?\ ?)$$

由于 J 在排列中对应于 H，我们可以将其加入旅程

$$\text{旅程 } 1' = (J\ H\ ?\ ?\ ?\ ?\ ?\ ?\ ?\ ?)$$

继续这一映射过程，H 对应于 G，G 对应于 I，I 对应于第一个选中的城市 J，从而完成了一个循环。这时，已经得到的部分旅程变为排列的一个循环

$$\text{旅程 } 1' = (J\ H\ ?\ ?\ ?\ I\ G\ ?\ ?\ ?)$$

其余的空位可由旅程 2 中相应位置上的城市填充。这样，新的旅程变为

$$\text{旅程 } 1' = (J\ H\ E\ B\ C\ I\ G\ A\ D\ F)$$

第二个新旅程可选用旅程 2 中的第一个城市为起始，用类似方式通过找出循环而构造出来。第二个被生成的新旅程为

$$\text{旅程 } 2' = (H\ C\ D\ E\ F\ J\ I\ C\ B\ A)$$

OX 操作[50]将在 7.4 节中描述。对这三种重排操作的理论和实验结果的对比可参见文献[122]。

遗传算法在 TSP、模式匹配问题、多处理机任务调度问题和作业调度问题上的应用

将分别在第 7,9,10 和 11 章中讨论。

6.5 习题

6.1 证明一个 n 位二值符号串可由 2^n 模板匹配。

6.2 用一个 20 位的二值符号串表示和简单遗传算法求解将下式最小化的值。

$$f(x) = |x|^{\frac{1}{2}}\sin(4\pi x),\ x \in [-2,2]$$

6.3 应用带 PMX 操作的遗传算法求解习题 3.7 给出的 TSP。

6.4 证明对于 n 位二值符号串,一共存在 3^n 模板。

6.5 证明一个 n 位二值符号串可被 2^n 个模板匹配。

6.6 推导(6.4)式。

6.7 证明若比率 $\overline{F(S)}/\overline{F(t)}=1+\varepsilon,\varepsilon>0$,则

$$\zeta(S,t) = \zeta(S,0)(1+\varepsilon)^t$$

第7章 旅行商问题

在解决TSP问题时，霍氏网因以大百分比生成无效解答而著称[11][138]。这是由于霍普费尔德能量函数，即等式(3.22)的参数选择得不好而造成的。这一章考察霍氏网经常失败而不能生成有效解答的原因，并给出改进的方法。前面各章介绍的不同算法在解决TSP中的应用也将在本章加以比较。选用TSP进行算法比较是因为它有大量潜在应用，已被许多研究人员用为测试基准，并是一个众所周知的重要NP完全问题。

7.1 为什么霍氏网经常不能生成有效解答？

这一节考察霍氏网在解决TSP时的特点和动态特性以说明上述缺陷的原因。最关键的分析(引自[11])是基于对霍氏网稳态、霍普费尔德能量函数以及连接性矩阵的特征值之间关系的理解。回想第三章中给出的霍普费尔德能量函数

$$E(S) = -\frac{1}{2}S^{\mathrm{T}}WS - I^{\mathrm{T}}S \tag{7.1}$$

对称的$N\times N$连接矩阵W的特性完全由其本征值λ_i和正交本征向量e_i决定，其中$i=1,2,\cdots,M$，而M是不相重复的本征值个数，且$M\leqslant N$。一些本征值会退化为一个单一的值(即多重本征值)。在这种情况下，与此本征值相对应的就不再是一个本征向量了。取而代之的是一个子空间，称为该本征值的本征空间。另外，W可包含一个对应于退化到零的本征值的一个零空间。因此，每个向量都可以用W的本征向量/本征值以及零空间来表达：

$$S = \sum_{i=1}^{M} s^i + q \tag{7.2}$$

其中$s_i=Se^i=\lambda_i e^i$，$q\in N(W)$，且$N(W)$是W的零空间。同理，

$$W = \sum_{i=1}^{M} \lambda_i e^i (e^i)^{\mathrm{T}} \tag{7.3}$$

为简化起见，忽略外部输入，可得

$$E(S) = -\frac{1}{2}\sum_{i=1}^{M}\lambda_i S^{\mathrm{T}} e^i (e^i)^{\mathrm{T}} S = -\frac{1}{2}\sum_{i=1}^{M}\lambda_i \| S\cdot e^i \|^2 = -\frac{1}{2}\sum_{i=1}^{M}\lambda_i \gamma_i^2 \tag{7.4}$$

注意，当

(1) γ_i对于$\lambda_i>0$取最大值；

(2) γ_i对于$\lambda_i<0$被置零。

能量函数$E(S)$最小化。

如果网络按照上述原则改变S，则S移动的方向会越来越偏于较大的正数本征值。由于S是以单位立方体为约束的，当$S=\gamma_{\max}e^{\max}$时能量最小，其中$e^{\max}$是对应于最大正数本征值的本征向量。

让我们回到 3.4.1 节中讨论过的 TSP 的连接矩阵 W。回想每个神经元变量都是双下标的，而连接矩阵的元素都是四下标的。然而，从(7.1)式可知，S 是一个向量而 W 是一个矩阵。这是否暗指某种表示方式的不一致呢？事实上，这纯粹只是表达上的取舍不同而已。只要知道了解答向量 S 可由一行行依次排列的 $\{s_{Xi}\}$ 如下构成，本征值分析同样可以适用。

$$S=\begin{bmatrix} s_{11} \\ s_{12} \\ \vdots \\ s_{1N} \\ s_{21} \\ \vdots \\ s_{2N} \\ \vdots \\ s_{NN} \end{bmatrix} \tag{7.5}$$

在(3.22)式的第三项中不多不少正好有对应于旅程中 N 个城市的 N 个神经元处于激活状态，根据这个前提，(3.22)～(3.24)式给出的霍普费尔德能量函数可由 N 取代 $\tilde{N}$ 而改写为

$$\begin{aligned} E=&\frac{A}{2}\sum_X\sum_i\sum_{j\neq i}s_{Xi}s_{Xj}+\frac{B}{2}\sum_i\sum_X\sum_{Y\neq X}s_{Xi}s_{Yi}\\ &+\frac{C}{2}\Big(\sum_X\sum_i s_{Xi}-N\Big)^2\\ &+\frac{D}{2}\sum_X\sum_{Y\neq X}\sum_i d_{XY}s_{Xi}(s_{Y,(i+1)}+s_{Y,(i-1)})\\ =&-\frac{1}{2}\sum_X\sum_Y\sum_i\sum_j w_{Xi,Yj}s_{Xi}s_{Yj}-C\sum_X\sum_i Ns_{Xi}+\frac{C}{2}N^2 \end{aligned} \tag{7.6}$$

其中

$$w_{Xi,Yj}=-A\delta_{XY}(1-\delta_{ij})-B\delta_{ij}(1-\delta_{XY})-C-Dd_{XY}(\delta_{j,i+1}+\delta_{j,i+1}) \tag{7.7}$$

基于第 3 章中所述的原因，(7.6)式中的最后一项可被忽略。连接矩阵 W 的前三项对应于约束，而最后一项则对应于 TSP 的代价。一个无效解答使得(7.7)式的前三项中有一项或多项异常。因为前两项引入的是分别作用于 s_{Xi} 的行和列上的相似的约束，它们应有相同的权值，因此 $A=B$。另外，由于只有前三项决定是否能得到有效解答，本征值分析最初是在去掉最后一项的连接矩阵上进行。由这些假定出发，连接矩阵变为

$$w_{Xi,Yj}=-A\delta_{XY}(1-\delta_{ij})-A\delta_{ij}(1-\delta_{XY})-C \tag{7.8}$$

可以证明(7.8)式给出的连接矩阵 W 具有下列三个不同的本征值：

(1) $\lambda_1=-CN^2-2A(N-1)$ 对应的本征向量为

$$e^1=\begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}$$

(2) $\lambda_2=2A$ 对应于一个本征空间，称为有效子空间。亦即每一个有效解答都属于这一本征空间。

(3) $\lambda_3=-A(N-2)$对应于一个同时正交于 e^1 和有效子空间的本征空间。

7.5 节给出了对本征值分析的概述，λ_1 的推导则作为 7.6 节中的例子。λ_2 和 λ_3 的推导则留作习题(7.2 和 7.3)。

7.1.1 霍氏网的动力学

霍氏网在解决 TSP 时的动态特性取决于下式：

$$\dot{\boldsymbol{u}}=\boldsymbol{WS}+\boldsymbol{I} \text{ 或} \frac{\mathrm{d}}{\mathrm{d}t}ux_i=\sum_Y\sum_j w_{Xi,Yj}s_{Yj}+CN \tag{7.9}$$

其中

$$s_{Xi}=\zeta(u_{Xi})=\frac{1}{2}\left(1+\tanh\left(\frac{u_{Xi}}{\beta}\right)\right) \tag{7.10}$$

霍普费尔德和坦客[86]选用的是 $\beta=0.05$。在揭示出 $\boldsymbol{W}$ 的本征值和本征空间与外部输入 $\boldsymbol{I}$ 是怎样一起影响着 $\dot{\boldsymbol{u}}$ 使 $\boldsymbol{S}$ 从超立方体中的一个点转移到一个最终解答的移动方向。就有可能预计网络到达有效解答的条件。因此，拉格朗日参数可被优化以确保网络收敛于有效解答。对(7.10)式进行泰勒级数展开可得

$$\begin{aligned}s_{Xi}=\zeta(u_{Xi})&=\frac{1}{2}\left(1+\tanh\left(\frac{u_{Xi}}{\beta}\right)\right)\\&=\frac{1}{2}\left(1+\frac{u_{Xi}}{\beta}-\frac{1}{3}\left(\frac{u_{Xi}}{\beta}\right)^3+\frac{2}{15}\left(\frac{u_{Xi}}{\beta}\right)^3-\cdots\right)\end{aligned} \tag{7.11}$$

当$\frac{u_{Xi}}{\beta}$较小时，

$$2s_{Xi}=2\zeta(u_{Xi})\approx 1+\frac{u_{Xi}}{\beta} \tag{7.12}$$

它是一个线性方程，且有

$$2\frac{\mathrm{d}s_{Xi}}{\mathrm{d}t}\approx\frac{1}{\beta}\frac{\mathrm{d}u_{Xi}}{\mathrm{d}t}\Rightarrow\dot{u}_{Xi}\approx 2\beta\dot{s}_{Xi} \tag{7.13}$$

$$2\beta\frac{\mathrm{d}s_{Xi}}{\mathrm{d}t}=\sum_Y\sum_j w_{Xi,Yj}s_{Yj}+CN \text{ 或 } 2\beta\dot{\boldsymbol{S}}=\boldsymbol{WS}+CN\boldsymbol{e}^1 \tag{7.14}$$

将解答向量 $\boldsymbol{S}$ 分解为三项：有效空间项 $\boldsymbol{S}^{\mathrm{val}}$，无效空间项 $\boldsymbol{S}^{\mathrm{inv}}$以及沿 $\boldsymbol{e}^1$ 方向的 $\boldsymbol{S}^1$ 项：

$$\boldsymbol{S}=\boldsymbol{S}^1+\boldsymbol{S}^{\mathrm{val}}+\boldsymbol{S}^{\mathrm{inv}} \tag{7.15}$$

其中

$$\boldsymbol{S}^1=(\boldsymbol{S}\hat{\boldsymbol{e}}^1)\hat{\boldsymbol{e}}^1,\ \hat{\boldsymbol{e}}^1=\frac{1}{N}[1\quad 1\quad\cdots\quad 1]^{\mathrm{T}},$$

$\boldsymbol{S}^{\mathrm{val}}\in$ 有效子空间，且

$\boldsymbol{S}^{\mathrm{inv}}\in$ 无效空间

因此，

$$\begin{aligned}\boldsymbol{WS}&=\lambda_1\boldsymbol{S}^1+\lambda_2\boldsymbol{S}^{\mathrm{val}}+\lambda_3\boldsymbol{S}^{\mathrm{inv}}\\2\beta\dot{\boldsymbol{S}}&=\boldsymbol{WS}+CN\boldsymbol{e}^1\end{aligned} \tag{7.16}$$

$$
\begin{aligned}
&= \lambda_1 S^1 + \lambda_2 S^{val} + \lambda_3 S^{inv} + CN^2 e^1 \\
&= [CN^2 + \lambda_1 (S\hat{e}^1)]\hat{e}^1 + \lambda_2 S^{val} + \lambda_3 S^{inv} \\
&= [CN^2 - (CN^2 + 2A(N-1))(S\hat{e}^1)]\hat{e}^1 \\
&\quad + 2AS^{val} - A(N-2)S^{inv}
\end{aligned} \tag{7.17}
$$

如果 $\dot{S}$ 使得 S 向有效子空间移动而离开 $\hat{e}^1$ 和无效空间，则最终解答很可能会在有效空间。换句话说，

(1) $\dot{S}$ 的 $\hat{e}^1$ 项将 S 移向如下超平面：

$$
CN^2 - (CN^2 + 2A(N-1))(S\hat{e}^1) = 0 \tag{7.18}
$$

迫使最终解答处于这一超平面。

(2) 因为 $\lambda_2 = 2A > 0$，$\dot{S}$ 使得 S 向其有效子空间项靠近。这使得该项的幅值增大。

(3) 因为 $\lambda_3 = -A(N-2)$，$\dot{S}$ 使得 S 向着远离其无效空间项的方向移动。因而该项的幅值会减小。

上述步骤可以确保霍氏网的能量最小化。回想霍氏网的能量函数为

$$
\begin{aligned}
E(S) &= -\frac{1}{2} S^T W S - I^T S \\
2E(S) &= -\lambda_1 \| S^1 \|^2 - \lambda_2 \| S^{val} \|^2 - \lambda_3 \| S^{inv} \|^2 - 2CN^2 (\hat{e}^1 S) \\
&= -\lambda_1 \| S^1 \|^2 - 2CN^2 (\hat{e}^1 S) - 2A \| S^{val} \|^2 + A(N-2) \| S^{inv} \|^2 \\
&= -\lambda_1 \left(\| S^1 \| + \frac{CN^2}{\lambda_1} \right)^2 + \frac{(CN^2)^2}{\lambda_1} - 2A \| S^{val} \|^2 + A(N-2) \| S^{inv} \|^2 \\
&= (CN^2 + 2A(N-1)) \left(\| S^1 \| + \frac{CN^2}{\lambda_1} \right)^2 + \frac{(CN^2)^2}{\lambda_1} \\
&\quad - 2A \| S^{val} \|^2 + A(N-2) \| S^{inv} \|^2
\end{aligned} \tag{7.19}
$$

由于 A，C 和 N 都是正数，$E(S)$ 在如下情况下最小化：

$$
\| S^1 \| = -\frac{CN^2}{\lambda_1} = \frac{CN^2}{CN^2 + 2A(N-1)} \tag{7.20}
$$

$$
\| S^{val} \| \to \infty \tag{7.21}
$$

$$
\| S^{inv} \| \to 0 \tag{7.22}
$$

这与前面的讨论是一致的。显然，当 $|u|$ 变大时，(7.13)式所给出的近似不再有效。然而，在 $|u|$ 增大到足以破坏这种近似之前，$|u|$ 在多数情况下应该已经大致指向超立方体中处于有效子空间的顶点。为了确保这一情况的出现，我们可以从一个使得这种近似即(7.13)式有效的小幅值 S 出发，然后将其推向有效空间。当 $|u|$ 变大时，非线性迫使网络稳定在有效空间的超立方体的顶点上。

当 S 向超平面移动时，由(7.18)式可得

$$
\begin{aligned}
&(S\hat{e}^1)(CN^2 + 2A(N-1)) = CN^2 \\
\Rightarrow &(Se^1) \left(CN + \frac{2A(N-1)}{N} \right) = CN^2 \\
\Rightarrow &(Se^1) = \frac{CN^2}{CN + \frac{2A(N-1)}{N}} = \frac{N}{1 + \frac{2A(N-1)}{CN^2}}
\end{aligned} \tag{7.23}
$$

(7.23)式将稳态下被激活的神经元个数量化了。由于 $A>0$ 且 $C>0$，被激活的神经元个数小于 N。这就满足了由(7.6)式中能量函数第三项给出的前提条件：网络的最终解答有刚好 N 个神经元被激活。这就是霍氏网经常无法收敛于有效解答的主要原因。这也是为什么霍普费尔德和坦客在解决他们的 10 城市 TSP 时选用 $\tilde{N}=15$ 而不是 $N=10$ 的原因。网络不能到达有效解答的另外一个原因是 λ_1 和 λ_3 的限制效应与外部输入 CN 之间以及它们与 λ_2 的扩张效应之间平衡得不好。换句话说，参数 A 与 C 选得不好。

7.1.2 拉格朗日参数的另一种表达方式

霍普费尔德和坦客在文献[86]中所选用的拉格朗日参数是通过实验获得的。不过，基于前述分析，可以开发出一种更稳健的方法以获取这些参数的值。将 W 和 I 影响网络动态特性的成分分离出来，就可以推导出一个能够确保网络收敛于有效解答的拉格朗日参数的数学表达式。网络的动态特性在相当大的程度上是由具有如下效应的三个本征值 λ_1，λ_2 和 λ_3 来决定的。

• $\lambda_1=-CN^2-2A(N-1)$ 在理想情况下将 S 限制在由 $Se^1=N$ 决定的子空间中。然而，从(7.23)式可知，$Se^1\neq N$。因此，为使 $Se^1=N$，λ_1 应该刚好等于 $-CN^2$。这可由给 W 的每一项加上 $2A(N-1)/N^2$ 而得到。

• 正如(7.19)～(7.22)式所显示的，$\lambda_2=2A$ 具有将 S 经由有效子空间移出超立方体边的扩展效应，而 $\lambda_3=-A(N-2)$ 则具有通过将 S 从无效空间“反弹”回来从而将其约束于有效空间的限制效应。

为了引入一个附加的自由度以便更好地协调上述效应[11]，从 W 的主对角线上再额外减去一个常数 $2A_1$，相当于从 W 的每个本征值减去 $2A_1$。然而，要保证 $\lambda_1=-CN^2$ 必须在 W 的每一项上进一步加上 $\frac{2A_1}{N^2}$。经过如上调整，连接矩阵 W 被变换为

$$\begin{aligned} w_{Xi,Yj}=&-A\delta_{XY}(1-\delta_{ij})-A\delta_{ij}(1-\delta_{XY})-2A_1\delta_{XY}\delta_{ij}\\ &-C+\frac{2(AN-A+A_1)}{N^2}-Dd_{XY}(\delta_{j,i+1}+\delta_{j,i-1}) \end{aligned} \tag{7.24}$$

且 I_{Xi} 总被置为 CN，其中 N 是城市个数。另外，$B=A$。相应的本征值为

$$\begin{aligned} \lambda_1&=-CN^2<0\\ \lambda_2&=2(A-A_1)>0\\ \lambda_3&=-AN+2(A-A_1)<0 \end{aligned}$$

像以前一样，这些本征值的正负号应该是一样的。一般来讲，由于解答在不受“限制”时变为无效，限制效应应该比扩展效应要求更严格。因此，埃亚(Aiyer)等人在文献[11]中建议 $|\lambda_1|\approx 160|\lambda_2|$ 和 $|\lambda_3|\approx 160|\lambda_2|$。到目前为止，我们只讨论了与“约束”条件有关的拉格朗日参数。如果不强行引入“代价”项，最终解答将是有效的，但却可能与最优解答南辕北辙。因为具有最小代价的旅程通常并非有效，这一代价项无疑将在网络的动态特性中引入使得 S 离开有效空间的分量。因此，D 的取值应使其影响远小于限制效应 λ_1，λ_3 和 I。通过实验，埃亚等[11]建议 $D=|\lambda_3|/80$。在连接矩阵本征值的大小与步进值 Δt 的大小之间需要有所权衡。实验数据表明，Δt 的值应该小到足以使得 u 在每一步(迭代)中的任何改变都小

于 5%。总之，

$$A_1 \approx A\left(1 - \frac{N}{322}\right) \tag{7.25}$$

$$C \approx \frac{A}{N} \tag{7.26}$$

$$D \approx \frac{AN}{80} \tag{7.27}$$

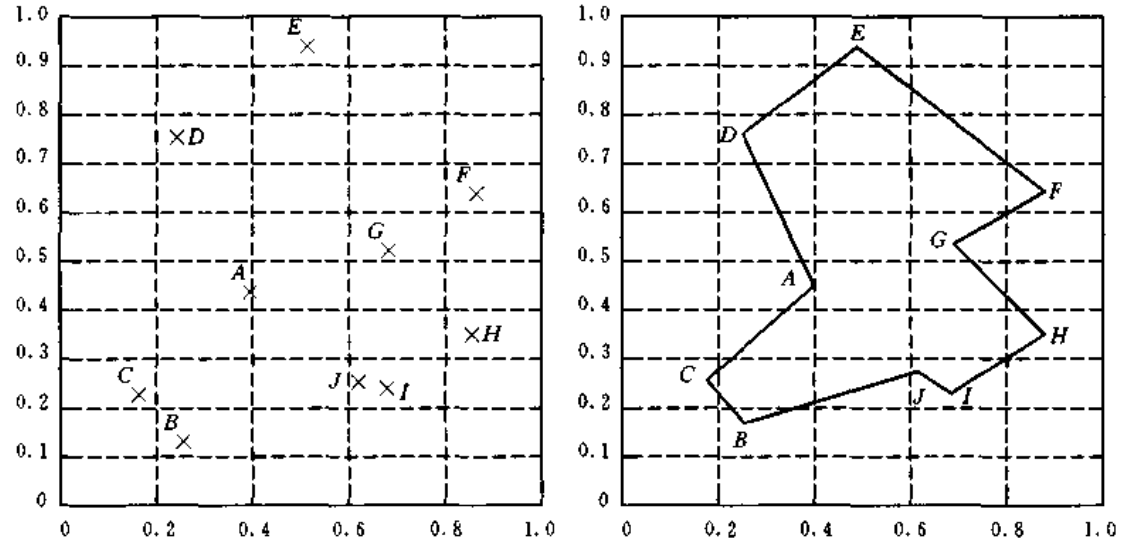
埃亚等[11]建议的用于求解 10 城市和 50 城市 TSP 的参数分别为

10 城市：$A=8;A_1=7.75;C=0.80;D=1;\Delta t=0.02$。

50 城市：$A=8;A_1=7.75;C=0.16;D=5;\Delta t=0.05$。

7.1.3 霍普费尔德的 10 城市问题

回顾第 3 章介绍的霍普费尔德 10 城市 TSP。图 7.1(a)给出了这 10 个城市的分布，而图 7.1(b)则给出了对应于最短旅途 2.691 的旅程。由于拉格朗日参数选择得不适当，正如读者在解决习题 3.7 时已经体会到的，这一霍氏网以很大百分比生成无效解答。使用上一小节中建议的求解 10 城市问题的参数，生成的解答几乎 100%有效。为了验证这些参数和(7.13)式给出的线性近似，这里介绍四组对应于四种非线性的模拟实验。其中一个非线性是如下定义的软限幅：



(a) 10城市的布局　　(b) 最短旅程

图 7.1　霍普费尔德的 10 城市问题及其最短旅程

$$\zeta(x) = \begin{cases} x + 0.5 & |x| \leqslant 0.5 \\ 1 & x > 0.5 \\ 0 & x < -0.5 \end{cases} \tag{7.28}$$

它是线性化近似的一种合理选择。另外三个非线性对应于令(7.11)式中 $\beta=0.02,0.3$ 和 2 所得的 Sigmoid 函数。这四个非线性的比较可见图 7.2。对于每个非线性，从随机起始条

件出发模拟运行 1000 次。每次模拟都在下列条件之一满足时终止：

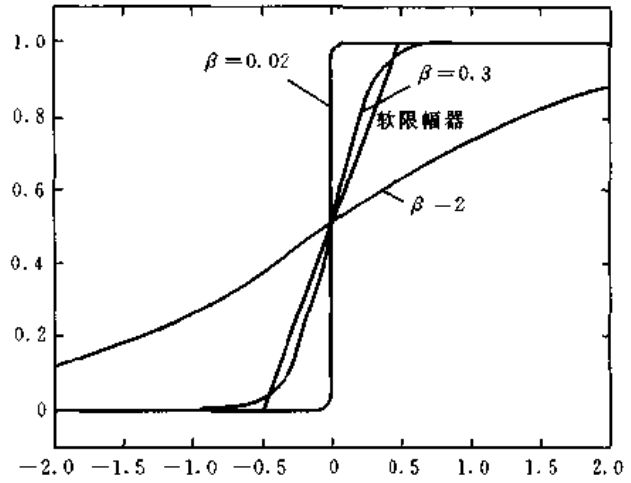


图 7.2 四种非线性

(1) 霍氏网中的每个神经元无一例外地取值在[0.9,1]或[0,0.1]之中，分别对应于神经元处于“激活”(取值落在[0.9,1]中)或“抑制”(取值落在[0,0.1]中)的状态。

(2) 模拟运行已经迭代了多于 100000 次扫描。

表 7.1～7.3 分别总结了软限幅以及对应于 $\beta=0.3$ 和 0.02 的 Sigmoid 非线性的模拟运行结果。表中，出现次数指每 1000 次模拟中出现的次数，最小次数指获得解答所用的最小扫描次数，最大次数指获得解答所用的最大扫描次数，平均次数指获得解答所用的平均扫描次数，标准偏差指扫描次数的标准差，而旅程自然是指访问城市的顺序。图 7.3 中的直方图描述了选中旅程的出现频率随旅途长度的变化。在 $\beta=2$ 的 Sigmoid 非线性情况下，网络给不出任何有效解答。从图 7.2 可以看到，这是因为网络中的神经元无法收敛于它们的稳态(“激活”和“抑制”)。相反，利用软限幅的霍氏网取得 100%的有效解答，其中 76.7%对应于图 7.1(b)所示最短旅程，且网络达到最佳解答所需的平均扫描次数为 464。其它有关软限幅霍氏网的统计数据可见表 7.1。软限幅霍氏网在四个非线性中的出众表现归功于软限幅与线性化近似极度相似。利用 $\beta=0.3$ 的 Sigmoid 非线性，网络生成的有效解答占 99.9%，其中 55.5%获得最短旅程。当 $\beta=0.02$ 时，Sigmoid 非线性接近硬限幅因而收敛得更快。在这种情况下，网络生成的有效解答占 58.2%，其中 1.9%获得最短旅程。图 7.4 给出了一些不同的旅程。

表 7.1 利用软限幅求解 10 城市问题的统计

100%有效解答	旅程长度	出现次数	最小次数	最大次数	平均次数	标准偏差	旅　程
最优	2.691	767	299	1366	464	127	*BCADEFGHIJ*
次优	2.752	73	366	1423	640	239	*BCADEGFHIJ*
第三	2.769	5	475	616	547	57	*BADEFGHIJC*
最差	3.658	1	670	670	670	0	*ACBJEFGHID*

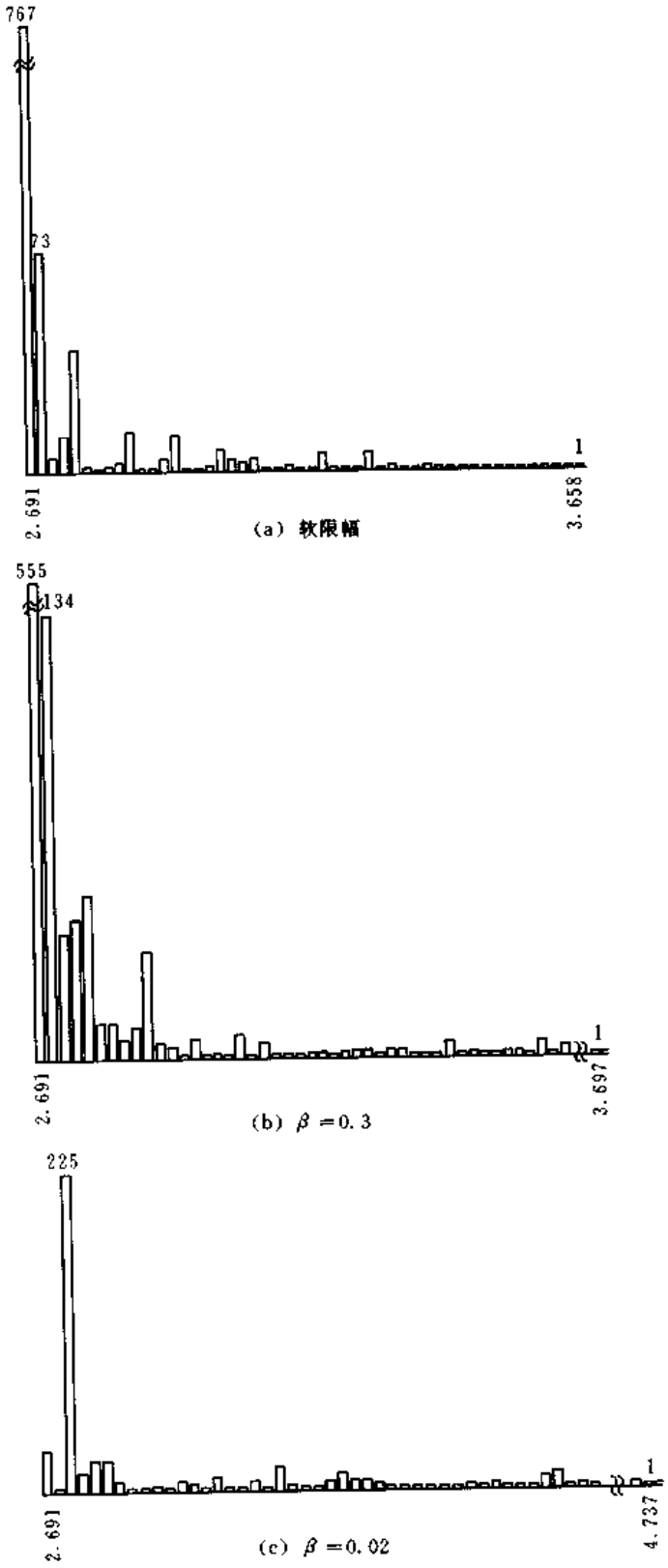
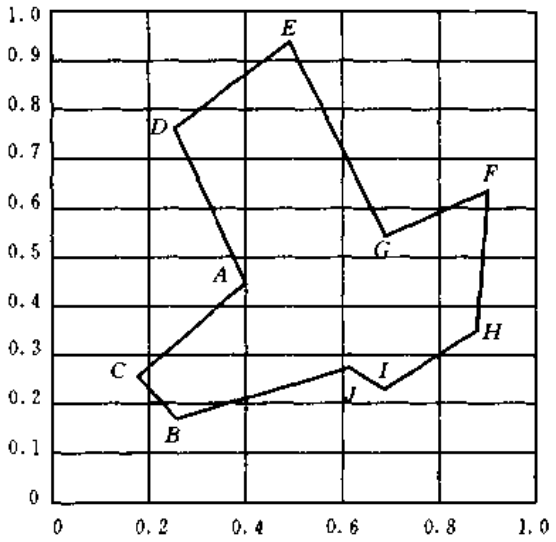
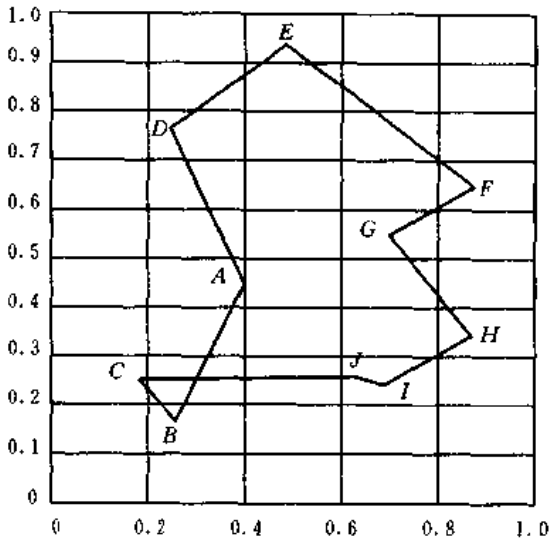


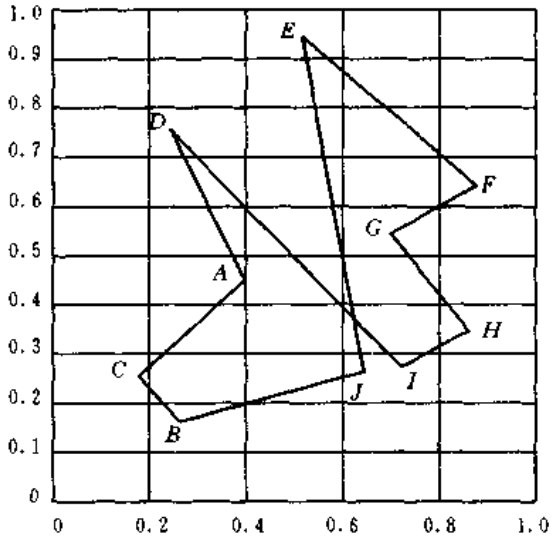
图 7.3　利用软限幅和 Sigmoid 非线性求解霍普费尔德 10 城市问题的旅程个数直方图



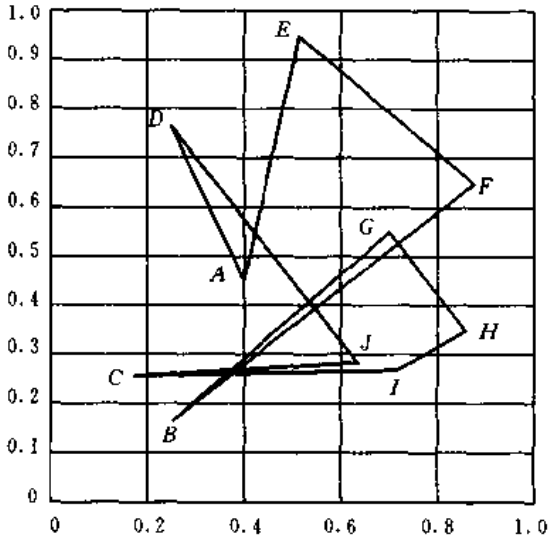
(a) 次优旅程



(b) 排第三位的旅程



(c) 利用软限幅所得的最差旅程



(d) 对应于$\beta=0.02$的最差旅程

图 7.4 各种有效旅程

表 7.2 利用 $\beta=0.3$ 的 Sigmoid 非线性求解 10 城市问题的统计

99.9%有效解答	旅程长度	出现次数	最小次数	最大次数	平均次数	标准偏差	旅　　程
最优	2.691	555	289	896	405	83	*BCADEFGHIJ*
次优	2.752	134	339	683	492	103	*BCADEGFHIJ*
第三	2.769	38	340	605	436	64	*BADEFGHIJC*
最差	3.697	1	2026	2026	2026	0	*AJGFHEDICB*

表 7.3　利用 $\beta=0.02$ 的 Sigmoid 非线性求解 10 城市问题的统计

58.2%有效解答	旅程长度	出现次数	最小次数	最大次数	平均次数	标准偏差	旅　　程
最优	2.691	19	7	21	13	4	*BCADEFGHIJ*
次优	2.769	1	20	20	20	0	*BADEFGHIJC*
第三	2.778	225	6	25	25	3	*ABCDEFGHIJ*
最差	4.737	1	11	11	11	0	*AEFBGHICJD*

7.2　应用启发式搜索算法求解 TSP

启发式搜索算法可应用于求解 TSP。我们可以任选有关城市(例如 A)作为搜索最优旅程的起始节点(见图 7.5)。在扩展这一起始节点时,它的子节点包括了所有从 A 到与其相通城市的路径。继续这一扩展过程,最终会在搜索树的底部得到所有可能的 TSP 旅程。

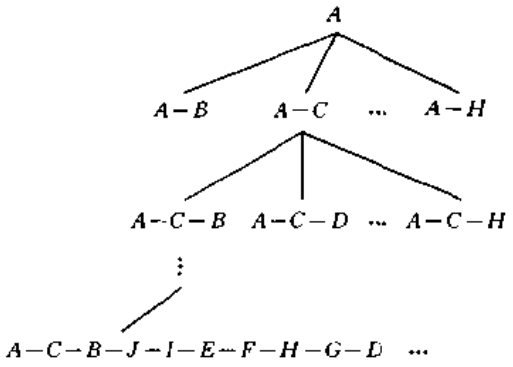


图 7.5　TSP 和启发式搜索

为了减小搜索空间并应用 A^* 算法,必须引入如第 2 章所介绍的代价函数 $f=g+h$。函数 g 的值可以是到目前为止已经构造出来的旅程的代价。用于估计后续距离的启发函数 h 则可用尚未访问的城市的最小生成树来计算。

设有一个连线长度由 $length(e_{ij})$ 给出的图 $G=(V,E)$,其中 $e_{ij}\in E$。G 的最小生成树为 $T=(V,F)$,$F\subseteq E$,其中 F 将 V 中所有节点连接起来并使边长总和最小。例如,考虑图 7.6 所示图 G,$T1$,$T2$ 和 $T3$ 是 G 的生成树,其中 $T2$ 是最小生成树。

图 $G=(V,E)$ 的最小生成树可由时间复杂度为 $O(|E|\log|V|)$ 的算法[125]找到,其中 $|V|$ 是 V 中节点的个数,$|E|$ 是 E 中连线的个数。让我们以习题 3.7 中的 10 城市问题为例描述搜索的过程。假定在数步节点展开之后,下一个进行节点展开的子旅程是 $ACBJI$。节点的展开是通过将与城市 I 相连的城市加到当前子旅程上实现的。这一展开将生成 5 个节点,分别对应于子旅程 $ACBJID$,$ACBJIE$,$ACBJIF$,$ACBJIG$ 和 $ACBJIH$。

这里计算具有子旅程 $ACBJIE$ 的节点 E 的 f 值。g 的值可由对这一子旅程上的城市间距离求和而得,即

$$g(ACBJE) = 0.3140 + 0.1107 + 0.3934 + 0.0498 + 0.0742 = 1.5721$$

为求解 $h(ACBJIE)$ 的值,我们需要利用最小生成树来估计获得完整旅程所需的距离。尚未访问到的城市只有 D,F,G,H,另外加上把这些城市连接到子旅程上的端点城市 A 和 E。因此,

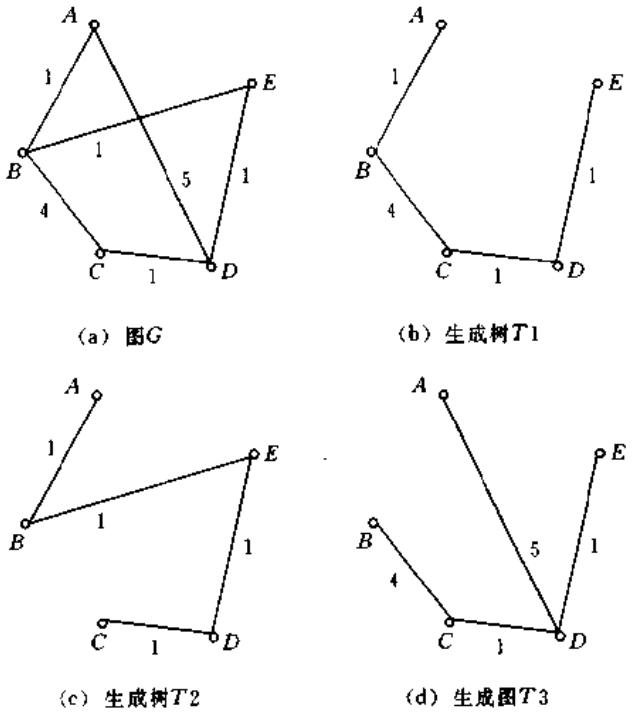


图 7.6 图的生成树和最小生成树

$h(ACBJIE)=\{A,E,D,F,G,H\}$ 的最小生成树的连线长度之和

上述最小生成树由连线 A-G,F-G,G-H,A-D 和 D-E 构成。由此可得 $h(ACBJIE)=1.4531$ 和 $f(ACBJIE)=3.0253$。

虽然子旅程和最小生成树并非一定构成一个有效旅程,它的总距离显然小于实际有效旅程。这就是说,h 是 h^* 的下界,因而这一启发函数是可行的,而且在算法终止时最优旅程一定可找到(表 7.4)。

表 7.4 20 城市 TSP 的坐标

城市	x 坐标	y 坐标	城市	x 坐标	y 坐标
A	5.294	1.558	K	4.399	1.194
B	4.286	3.622	L	4.660	2.949
C	4.719	2.774	M	1.232	6.440
D	4.185	2.230	N	5.036	0.244
E	0.915	3.821	O	2.710	3.140
F	4.771	6.041	P	1.072	3.454
G	1.524	2.871	Q	5.855	6.203
H	3.447	2.111	R	0.194	1.862
I	3.718	3.665	S	1.762	2.693
J	2.649	2.556	T	2.682	6.097

对于 10 城市问题，在 116 步扩展后得到最优旅程 *BCADEFGHIJ*。同样可求解具有表 7.4 给出的城市分布的 20 城市问题。这一 20 城市问题的最优旅程 *ACLBIQFTMEPRGSOJHDKN* 在展开 17222 个节点后被找到。

7.3 应用模拟退火算法求解 TSP

正如 4.2.2 节中所讨论过的，模拟退火可用于求解 *TSP*。图 4.3 给出了求解的步骤。生成一个旅程近邻的是二交换产生机制。它任选两个城市并通过反转在其被访城市中的次序而得到一个新的旅程。详见(4.25)～(4.26)式和图 4.2。

图 4.3 所示步骤的实现遵守如下规定：

(1) 初始温度 T_1 是这样得到的：从 1 出发并以 $T_1^+=1.05\times T_1^-$ 增大直至接受概率大于或等于 0.9。

(2) 第 k 个马尔科夫链的允许长度为 $L_k=10\times$ 城市个数。

(3) 第 k 个马尔科夫链的温度为 $T_{k+1}=0.95T_k$。

(4) 停止准则为①温度低于 0.01，或②没有新旅程生成。

表 7.5 总结了利用模拟退火求解 10 城市问题的模拟运行结果。表中，出现次数指每 1000 次模拟中出现的次数，平均次数指获得解答所用的平均扫描次数，而旅程自然是指访问城市的顺序。在从随机初始旅程出发的 1000 次运行中，模拟退火找到最优旅程的情况占 90%，且每次运行中被接受的转移平均为 3952 次。

表 7.5 利用模拟退火求解 10 城市问题的统计

90%有效解答	旅程长度	出现次数	平均次数	旅程
最优	2.691	906	3952	*BCADEFGHIJ*
次优	2.752	46	4056	*BCADEGFHIJ*
第三	2.769	10	4053	*DEFGHIJCBA*
最差	2.898	5	4497	*ABCDEFHIJG*

表 7.6 总结了利用模拟退火求解 20 城市问题(表 7.4)的模拟运行结果。在从随机初始旅程出发的 1000 次运行中，模拟退火找到最优旅程的情况占 79.2%且每次运行中被接受的转移平均为 8740 次。

表 7.6 利用模拟退火求解 20 城市问题的统计

79.2%有效解答	旅程长度	出现次数	平均次数	旅程
最优	24.38	792	8740	*ACLBIQFTMEPRGSOJHDKN*
次优	24.62	167	8638	*ADCLBIQFTMEPRGSOJHKN*
第三	25.17	39	9902	*ANKDHIOJSGRPEMTFQBLC*
最差	25.50	1	5794	*AQFTMEPRGSJOIBLCDHKN*

7.4 应用遗传算法求解 TSP

用遗传算法求解 TSP，必须具备：

(1) 旅程的符号串表示。

(2) 衡量旅程适存度的适度函数。

(3) 生成新旅程的遗传操作。

正如 6.4 节所讨论过的，一个 N 城市问题可被描述为一个被访问的城市序列并用这些城市的排列来表达。这一排列可用作遗传算法的符号串表示。

一个旅程的适度值 Ξ 定义为

$$fitness(\Xi) = C_{max} - f(\Xi) \tag{7.29}$$

其中 C_{max} 是整个群体里面所有符号串中的最大旅程距离，f 是由(4.24)式定义的旅程距离总和。

基于偏置轮盘概念的繁殖操作(见 6.2.1 节)被用来强化适者生存的规则。

用于生成新符号串的遗传操作是有序杂交(OX)操作[50]。OX 操作能保留排列并融合不同排列的有序结构单元。例如，假定旅程 1 和旅程 2 是 10 城市问题中的两个旅程：

旅程 1=(*J H D E F I G C B A*)

旅程 2=(*H G E B C J I A D F*)

由 *OX* 操作随机选择交叉截点并用“‖”表示，则

旅程 1=(*J H D* ‖ *E F I* ‖ *G C B A*)

旅程 2=(*H G E* ‖ *B C J* ‖ *I A D F*)

用“*”把旅程 1 的中段映射到旅程 2，反过来也一样，可得

旅程 1′=(* *H D* ‖ *E F I* ‖ *G* * * *A*)

旅程 2′=(*H G* * ‖ *B C J* ‖ * *A D* *)

将“*”从右向左滑动直到它们都在中段，由此可得

旅程 1′=(*E F I* ‖ * * * ‖ *G A H D*)

旅程 2′=(*B C J* ‖ * * * ‖ *A D H G*)

用另一个旅程的中段取代“*”串，即可得到后代

旅程 1′=(*E F I B C J G A H D*)

旅程 2′=(*B C J E F I A D H G*)

从上例可以看出，旅程 1′包含其双亲的一部分(旅程 1 的 *EFI* 和旅程 2 的 *BCJ*)以及其余城市的随机排列。一般来说，OX 操作倾向于保持城市的相对位置。

应用繁殖操作和 OX 操作，从随机选取的起始群体出发，10 城市和 20 城市(表 7.4)问题分别模拟运行 500 次。

在 10 城市问题的模拟运行中，用到了下列参数：

(1) 群体大小=100。

(2) 交叉概率=0.45。

(3) 迭代次数=100。

最优解答 *BCADEFGHIJ* 在所有 500 次运行中均被找到。生成的旅程总数为 10000。

在 20 城市问题的模拟运行中，用到了下列参数：

(1) 群体大小＝300。

(2) 交叉概率＝0.45。

(3) 迭代次数＝300。

最优解答 *ACLBIQFTMEPRGSOJHDKN* 在所有 500 次运行中均被找到。生成的旅程总数为 90000。

7.5 本征值分析概述

给定一个 $N\times N$ 的实数或复数矩阵 $\boldsymbol{W}$，一个实数或复数 λ 被称为 $\boldsymbol{W}$ 的一个本征值，当且仅当

$$\boldsymbol{W}\boldsymbol{x} = \lambda\boldsymbol{x} \tag{7.30}$$

其中 $\boldsymbol{x}$ 是对应于 λ 的本征向量。因此，λ 是 $\boldsymbol{W}$ 的一个本征值，当且仅当$(\boldsymbol{W}-\lambda\boldsymbol{I})$为奇异矩阵(在本节中 $\boldsymbol{I}$ 为单位矩阵)，亦即

$$\det(\boldsymbol{W} - \lambda\boldsymbol{I}) = 0 \tag{7.31}$$

其中 det(·)是矩阵的行列式。(7.31)式被称为 $\boldsymbol{W}$ 的本征方程。它是变量 λ 的 N 阶多项式，因而也被称为以 λ 为变量的 $\boldsymbol{W}$ 的本征多项式。可以证明本征多项式中 λ^N 的系数为 $(-1)^N$，λ^{N-1}的系数为$(-1)^{N-1}\mathrm{tr}(\boldsymbol{W})$，且常数项为 $\det(\boldsymbol{W})$，其中 $\mathrm{tr}(\boldsymbol{W})$为 $\boldsymbol{W}$ 的迹。

这里我们不深入到数学细节中去，而是用一个简单的例子来复习本征值分析的概念。考虑如下 6×6 矩阵 $\boldsymbol{W}$：

$$\boldsymbol{W} = \begin{bmatrix} 5 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 5 & 2 & 0 \\ 0 & 0 & 0 & 0 & 5 & 0 \\ 0 & 0 & 0 & 0 & 0 & 4 \end{bmatrix} \tag{7.32}$$

$\boldsymbol{W}$ 的本征值和本征向量的推导如下：

(1) 本征值

令本征方程 $\det(\boldsymbol{W}-\lambda\boldsymbol{I})=0$，得

$$\det(\boldsymbol{W} - \lambda\boldsymbol{I}) = 0 \Rightarrow (1-\lambda)^2(4-\lambda)(5-\lambda)^3 = 0 \tag{7.33}$$

本征多项式的根就是本征值。从(7.33)式可知，共有三个本征值：$\lambda_1=1$，$\lambda_2=4$ 和 $\lambda_3=5$。注意(7.33)式的各项指数之和与矩阵 $\boldsymbol{W}$ 的行或列数相等，亦即 2+1+3=6。每个不同本征值所对应的指数被称为该本征值的代数重数。λ_1，λ_2 和 λ_3 的代数重数分别为 2，1 和 3。λ_2 被称为简单本征值，亦即代数重数为 1 的本征值。

(2) 本征向量

① $\lambda_1=1 \Rightarrow (\boldsymbol{W}-\boldsymbol{I})\boldsymbol{x}=0$，即

$$\begin{bmatrix}4&0&0&0&0&0\\0&0&0&0&0&0\\0&0&0&0&0&0\\0&0&0&4&2&0\\0&0&0&0&4&0\\0&0&0&0&0&3\end{bmatrix}\begin{bmatrix}x_1\\x_2\\x_3\\x_4\\x_5\\x_6\end{bmatrix}=0 \tag{7.34}$$

$\Rightarrow x_1=x_4=x_5=x_6=0$，且 x_2 和 x_3 可为任意值。由此可得，

$$\boldsymbol{x}=\alpha[0\quad 1\quad 0\quad 0\quad 0\quad 0]^{\mathrm{T}}+\beta[0\quad 0\quad 1\quad 0\quad 0\quad 0]^{\mathrm{T}}, \tag{7.35}$$

与这一本征值相应的是两个线性独立的本征向量，分别为

$$[0\quad 1\quad 0\quad 0\quad 0\quad 0]^{\mathrm{T}}\text{ 和}[0\quad 0\quad 1\quad 0\quad 0\quad 0]^{\mathrm{T}}$$

它们构成了一个本征空间。因此，λ_1 的几何重数为 2，而且本征空间的维数为 2。

② $\lambda_2=4\Rightarrow(\boldsymbol{W}-4\boldsymbol{I})\boldsymbol{x}=0$，即

$$\begin{bmatrix}1&0&0&0&0&0\\0&-3&0&0&0&0\\0&0&-3&0&0&0\\0&0&0&1&2&0\\0&0&0&0&1&0\\0&0&0&0&0&0\end{bmatrix}\begin{bmatrix}x_1\\x_2\\x_3\\x_4\\x_5\\x_6\end{bmatrix}=0 \tag{7.36}$$

$\Rightarrow\boldsymbol{x}=\gamma[0\quad 0\quad 0\quad 0\quad 0\quad 1]^{\mathrm{T}}$ 是本征向量，γ 为任意值且 λ_2 的几何重数为 1。

③ $\lambda_3=5\Rightarrow(\boldsymbol{W}-5\boldsymbol{I})\boldsymbol{x}=0$，即

$$\begin{bmatrix}0&0&0&0&0&0\\0&-4&0&0&0&0\\0&0&-4&0&0&0\\0&0&0&0&2&0\\0&0&0&0&0&0\\0&0&0&0&0&-1\end{bmatrix}\begin{bmatrix}x_1\\x_2\\x_3\\x_4\\x_5\\x_6\end{bmatrix}=0 \tag{7.37}$$

$\Rightarrow\boldsymbol{x}=\zeta[1\quad 0\quad 0\quad 0\quad 0\quad 0]^{\mathrm{T}}+\xi[0\quad 0\quad 0\quad 1\quad 0\quad 0]^{\mathrm{T}}$ 是本征空间，ζ,ξ 为任意值。因而 λ_2 的几何重数为 2，且对应的本征空间维数是 2。

正如上述例子所示的，本征值的代数重数与其几何重数是截然不同的。

7.6 连接矩阵的 λ_1 的推导

在 7.1.1 节中，连接矩阵 $\boldsymbol{W}$ 的本征值 λ_1 是由证明下式决定的：

$$\boldsymbol{We}^1=\lambda_1\boldsymbol{e}^1 \tag{7.38}$$

注意 $\boldsymbol{We}^1$ 依(7.5)式是一个列向量，且其每个元素均等于 λ_1。因此，由(7.9)式可得 $\boldsymbol{We}^1$ 的每个元素为

$$\lambda_1=\sum_Y\sum_j w_{X_i,Y_j}$$

$$
\begin{aligned}
&= -A\sum_{X}\sum_{j}\delta_{XY}(1-\delta_{ij}) - A\sum_{Y}\sum_{j}\delta_{ij}(1-\delta_{XY}) - CN^2 \\
&= -A\sum_{j}(1-\delta_{ij}) - A\sum_{Y}(1-\delta_{XY}) - CN^2 \qquad (7.39) \\
&= -2A(N-1) - CN^2
\end{aligned}
$$

7.7 习题

7.1 说明为什么霍普费尔德和坦客[86]在求解 10 城市问题时选用了 $\tilde{N}=15$(提示：用(3.22)式作为能量函数推导一个与(7.23)式类似的公式)。

7.2 推导对应于有效子空间的 W 的本征值((7.9)式)。

7.3 推导对应于无效子空间的 W 的本征值((7.9)式)。

7.4 考察并说明(7.8)式第 4 项 D 是如何影响连接矩阵的本征值分析的。推导出 D 的适当表达式(提示：见[138])。

7.5 证明当 $\lambda_1=-CN^2$ 时解答 S 被限制在刚好有 N 个神经元被激活的子空间中。

7.6 令 $\lambda_i, i=1,2,\cdots,N$ 为一个 $N\times N$ 矩阵 W 的本征值。W 的每个元素用 w_{ij} 表示。给定(1) $a_{ij}=w_{ij}, \forall\ i\neq j; a_{ii}=w_{ii}-\alpha, \forall\ i$ 和(2) $a_{ii}=w_{ii}-\alpha, \forall\ i,j$。找出一个每个元素用 a_{ij} 表示的矩阵 A 的本征值。

7.7 假定 $|\lambda_1|\approx 160|\lambda_2|, |\lambda_3|\approx 160|\lambda_2|$ 和 $D=\dfrac{|\lambda_3|}{80}$，检验(7.25)式～(7.27)式。

7.8 求出 N 城市问题的互不相同旅程的闭环旅程个数。

7.9 比较将 PMX,OX 和 CX 操作应用于 7.2 节讨论的 20 城市问题的性能。

第8章 电 信

为了提供现有的和正在形成的电信服务，要用到某些形式的计算智能。虽然传统的方法已经得到了非常成功的应用，然而对更快捷、更可靠的传输的不断增长的需要呼唤着新方法的出现。计算智能在电信中的应用包括异步传输方式(ATM)宽带网络的交通管制、网络管理、信道均衡、数据和视频压缩、参数估计、路由选择、卫星通信、波传播建模以及无线通信。本章将向读者介绍一些实际应用的例子，其中有卫星广播调度[19]和集成通信系统中的数据传输率最大化问题[166]。更多的应用可参见文献[1,14,15,28,38,40,41,42,47,69,74,78,79,111,113,114,121,126,147,151,160,165,167,175]。

8.1 卫星广播调度

优化一组卫星对一组地面站的广播时间，即卫星广播调度问题(SBS)，是低空卫星通信系统中必须解决的重要问题。波莱特(Bourret)等人在文献[34,35]中用一个三层相连神经元模型来求解这一问题，其顺序搜索是由一个基于卫星的动态优先权竞争激活机制来控制的。在局域进行的顺序搜索也十分耗时。此外还需要两个附加的先决条件：一组互不相同的卫星优先权和一组在解决大型问题时很难确定的适当要求。

在这一节中，介绍另外一种基于均场退火的更加有效的解决SBS问题的方法。这个方法不需要前述的两个附加条件。它没有采取文献[132]介绍的特殊神经元模型(分级神经元)以缩小解答空间并避免破坏性的冗余度，而是使用了由一个结合矩阵箝位的通用神经元模型。这一常用于学习算法[13][116][128]的箝位技术可以大大减小解答空间。由于Sigmoid函数的非线性，存在一个所谓的临界温度 T_c。可推导估算 T_c 的公式以便取代决定 T_c 的试错法。

8.1.1 SBS问题的神经网络表达

自从第一颗商用卫星“远星”在60年代被成功发射以来，卫星通信工业的年产值已增长为数十亿美元。尽管还不能覆盖远北纬[154]，发射成本高，且需要很大天线等，但多数商用系统都已被发射到对地球静止的轨道上。对地球静止的轨道的优点是卫星与地面站呈静态，因而不需要从一个卫星到另一个卫星的“交接”。选用对地球静止的轨道主要是为了适应国际长途电话的大通信量。当各种电信应用变得越来越复杂时，有时需要用不是对地球静止的轨道。在这种情况下，就需要对从一个卫星到另一个卫星的“交接”[134]加以调度。这正是本章所要介绍的内容。因为低空卫星的发射数目很小，而且多数是用于监视和数据采集的专有或保密卫星，只有很少人涉及这一问题。不过，低空卫星具有卫星能耗低、天线减小、传播延迟变短以及图象分辨率高等潜在优点，这正促使工业界建立一个这样的卫星

系统。因此卫星广播调度成为一个重要问题。对于当前已经和正在开发的用于个人通信系统的现代化卫星，文献[6]给出了较为详尽的综述。

求解 SBS 问题的目的是将每个卫星的广播时间最大化，并使如下约束条件得到满足：

(1) 一个卫星不能同时向多于一个地面站广播；

(2) 一个地面站不能同时从多于一个卫星接收信息；

(3) 一个卫星的播出必须离要求时间尽可能接近，而且系统不能分配比要求时间还长的时间，除非其它卫星的要求时间都得到完全满足；

(4) 卫星只当它能被地面站接收时才播出。

下列术语用于表述 SBS 问题：

(1) $s=\{a,b,c,d,\cdots\}=\{1,2,3,\cdots,i,\cdots,N_s\}$是由 N_s 个元素(卫星)组成的卫星集合，其中 $a,b,c,d,\cdots$表示不同的卫星，每一个都用在 1 和 N_s 之间的整数 i 编号。

(2) $a=\{z,y,x,w,\cdots\}=\{1,2,3,\cdots,j,\cdots,N_a\}$是由 N_a 个元素(地面站)组成的地面站集合，其中 $z,y,x,w,\cdots$表示不同的地面站，每一个都用在 1 和 N_a 之间的整数 j 编号。

(3) $t=\{1,2,3,\cdots,k,\cdots,N_t\}$是由 N_t 个元素(时间段)组成的时间段集合。

(4) $r=[r_1,r_2,\cdots,r_{N_s}]^T$ 是一个向量，表示一个给定问题所要求的时间段数集合。它包括 N_s 个元素(时间段)。这里，$r_1,r_2,\cdots,r_{N_s}$分别为卫星 $1,2,3,\cdots,N_s$ 的时间段个数。

(5) $u=[u_1,u_2,\cdots,u_{N_s}]^T$ 是一个向量，表示一个系统对每个卫星所分配的时间段数集合。它包括 N_s 个元素(时间段)。这里，$u_1,u_2,\cdots,u_{N_s}$分别是分配给卫星 $1,2,3,\cdots,N_s$ 的时间段个数。

求解的目标是给每一个卫星都尽可能地分配所需时间段，以同时满足下列两个准则：

(1) 调度方案必须合理，亦即满足所有约束；

(2) 向量 u 和 r 之间的距离必须被最小化。

1. 神经元编码

用 s_{ijk}表示神经元。s_{ijk}的“激活”或“抑制”是由卫星 i 在 k 时间段是否被分配给地面站 j 来决定的。因此，s_{ijk}的数学定义为

$$s_{ijk}=\begin{cases}1 & \text{如果卫星 } i \text{ 在 } k \text{ 时间段被分配给地面站 } j\\ 0 & \text{其它}\end{cases} \tag{8.1}$$

2. 结合矩阵

从上面定义的神经元可以清楚地看到，基于前述约束条件(4)，一些神经元的值恒定为零。这是因为即使所有地面站都不工作，也没有一个地面站对该卫星是可见的。通常，因约束条件(4)而取值为零的神经元数目很大。通过在整个优化过程中将不满足约束条件(4)的神经元箝制为零，这一约束条件可以被吸纳到神经元网络中去。下面 N_sN_a 行 N_t 列的结合矩阵 A 正是为此而定义的。

$$A=\begin{bmatrix}A_1\\A_2\\\vdots\\A_{N_s}\end{bmatrix}=\begin{bmatrix}a_{111} & a_{112} & \cdots & a_{11N_t}\\ \vdots & \vdots & \vdots & \vdots\\ a_{1N_a1} & a_{1N_a2} & \cdots & a_{1N_aN_t}\\ \vdots & \vdots & \vdots & \vdots\\ a_{N_sN_a1} & a_{N_sN_a2} & \cdots & a_{N_sN_aN_t}\end{bmatrix} \tag{8.2}$$

其中 A_i 是一个 $N_a \times N_t$ 的子矩阵。它与加在卫星 i 上的约束关联，且有

$$s_{ijk} = \begin{cases} 1 & \text{如果卫星 } i \text{ 在 } k \text{ 时间段可被地面站 } j \text{ 接收} \\ 0 & \text{其它} \end{cases} \tag{8.3}$$

从 A 的定义和问题的约束条件出发，可以得到两个重要的关系：

(1) 卫星 i 所要求的最大时间段个数 $r_{\max}(i)$ 必须小于或等于子矩阵 A_i 中的非零列数，亦即

$$r_{\max}(i) \leqslant \sum_{k=1}^{N_t} (A_i \text{ 中的非零列}) \tag{8.4}$$

(2) 可用的时间段 u_i 必须小于或等于 $r_{\max}(i)$，亦即

$$u_i = \sum_{j}^{N_a} \sum_{k}^{N_t} s_{ijk} \leqslant r_{\max}(i) \tag{8.5}$$

上述关系会在检验解答是否合理时用到。

3. 能量函数

将一个有约束的最优化问题转化为无约束的最优化问题而得到的 SBS 问题的能量函数由一个代价项和一些反映约束的惩罚项构成，即

$$E = w_0 E_0 + w_1 E_1 + w_2 E_2 + w_3 E_3 \tag{8.6}$$

其中拉格朗日参数 w_0, w_1, w_2, w_3 分别为 E_0, E_1, E_2, E_3 项的权重。作为代价项的第一项为

$$E_0 = -\frac{1}{2} \sum_{i}^{N_s} \sum_{j}^{N_a} \sum_{k}^{N_t} (s_{ijk} \cdot s_{ijk}) \tag{8.7}$$

这一项的目的是使总播出时间最大，其中的负号表示要将 E_0 项最小化。根据 4 个约束定义以下的惩罚项：

(1) 一个卫星不能同时向多于一个地面站广播。可以证明这一约束条件的惩罚项为

$$E_1 = \sum_{i}^{N_s} \sum_{k}^{N_t} \sum_{j}^{N_a} \sum_{j_1 \neq j}^{N_a} (s_{ijk} \cdot s_{ij_1k}) \tag{8.8}$$

(2) 一个地面站不能同时从多于一个卫星接收信息。约束(2)与约束(1)对偶。它的惩罚项类似地定义为

$$E_2 = \sum_{j}^{N_a} \sum_{k}^{N_t} \sum_{i}^{N_s} \sum_{i_1 \neq i}^{N_s} (s_{ijk} \cdot s_{i_1jk}) \tag{8.9}$$

(3) 一个卫星的播出必须离要求时间段尽可能接近，而且系统不能分配比要求时间还长的时间，除非其它卫星的要求时间都得到完全满足。上面所述的前一部分表明 u 与 r 之间的距离应为最小。因而，与之对应的惩罚项为

$$E_3 = \sum_{i}^{N_s} \left(\sum_{j}^{N_a} \sum_{k}^{N_t} s_{ijk} - r_i \right)^2 = \sum_{i}^{N_s} (u_i - r_i)^2 \tag{8.10}$$

(4) 一个卫星只有当它能被一个地面站可见时才能播出。这一约束由结合矩阵将神经元的取值箝制为零来实现。

8.1.2 SBS 问题的均场公式

对于二值神经元变量 $s_{ijk} \in \{0,1\}$，可得与(5.14)式类似的 SBS 问题的均场公式，即

$$v_{ijk}=\frac{1}{2}+\frac{1}{2}\tanh\left(-\frac{1}{2T}\frac{\partial E}{\partial v_{ijk}}\right) \tag{8.11}$$

其中 v_{ijk} 是与神经元 s_{ijk} 对应的均场变量。把上述将神经元的取值箝制为零的技术引入(8.11)式，SBS 问题的均场公式成为

$$v_{ijk}=a_{ijk}\left(\frac{1}{2}+\frac{1}{2}\tanh\left(-\frac{1}{2T}\frac{\partial E}{\partial v_{ijk}}\right)\right) \tag{8.12}$$

这一等式事实上已将前一小节所述约束(4)包含在内。

8.1.3 算法的参数

需要规定的参数包括拉格朗日参数(w_0,w_1,w_2,w_3)、临界温度、饱和温度和退火流程。本节主要介绍拉格朗日参数。

一般来讲，好的解答可从 w_0,w_1,w_2,w_3 空间的一个合理的较大范围内获得。然而，为了确保所选的参数落在这一范围之内，一些指导性原则还是需要的。在均场域中，用 v_{ijk} 简单地取代(8.7)式～(8.10)式中的 s_{ijk}，能量函数 E_0,E_1,E_2,E_3 成为均场变量的函数：

$$E_0=-\frac{1}{2}\sum_{i}^{N_s}\sum_{j}^{N_a}\sum_{k}^{N_t}(v_{ijk}\cdot v_{ijk}) \tag{8.13}$$

$$E_1=\sum_{i}^{N_s}\sum_{k}^{N_t}\sum_{j}^{N_a}\sum_{j_1\neq j}^{N_a}(v_{ijk}\cdot v_{ij_1k}) \tag{8.14}$$

$$E_2=\sum_{j}^{N_a}\sum_{k}^{N_t}\sum_{i}^{N_s}\sum_{i_1\neq i}^{N_s}(v_{ijk}\cdot v_{i_1jk}) \tag{8.15}$$

$$E_3=\sum_{i}^{N_s}\left(\sum_{j}^{N_a}\sum_{k}^{N_t}v_{ijk}-r_i\right)^2 \tag{8.16}$$

均场域中总能量函数的微分为

$$\frac{\partial E}{\partial v_{ijk}}=w_0\frac{\partial E_0}{\partial v_{ijk}}+w_1\frac{\partial E_1}{\partial v_{ijk}}+w_2\frac{\partial E_2}{\partial v_{ijk}}+w_3\frac{\partial E_3}{\partial v_{ijk}} \tag{8.17}$$

参数 w_0 用于平衡代价项与约束条件项。参数 w_1,w_2,w_3 则反映了约束(1)到(3)之间的相对重要性。因为 $\frac{\partial E_1}{\partial v_{ijk}}$ 与 $\frac{\partial E_2}{\partial v_{ijk}}$ 性质相近且比其它项重要得多，它们被赋予相对较大的相同权值。例如，可选 $w_0=0.4,w_1=2.0,w_2=2.0$。

在考虑每一个单独参数对任意均场变量的影响时，从(8.14)式和(8.15)式可知 $\frac{\partial E_1}{\partial v_{ijk}}$ 与 $\frac{\partial E_2}{\partial v_{ijk}}$ 总为正值，因而根据(8.12)式，它们使 v_{ijk} 的值趋向于“0”。从(8.13)式可见 $\frac{\partial E_0}{\partial v_{ijk}}$ 总为负数并使 v_{ijk} 的值趋向于“1”。基于所需时间段是否得到满足，$\frac{\partial E_3}{\partial v_{ijk}}$ 既可为正亦可为负，从而使神经元的值相应地趋向于“0”或“1”。

假定约束条件(1)和(2)已被满足，亦即 $E_1=E_2=0\Rightarrow\frac{\partial E_1}{\partial v_{ijk}}=\frac{\partial E_2}{\partial v_{ijk}}=0$。下面要做的就是决定 w_0 和 w_3 之间的关系。在 v_{ijk} 为 0 或 1 的特殊情况下，对于每一个特定的 i，如果取值为 1 的均场变量个数大于所需时间段数(见(8.16)式)，这表明系统已经分配了比需要还

多的时间段。因此，系统应在此时试图“抑制(置0)”一个均场变量。注意，“激活”的均场变量满足$\frac{\partial E_0}{\partial v_{ijk}}=-1$(见(8.13)式)与$\frac{\partial E_3}{\partial v_{ijk}}=2\left(\sum_j^{N_a}\sum_k^{N_t}v_{ijk}-r_i\right)\geqslant 2$(见(8.16)式)，这是因为系统分配了比需要还多的时间段。也就是说，$\left(\sum_j^{N_a}\sum_k^{N_t}v_{ijk}-r_i\right)\geqslant 1$。要抑制一个均场变量，需有(见(8.17)式)

$$\begin{aligned}\frac{\partial E}{\partial v_{ijk}}>0 &\Rightarrow w_0\frac{\partial E_0}{\partial v_{ijk}}+w_3\frac{\partial E_3}{\partial v_{ijk}}>0\\ &\Rightarrow w_0(-1)+2w_3\left(\sum_j^{N_a}\sum_k^{N_t}v_{ijk}-r_i\right)>0\\ &\Rightarrow w_0(-1)+2w_3>0\\ &\Rightarrow w_3>0.5w_0\end{aligned}\tag{8.18}$$

在另一个特殊情况下，网络分配的时间段个数小于所需时间段数。这时网络应试图激活一个均场变量。注意，“抑制”的均场变量有$\frac{\partial E_0}{\partial v_{ijk}}=0$且$\frac{\partial E_3}{\partial v_{ijk}}<0$。因此，只要$w_3>0$，$\frac{\partial E}{\partial v_{ijk}}<0$，均场变量被“激活”。

总之，对于SBS问题，给出以下经验值：

$$w_0=0.4,\ w_1=w_2=2,\ w_1>w_3>0.5w_0\tag{8.19}$$

8.1.4 临界温度T_c

为了深入了解求解过程的动态特性，均场等式可被改写如下：

$$v_{ijk}-\frac{1}{2}a_{ijk}=\frac{1}{2}\tanh\left(-\frac{1}{2T}\frac{\partial E}{\partial v_{ijk}}a_{ijk}\right)\tag{8.20}$$

在SBS问题中能量函数对于均场变量的偏导具有下列线性形式：

$$\frac{\partial(E)}{\partial v_{ijk}}=m(v_{ijk}-\alpha)\tag{8.21}$$

其中m和α是常数。从(8.20)式可以很容易看出，除了被结合矩阵强制为零的神经元之外，所有其它神经元在高温下的取值都为0.5。不过，随着温度的下降，非线性$\tanh(\cdot)$渐进于一个Signum函数，且均场变量开始稳定在“0”和“1”处。这里要确定的参数是具有显著状态转移的临界温度，这时系统能量急剧下降。由于未被强行箝制为零的神经元的起始值都为0.5，从而上述显著状态转移发生在神经元开始竞争取值为1或0时，因此可以给出临界温度的定义。

定义8.1 临界温度是至少一个均场变量v_{ijk}从起始状态值(亦即0.5)达到1或0的最高温度。

基于上述定义，临界温度可由下面的引理加以估计。

引理8.1 SBS问题的临界温度T_c可由下式近似：

$$T_c=\frac{m}{4\beta}(2\alpha-\beta-1)\tag{8.22}$$

其中m，α和β可从下列方程组得到：

$$\begin{cases} 2v_{ijk}-1=\beta \\ -\dfrac{1}{2T}m(v_{ijk}-\alpha)=\beta \end{cases} \tag{8.23}$$

证明：由定义 8.1，在至少一个均场变量 v_{ijk} 从起始状态值 0.5 达到 1 或 0 的条件下，求解(8.20)式中的参数 T 可得到临界温度。由于(8.20)式中有非线性项 tanh(·)，推导出它的严格数学解答即使可能也是非常困难的。下面将 $\tanh\left(-\frac{x}{2T}\right)$ 在 $x=0$ 处泰勒展开，对(8.20)式做近似处理，即

$$\tanh\left(-\frac{x}{2T}\right)=-\frac{x}{2T}+\frac{1}{24}\left(\frac{x}{2T}\right)^3+\cdots \tag{8.24}$$

取其第一项而得到

$$2v_{ijk}-1=-\frac{1}{2T}\frac{\partial E}{\partial v_{ijk}} \tag{8.25}$$

这里，没有考虑均场变量被强行置零的情况，即 $a_{ijk}=1$。将(8.21)式引入上述结果可得

$$2v_{ijk}-1=-\frac{m}{2T}(v_{ijk}-\alpha) \tag{8.26}$$

因此，求解方程组(8.23)可得临界温度。这里，因为显著状态转移发生在神经元取值从起始温度 0.5 达到 1 或 0 时，所以 β 为+1 或−1，亦即 $v_{ijk}=1$ 或 $0\Rightarrow\beta=\pm1$。进一步讲，如果 $\alpha>1$ 且 $m>0$，则稳态解[①] 为 $v_{ijk}=1(\beta=1)$；如果 $\alpha<1$ 且 $m>0$，则稳态解为 $v_{ijk}=0$ $(\beta=-1)$。从取值 0.5 的起始状态出发，下表列出了 m，α 和 β 之间的关系：

	$\alpha>0.5$	$\alpha<0.5$
$m>0$	$\beta=+1$	$\beta=-1$
$m<0$	$\beta=-1$	$\beta=+1$

严格地讲，当 $0<\alpha<1$ 时，v_{ijk} 不会取值 0 或 1。然而，在数次迭代后，这一特定 v_{ijk} 将收敛于 0 或 1。对于不同的均场变量 v_{ijk}，α 的取值可以是不同的。不过，只要临界温度的估计值比实际临界温度值高，退火理论保证系统可以收敛于一个(接近)全局最优。因此，对于一个给定的结合矩阵 A，每一个均场变量对应的 α 都应由(8.21)式计算出来，最大的 α 值将被选用。

退火流程 这里采用下列线性退火流程：

$$T(n+1)=0.9T(n) \tag{8.27}$$

其中 $T(0)=T_c$。退火过程的停止准则由网络饱和时的温度(饱和温度)定义。网络在满足下列条件时饱和：

(1) 所有神经元的值都在[0.0,0.2]或[0.8,1.0]范围内，无一例外。

(2) $\frac{1}{N}\sum_i\sum_j\sum_k(v_{ijk})^2>0.95$，其中 N 是取值在[0.8,1.0]范围内的均场变量个数。

---

① 由 $\frac{1}{2}+\frac{1}{2}\tanh\left(-\frac{m}{2T}(v_{ijk}-\alpha)\right)$ 估值的稳态解应该达到 0 或 1。

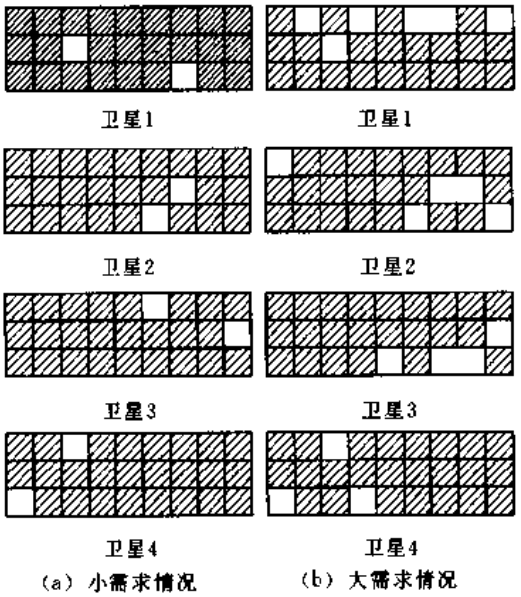
8.1.5 一个示例

上面提出的方法已被应用于解决多种不同需求量的SBS问题。需求播出时间小于系统可分配最大容量的情况被称为小需求型；而需求播出时间大于网络最大容量的情况被称为大需求型。这里以一个具有108个神经元(其中44个神经元未被强制置零)的网络示范求解小需求型和大需求型的例子。

设有4颗卫星、3个地面站和9个时间段，且此SBS问题的第4项约束条件由下面结合矩阵定义：

$$A=\begin{bmatrix}1&1&0&1&0&1&1&0&1\\0&0&1&0&0&0&0&1&0\\1&0&0&1&1&1&1&1&0\\1&1&1&1&0&0&0&0&1\\0&0&1&0&0&0&1&1&0\\0&0&0&1&0&1&0&0&1\\1&0&1&0&0&1&1&0&1\\0&0&0&0&0&0&0&0&1\\0&0&0&0&1&1&1&1&0\\0&1&1&0&0&0&0&0&1\\0&0&0&0&0&0&0&1&1\\1&0&0&1&0&0&0&1&1\end{bmatrix} \tag{8.28}$$

已知所需时间段为$r=[2,2,2,2]^T$。符合需求且满足约束条件的时间段分配如图8.1(a)所示。当所需时间段$r=[9,8,7,6]^T$明显超出系统容量时，解答如图8.1(b)所示，且时间段分配为$u=[6,5,4,3]^T$。虽然系统不能满足所有需求，但它仍在给定约束条件下提供了



(a) 小需求情况　　(b) 大需求情况

图 8.1　小需求情况和大需求情况的广播调度

每个方块代表被分配的时间；每行代表一个地面站；每列代表一个时间段。

最大播出时间。在更大运算代价下，对更大需求问题的求解给出了一致性的结果。建议读者用不同的例子进行实验。其它例如收敛性等问题的讨论可见文献[19]。

8.2 集成 TDMA 通信系统的数据吞吐量最大化

在集成分时多路存取(TDMA)通信系统中，语音和数据通过多路复用来分享一个共同的传输通道。时间轴被分为许多帧，每帧又由一定数目的定长时段组成。每一帧的时段都有特定部分用于语音，剩下的则用于数据。许多科技文献都将语音业务量用有损系统，而将数据量用排队系统作为模型。因此在信号到达时，如果不能获得有效时段，语音业务量就被阻断而不进行传输，而数据则会被排队，并在数据时段有空时按先来先送的规则传送。故此，系统设计的目标是将语音业务量被阻断的概率和数据量排队的延迟最小化。通常用到的是定长边界(FB)或可变边界(MB)两种方法之一。在定长边界法中，TDMA 的帧被划分为各有特定数目时段的两个区：一个给语音，另一个给数据。分配给语音的但不在工作的时段不能用来传输数据。显然，这种方法不能充分利用系统资源。相反，可变边界法利用剩余语音时段传送数据。这在很多时候可以减小排队延迟。为了充分利用集成系统的资源，已有许多多路复用策略被提出(文献[37,52,72,173])。

文献[37]中使用了一种数据传输的分段 ALOHA 随机存取协议。它不构成一个长队而是在出现冲突(当两个或更多数据包在同一时段被传送)时，数据被重传。最大数据吞吐量可以通过搜索语音与数据在帧中相对位置的最优构形来获得。这是一个 NP 完全约束下的最优化问题。作为又一个电信应用的示例，均场退火算法在数据吞吐量最大化中的应用体现了它在性能和计算复杂度之间很好的折衷。

8.2.1 多路复用方案与数据吞吐量

这里介绍的均场退火方法采用了与文献[37]中相同的假定和多路复用策略。在TDMA的传输通道中，语音以同步方式传输，数据则以争用方式传输。一个由 N 个定长时段组成的帧格式如图 8.2 所示。因为这里所用的是MB多路复用方式，数据传输既可使用数据时段也可利用即时可用的剩余语音时段。下面列出一些术语：

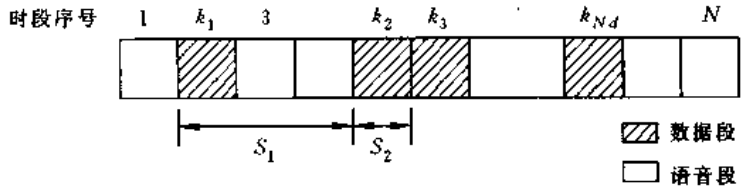


图 8.2 一个由 N 个时段组成的帧，其中 N_d 个分配给数据

N_d 是在一个给定帧中可被数据包所用的时段数；

k_i 是第 i 个可用数据时段的时段序号，且 $1 \leqslant k_i \leqslant N, 1 \leqslant i \leqslant N_d$；

s_i 是第 i 个数据时段与紧接其后的第$(i+1)$个数据时段的间距，且

$$s_i=\begin{cases}k_{i+1}-k_i & \text{如果 } i=1,2,\cdots,N_d-1\\ k_1+N-k_{N_d} & \text{如果 } i=N_d\end{cases} \tag{8.29}$$

$s=(s_1,s_2,\cdots,s_{N_d})$在集成通信系统中被称为帧模式。

从帧的格式可以很容易看出，

$$s_1+s_2+\cdots+s_{N_d}=N \tag{8.30}$$

为了使问题变得在数学上易解，引进下列假定：

(1) 语音传输的通话时间比帧长大得多，因而数据对一个给定帧模式的排队状态可达到稳态。

(2) 分段 ALOHA 随机存取协议[145]用作数据传输。总数据流通量(包括新的和重传)符合平均到达率为 G 包/时段的泊松(Poisson)过程。

在间距 s_i 上没有冲突的概率为在该时间段上没有泊松数据流通量被生成的概率。故此，

$$\Pr(\text{在间距 } s_i \text{ 上没有冲突})=\mathrm{e}^{-G\cdot s_i} \tag{8.31}$$

$$\Pr(\text{数据包在 } s_i \text{ 上成功传送})=G\cdot s_i\cdot \mathrm{e}^{-G\cdot s_i} \tag{8.32}$$

规一化的吞吐量为

$$\gamma_s=\frac{1}{N_d}\sum_{i=1}^{N_d}G\cdot s_i\cdot \mathrm{e}^{-G\cdot s_i} \tag{8.33}$$

数据吞吐量在很大程度上受语音和数据相对位置的影响。为了将数据吞吐量最大化，数据流通量被授予使用帧中 N_d 个时段的较高优先权。因此，给定 N 和 N_d，存在 $C_{N_d}^{N}$ 个帧模式。例如，假定在最大化数据吞吐量时有 $N=50$ 且 $N_d=15$，总共可以有的帧模式数为 $C_{15}^{50}=82.25\times10^{12}$。随着待解问题变大，利用穷举搜索在所有帧模式中寻找最优帧模式会变得难以处理。

8.2.2 数据吞吐量的最大化

数据吞吐量的最大化等价于找出一个特别的帧模式 s^{opt} 使得

$$\gamma_{s^{\mathrm{opt}}}=\max_{s\in S}\gamma_s=\max_{s\in S}\frac{1}{N_d}\sum_{i=1}^{N_d}G\cdot s_i\cdot \mathrm{e}^{-G\cdot s_i} \tag{8.34}$$

且有约束

$$\sum_{i=1}^{N_d}s_i=N \tag{8.35}$$

其中 $1\leqslant s_i\leqslant N-N_d+1$。

运用均场退火求解这一有约束条件的最优化问题的步骤可以归纳如下：

(1) 构造一个既能最大化数据吞吐量又能满足约束条件的能量函数。

(2) 选择能在数据吞吐量最大化和满足约束条件之间保持平衡的拉格朗日参数。

(3) 确定临界温度 T_c。

(4) 确定退火流程。

(5) 定义收敛准则。

(6) 进行迭代操作直至找到最优解：

① 将神经元 i 的均值初始化为 $v_i=0.5+0.001\times\text{rand}[-1,1],\forall i$，且从临界温度开始退火；

② 在每个温度 T 处，按照均场等式更新 $v_i,\forall i$，直到满足一定准则。在固定温度下对所有神经元的全部更新称为一次扫描。

③ 根据退火流程降温，重复步骤直到收敛准则被满足。在每个温度下，进行一系列扫描。

$v_i\forall i$ 的值在退火结束时对应于帧模式。

将这一最优化问题映射到均场退火的框架中去，上述步骤中所需参数的确定方式如下：

1. 能量函数

$$E(s)=-\frac{w_1}{N_d}\sum_{i=1}^{N_d}Gs_i\mathrm{e}^{-G\cdot s_i}+\frac{w_2}{2}\Big(\sum_{i=1}^{w_2}s_i-N\Big)^2+w_3\sum_{i=1}^{N_d}\sum_{j=1}^{N'}s_{ij}(1-s_{ij}) \tag{8.36}$$

其中 $s_i=\sum_{j=1}^{N'}s_{ij}2^{j-1}$，$w_l>0(l=1,2,3)$，$s_{ij}\in\{0,1\}$，

$$N'=\begin{cases}\lceil\log_2(N-N_d-1)\rceil+1 & \text{如果}\lceil\log_2(N-N_d-1)\rceil=\log_2(N-N_d-1)\\ \lceil\log_2(N-N_d-1)\rceil & \text{其它}\end{cases} \tag{8.37}$$

式中$\lceil\cdot\rceil$是取上整数操作。

第 i 个数据时段与紧接其后的第$(i+1)$个数据时段的间距由 s_i 表示。因为 s_i 是一个整数，且 $1\leqslant s_i\leqslant N-N_d+1$，$s_i$ 可用 N 个二值神经元 s_{ij}的各项表达。换句话说，s_{ij}对应于 s_i 的 N 位二值表示。第一项是权值为负的数据吞吐量，因而最大数据吞吐量对应于最小负吞吐量。第二项引进了对违反约束的惩罚。如果(8.35)式给出的约束被满足，则第二项对应的能量为零。第三项仅当所有神经元都收敛于 0 或 1 时为零。如果适当选取拉格朗日参数(w_1,w_2,w_3)，退火过程会将系统引向一个对应于最优帧模式的最小能量构形。

2. 拉格朗日参数

退火过程将系统松弛到具有最小能量的状态并同时满足所有的约束条件。一个违背约束的状态(帧模式)s'应该具有比满足约束的一个状态 s 的能量更大的能量，亦即 $E(s')>E(s)$。考虑状态 s 满足(8.35)式给出的约束条件，且每个神经元都已收敛到 0 或 1 的情况。如果 s 的邻近状态 s' 与其只有一个元素之差但却违背约束从而使得

$$\sum_{i=1}^{N_d}s'_i=N-1\neq N \tag{8.38}$$

则

$$E(s')=-\frac{w_1}{N_d}\sum_{i=1}^{N_d}G\cdot s'_i\cdot\mathrm{e}^{-G\cdot s'_i}+\frac{w_2}{2} \tag{8.39}$$

$$E(s)=-\frac{w_1}{N_d}\sum_{i=1}^{N_d}G\cdot s_i\cdot\mathrm{e}^{-G\cdot s_i} \tag{8.40}$$

其中

$$s'_i = \begin{cases} s_k - 1 & \text{如果 } i = k, \text{对于一定 } k \\ s_i & \forall\, i, \text{除非 } i = k \end{cases} \tag{8.41}$$

因此，

$$\begin{aligned} \Delta E &= E(s') - E(s) \\ &= -\frac{w_1}{N_d}\sum_{i=1}^{N_d} G \cdot s'_i \cdot \mathrm{e}^{-G \cdot s'_i} + \frac{w_2}{2} + \frac{w_1}{N_d}\sum_{i=1}^{N_d} G \cdot s_i \cdot \mathrm{e}^{-G \cdot s_i} \\ &= \frac{w_2}{2} - \frac{w_1 \cdot G}{N_d}\left[s'_k \mathrm{e}^{-G \cdot s'_k} - s_k \cdot \mathrm{e}^{-G \cdot s_k}\right] \\ &> \frac{w_2}{2} - \frac{w_1 \cdot G}{N_d} \cdot \mathrm{e}^{-(G \cdot s_k)}\left[(s_k - 1) \cdot \mathrm{e}^{G} - (s_k - 1)\right] \\ &\geqslant \frac{w_2}{2} - \frac{w_1}{N_d} \cdot (1 - \mathrm{e}^{-G}) \cdot \mathrm{e}^{-1} \geqslant 0 \\ &\Rightarrow w_2 \geqslant \frac{2w_1}{N_d} \cdot (1 - \mathrm{e}^{-G}) \cdot \mathrm{e}^{-1} \end{aligned} \tag{8.42}$$

在此仅考虑了一种特殊情况，表明 w_1 和 w_2 的选择是与 G 和 N_d 有关的。要取得更好的解答，必须根据 G 和 N_d 调整拉格朗日参数。w_3 项是一个弱约束，在后面介绍的模拟实验中取 $w_3=1$。

3. 均场公式

与 SBS 问题相似，这一问题的均场等式为

$$v_{ij} = \frac{1}{2} + \frac{1}{2}\tanh\left(-\frac{1}{2T}\frac{\partial E(v,t)}{\partial v_{ij}}\right), \quad \forall\, i,j \tag{8.43}$$

其中对于 $1<i<Nt$，$1\leqslant j\leqslant N'$，$v_{ij}=<s_{ij}>T$，且

$$\boldsymbol{v} = \{v_1, v_2, \cdots, v_{N_d}\}, \quad v_i = \{v_{i1}, v_{i2}, \cdots, v_{iN'}\}$$

4. 临界温度

临界温度被定义为剧烈状态转移开始稳定于 0 或 1 处时的温度。从(8.43)式可以看到，每个 v_{ij} 在高温时都在 1/2 左右浮动，且状态转移是慢的。因此存在一个快速状态转移开始变为稳定状态的临界温度。这里没有像在 SBS 问题中那样导出一个闭式解的估值，而是用试错法获得临界温度。也就是使温度从很高处缓慢下降，在每个温度处只作一次扫描，在每次扫描结束时，计算平均绝对误差

$$\varepsilon = \frac{1}{N_d \cdot N'}\sum_{i=1}^{N_d}\sum_{j=1}^{N'} |v_{ij}(t + \delta t) - v_{ij}(t) \tag{8.44}$$

其中 t 为扫描开始的时间，而 $t+\delta t$ 是扫描结束的时间。当 $\varepsilon\geqslant 0.1$ 时，上述步骤停止，且对应温度即为临界温度。

5. 退火流程

在此用到下列退火流程：

$$T_{n+1} = \frac{T_n}{1 + \alpha \cdot n} \tag{8.45}$$

其中 α 是一个小的正数，n 是迭代变量。

6. 终止准则

有两个终止准则。一个是在每个温度处扫描的终止准则，另一个是整个退火过程的终止准则。

在每个温度下，每个神经元的取值都依据(8.43)式更新。扫描在 $\varepsilon \leqslant \delta_1$ 时终止，其中 δ_1 是一个小的正数。另外，在某个温度下，这一情况可能在很多次扫描后仍不会出现。为了避免无穷无尽的扫描，扫描进程在扫描次数达到一个给定数值时被强行终止。然后，降低温度，一个新的迭代过程开始。

所有 v_{ij} 在稳态时都应收敛于 0 或 1。因此，采用下列收敛准则

$$\frac{1}{N_d \cdot N'} \sum_{i=1}^{N_d} \sum_{j=1}^{N'} v_{ij}(1 - v_{ij}) < \delta_2 \tag{8.46}$$

其中 δ_2 是一个小的正值。当准则被满足时，所有神经元的取值都为 1 或 0，且最优帧模式的间距为

$$s_i = \sum_{j=1}^{N'} \left\lceil v_{ij} - 0.5 \right\rceil 2^{j-1}, \ \forall\, i \tag{8.47}$$

7. 仿真结果

作为一个示例，图 8.3 给出了用均场退火从下列参数得到的数据吞吐量：

$$N = 40、N_d = 5、T_c = 5、\delta_1 = 0.05、\delta_2 = 0.01，且\ \alpha = 0.01$$

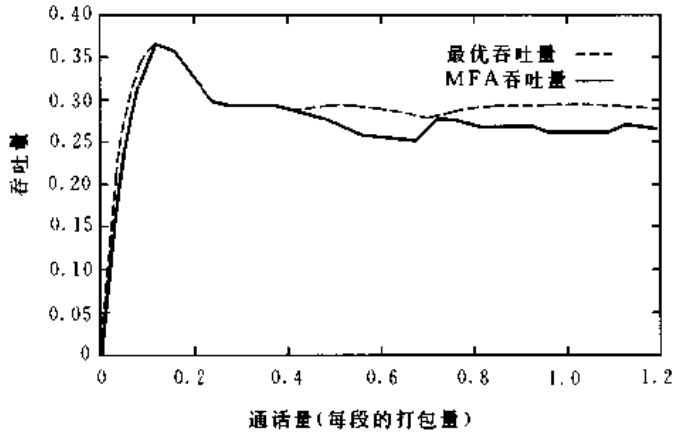


图 8.3 数据吞吐量与通话量的关系曲线

(8.37)式中 w_1 和 w_2 的关系是 G 和 N_d 的函数。有实验表明 w_2 不能随意设为一个大的正值，即使满足(8.42)式。一个大的 w_2 会给约束太大的权重。在这种情况下，退火过程可能会把系统带到一个虽满足约束但能量较高的状态。其结果是一个局部最优解。在这里的仿真中，这些参数凭经验选为：当 $G \geqslant 0.4$ 时，$w_1 = 750$，$w_3 = 1$，且 $w_2 = 750$；当 $G < 0.4$ 时，$w_2 = 6.5$。仿真结果表明均场退火方法给出的结果是或接近于全局最优。定义拉格朗日参数间更严格的关系并且根据 G 和 N_d 动态调整这些参数依然是一个挑战性的研究课题。

8.3 总结

本章通过介绍均场退火在两类远程通信最优化问题中的应用，概述了将待解问题映射到可由均场退火求解的框架中去的方法，并推导和讨论了诸如拉格朗日参数和临界温度等与问题本身有关的关键参数。随着 VLSI(超大规模集成电路)技术的飞速发展，适于实时应用的均场退火方法正在由可能变为现实。

均场退火可应用于许多其它通信问题，例如卫星通信网络的管理[15]和分组无线广播网络的播出调度[167]。IEEE Journal on Selected Areas in Communications 的专集[151]集中探讨了计算智能在高速网中的应用。

8.4 习题

8.1 证明(8.8)式的有效性。

8.2 推导(8.12)式给出的 SBS 问题的均场等式。

8.3 利用引理 8.1 估计 8.1.5 节描述的 SBS 问题的临界温度。

8.4 用模拟退火求解本章给出的两个例子。

第9章 点模式匹配

形状识别是计算机视觉和模式识别中的一个重要任务。术语"形状"指的是一个物体的一组静态空间特征中的相对距离不变的几何性质。这些静态空间特征就是物体的形状特征[17]。人的眼睛所感受的很多视觉数据对于识别的目的来说是高度冗余的。从人的视觉系统的观点来看,沿物体轮廓的某些支配点具有丰富的信息含量,足以用来表征物体的形状。因此用来作为形状特征的点的点模式匹配(PPM,point pattern matching)是一项关键的视觉课题。

本章所讨论的PPM问题就是在一个被观测的点模式中寻找这样一个子集,使其通过变换在某种最优意义上与一个模型点模式的子集相匹配。这些点的顺序信息是未知或已被提供。一般的情况是点分布在n维空间,且根据问题的几何和环境约束条件对变换加以规定。由于图象本来就是二维的,因此这里只考虑二维(2D)点模式,但是所提出的算法并不局限于二维点模式。大量文献报导过有关平面物体识别的研究工作,例如,文献[9],[16],[17],[18],[20],[21],[90],[148],[161],[140],[141],[142]。现有的用点作为特征的方法可能要求这些点排列顺序的先验知识。本章回顾了在不存在点的顺序知识的情况下,在文献[9]和[16]中讨论过的点模式匹配方法。

9.1 问题的表述

给定下式定义的两个点集:

$$\begin{aligned} P &= \{P_i : P_i \in R^N; \quad i = 1,2,3,\cdots,m\} \\ O &= \{O_i : O_i \in R^N; \quad i = 1,2,3,\cdots,n\} \end{aligned} \tag{9.1}$$

求这样一个指派$P' \to O'$,$P' \subseteq P$和$O' \subseteq O$使$T(P')$和O'之间的匹配误差最小化。下面要加以定义的匹配误差反映出了匹配程度。匹配误差越小,匹配品质越好。T是预先确定的相似变换:$T \equiv${旋转,平移,尺度}。应注意它不同于为了使两幅图象重合,用几何变换使图象对准的问题[135]。

令O是被观测的点模式,P是模型点模式。二维相似变换由映射$X \to U$定义,X和$U \in R^2$,且

$$\begin{bmatrix} u \\ v \end{bmatrix} = S \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} e \\ f \end{bmatrix} \tag{9.2}$$

式中

$$X = \begin{bmatrix} x \\ y \end{bmatrix}, \quad U = \begin{bmatrix} u \\ v \end{bmatrix}^T \tag{9.3}$$

S=尺度因子,θ=旋转角,e=x轴平移,f=y轴平移。令$a = S\cos\theta$,$b = S\sin\theta$,则相似变换

可以重新写成

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} a & b \\ -b & a \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} e \\ f \end{bmatrix} \tag{9.4}$$

为了确定模型点和被观测点之间的匹配程度，使 P 最优映射到 O 的相似变换参数是在最小平方误差的意义上得到的。在变换 $T[(x_i,y_i)]=[(u_i',v_i')]$ 下映射 P，得

$$\begin{bmatrix} u_i' \\ v_i' \end{bmatrix} = \begin{bmatrix} a & b \\ -b & a \end{bmatrix} \begin{bmatrix} x_i \\ y_i \end{bmatrix} + \begin{bmatrix} e \\ f \end{bmatrix} \tag{9.5}$$

模型点集和被观测点集之间的平方误差于是可以定义为

$$\begin{aligned} \varepsilon^2 &= \sum_{i=1}^{n} \left((u_i - u_i')^2 + (v_i - v_i')^2 \right) \\ &= \sum_{i=1}^{n} \left((u_i - ax_i - by_i - e)^2 + (v_i + bx_i - ay_i - f)^2 \right) \end{aligned} \tag{9.6}$$

为了求取使平方误差最小的参数，只要对(9.6)式定义的平方误差对 a,b,e 和 f 求偏导并使其等于 0，即

$$\frac{\partial \varepsilon^2}{\partial a} = 2\sum_{i=1}^{n} \left((u_i - ax_i - by_i - e)(-x_i) + (v_i + bx_i - ay_i - f)(-y_i) \right) = 0$$

$$\frac{\partial \varepsilon^2}{\partial b} = 2\sum_{i=1}^{n} \left((u_i - ax_i - by_i - e)(-y_i) + (v_i + bx_i - ay_i - f)(-x_i) \right) = 0$$

$$\frac{\partial \varepsilon^2}{\partial e} = 2\sum_{i=1}^{n} (u_i - ax_i - by_i - e)(-1) = 0 \text{ 和}$$

$$\frac{\partial \varepsilon^2}{\partial f} = 2\sum_{i=1}^{n} (v_i - bx_i - ay_i - f)(-1) = 0$$

解以上方程，并重写成矩阵形式

$$A[a \quad b \quad e \quad f]^{\mathrm{T}} = C \tag{9.7}$$

其中

$$\boldsymbol{A} = \begin{bmatrix} \sum (x_i^2 + y_i^2) & 0 & \sum x_i & \sum y_i \\ 0 & \sum (x_i^2 + y_i^2) & \sum y_i & -\sum x_i \\ \sum x_i & \sum y_i & n & 0 \\ \sum y_i & -\sum x_i & 0 & n \end{bmatrix} \tag{9.8}$$

$$\boldsymbol{C} = \begin{bmatrix} \sum (u_i x_i + v_i y_i) \\ \sum (u_i y_i - v_i x_i) \\ \sum u_i \\ \sum v_i \end{bmatrix} \tag{9.9}$$

可求得使平方误差最小的最优变换参数为

$$[a \quad b \quad e \quad f]^{\mathrm{T}} = [\boldsymbol{A}^{-1} \quad \boldsymbol{C}] \tag{9.10}$$

但是这样得到的最小平方误差只是定量表征了模型点集的子集 P' 和被观测点集的子集 O' 之间的匹配程度。为了度量两个模式点集之间的全面的匹配程度，定义下式的启发式度量[17]

$$\hat{\varepsilon} = \begin{cases} \dfrac{\varepsilon^2}{S \cdot k}\left(1 + \left(\dfrac{m-2}{k-2}\right)\log_2\left(\dfrac{m-2}{k-2}\right)\right) & k \geqslant 3 \\ \infty & k = 0,1,2 \end{cases} \tag{9.11}$$

为匹配误差，它体现了对不完全匹配的惩罚。这里 k 表示匹配上被观测点的模型点的数量，S 表示尺度因子。应该注意，上述方程意味着两点或更少的点的匹配被认为是不确定的。对数项作为模式集合不完全匹配的惩罚因子。当两个模式所有的点都匹配上时($k=m$)，匹配误差就等于规范的最小平方误差。以上所定义的问题是一个组合优化问题，并可用前几章所阐述的技术很好地进行求解。这样一个问题的解空间的大小是

$$\sum_{i=0}^{\min(m,n)} \binom{m}{i}\binom{n}{i}(i!) \tag{9.12}$$

其中 $\binom{n}{m} = \dfrac{n!}{m!\ (n-m)!}$；$n!\ = n(n-1)(n-2)\cdots(2)(1)$

为了说明 PPM 问题，我们把它置于用模拟退火和进化方法求解的框架上来讨论。

9.2 模拟退火框架

把 PPM 问题置于模拟退火框架内求解需要一个编码方案(一种表示点模式的方法)、一个合适的代价函数、为生成新的指派(构形或状态)的扰动规则、接受规则、冷却流程和收敛准则。一旦确定了框架，PPM 问题就可以用第 4 章所讨论的模拟退火方法加以求解。

9.2.1 编码方案

为了找到最佳指派，描述模式点之间指派的数据结构在以下方面构成最理想的表示：

• 它是在问题定义域中的一个简单和直接的标记与指派，并且是模式点的一种最紧凑表示(最小存储/暂存空间)。

• 在展开完备的解空间时，具有稳健性且不存在非法指派的可能性，在不完备模式集情况下，具有自然预防零标记的能力。

一个点模式用结点(即不同的点坐标的标记)组成的符号串来编码，从而形成搜索/解空间。一个码是两个点集合之间的一个指派。符号串中每一个单元的单元号相当于模型点集中的某个点，其值表示指派给这一点的被观测点。假设有 m 个模型点和 n 个被观测点。这样，一个码由 m 个单元组成，其中的每一个单元可以取从 0 到 n 之间的任一个整数(0 值表示没有一个被观测点能够与一个模型点匹配)。因此，符号串中的单元位置值(相应于模型点)与赋予此单元的数值(相应于被观察点)，在每一个单元值唯一的约束下(除了"0"以外，两个单元不能有相同的数值)，给出了从模型点到一个被观测点的匹配指派。例如考虑以下的 12 个单元的指派：

5	7	2	0	9	1	4	3	10	12	11	6

从左边算起的单元位置表示模型点标号；如第 6 个单元对应于第 6 个模型点。相应地，第 1 个模型点被指派到第 5 个被观测点，第 2 个到第 7 个，第 3 个到第 2 个，第 4 个未被指派，第 5 个到第 9 个，等等。这样，每个码(指派)类似于液体的状态。

9.2.2 能量函数

代价函数或目标函数应能反映不适当指派所造成的误差，并对不完全的匹配给以惩罚。一个好的指派应产生小的匹配误差，反之亦然。(9.11)式中所定义的匹配误差是代价函数的一种合适选择。从统计力学的角度看，一个解的代价类似于第 4 章所讨论的物质的一种状态的能量。

9.2.3 扰动规则

扰动规则可采用不同的机制。一个用于求解 TSP 问题的成对改变的产生机制能够类似地在此得到应用。这里对此机制略加变化。我们从 9.2.1 节所给的符号串的例子继续讨论。首先在范围 1 到 m(模型点的数量)和范围 1 到 n(被观测点的数量)随机地产生两个整数，例如“3”和“6”。第 1 个随机数表示符号串的这样一个单元位置，其值将被代表被观测点的第 2 个随机数所取代。因此，第 3 个位置的单元值被置换为“6”，由于单元值只能是唯一的，因此原来单元值是“6”的第 12 个位置上的值要被置换成第 3 个位置原来的数值“2”。这样置换后新的符号串就成为

5	7	6	0	9	1	4	3	10	12	11	2

9.2.4 接受规则

这个规则用于决定接受还是丢弃下一个搜索结点。能得到较低代价的指派总是被接受的，而为了避免陷入局部最小值，一个较高能量的新的指派有时也被接受。我们采用与(4.28)式类似的接受规则。必须注意，当温度降低时，接受概率趋近于 0，从而使得在低温时，具有较高代价状态的指派的接受概率大大减小。这就是为什么一个其局部最小值的代价与全局最小值的代价接近相等的搜索空间，有时可以在很低的温度下得到与最优值很接近的解。

9.2.5 冷却流程

第 4 章中所讨论的冷却流程是模拟退火技术中重要的一环。可以采用不同的冷却流程。我们可以预先启发式地确定一个最大搜索次数(即状态转移的总数)，使温度在这最终情况下的值为 0。冷却流程中的温度是线性下降的，那么温度按下式

$$T = T_0(1 - (n/n_{\max})) \tag{9.13}$$

降低，式中 $T_0=1$ 选为起始温度，n 表示第 n 次的搜索试验，$n_{\max}$表示这种搜索试验的最大

搜索次数。

9.2.6 停止准则

当得到一个最优解(匹配误差 $\hat{\varepsilon} \approx 0$)或者当冷却流程的温度 T 不能再降低(即 T 达到 0)时,退火过程自然终止。

9.3 进化程序设计

把 PPM 任务置于进化程序设计框架中求解[①] 要有一个表示方法、一个适度函数、一组遗传算子以及控制遗传算子的规则。进化程序设计的核心有以下几部分:

(1) 起始群体的初始化　随机产生一个搜索结点数 $N=30$ 的起始群体。

(2) 对群体符号串的代价估值　适度函数决定了群体空间每一个染色体的适度值。

(3) 后代的繁衍　如后面将给出的,根据群体中符号串的适度值,产生新符号串的群体。

(4) 繁衍后代的重新组合　用交叉、突变和转位遗传算子使繁衍后代进行随后的重组。

(5) 收敛准则　重复步骤(2)～(4),直到收敛或者经历了预先给定的代数。

9.3.1 解空间的表示

用与模拟退火相同的一种指派的符号串表示进行进化程序设计。除了在 9.2.1 节中所描述的特性外,在用遗传算子对符号串表示进行运算时,也必须只产生有效的指派。这样就不再需要耗时的修复算法。用结点组成的符号串表示每一指派(即各个点坐标的标号),从而形成搜索/解空间。

9.3.2 群体空间:规模、初始化及其利用

理论上,为了达到运算的高度并行性,希望代的规模尽可能大,甚至是无限的。实际上这是不可能的,因此群体的大小取经验值为 $N=30$。初始群体是在唯一性约束下通过随机产生染色体形成的。

有效的利用可用的解空间要避免适度值高的个体集中在一起(占统治地位),并保持一个进化群体,这个群体开始是探索性的,最终引向解域的有效搜索。

9.3.3 适度函数

适度(目标或代价)函数应能反映不适当的指派所造成的误差(基于最优参数的),并对部分的匹配进行惩罚。同时,一个好的指派应产生一个高的适度值,反之亦然。(9.11)式所定义的匹配误差满足上述要求,可用作适度值估算子。适度值与匹配误差成反比。

9.3.4 繁殖

在产生每一代以后,选择好的染色体以利用其基因遗传知识形成下一代的潜在染色

① 一个进化程序可以看成是一个经过加工修改的、能够引入和利用所讨论问题的结构的遗传算法。

体。通常采用的偏置轮盘选择过程有缺点。具有高适度值的染色体可能被多次挑选，使群体空间很拥挤，且由于不选择其它可能的染色体而造成群体空间不断狭窄，从而丢掉了有用的遗传信息。太多相同的染色体也造成计算资源的浪费(好的染色体信息利用因子不是无限的)。

为了克服上述缺点，可用一个修正的选择机制来实现有效的繁殖。通过执行以下步骤，不仅能修补前述机制的缺点[109]，而且还有显著加快收敛的优点。

(1) 在偏置轮盘方法的基础上，从 $P(t)$中选择 r 个(不一定是各不相同的)高适度值的染色体作为繁殖的父辈。

(2) 从 $P(t)$中确定性地除掉 r 个不同的适度值最小的染色体(不依赖于步骤(1)的选择)。这样得到的$(N-r)$个染色体进入繁殖库。

(3) 从步骤(1)～(2)得到的 N 个符号串组成的繁殖库中，通过一种遗传算子的运算对步骤(1)所得到的 r 个父辈进行重新组合以产生 r 个后代。

(4) 从步骤(2)得到的$(N-r)$个染色体和从步骤(3)得到的 r 个后代形成新群体 $P(t+1)$，从而完成新群体的选择过程，这是在重组之前的重要预处理步骤。此项技术使 r 个后代是从高适度值的父辈产生的，每一个后代在某种方式上与其它后代又不相同。其结果就是有可能形成比父辈的代价略高，而又能进入用于重新组合的下一个繁殖库的后代。就像模拟退火方法那样，这类似于偶然接受有较高代价的染色体的概念[2]。

由于执行了步骤(2)，最好的染色体总是与$(N-r-1)$个适度值高的个体一起进入下一次的产生过程。这样选择得到的群体有较丰富的遗传信息，有较大的潜力产生好的后代。

9.3.5 遗传算子

遗传或重组算子控制着新的信息的形成手段，且现有的信息也在染色体之间进行交换，以利于随后进化成有较高存活概率的染色体。在 EP 算法中，用满足 1 对 1 映射的三个这样的算子，从而避免了采用修补算法。

1. 突变

这里所用的突变算子是对染色体的单个基因进行操作的，这单个基因通常是在根据适度值计算所提取知识的基础上确定性地选择出来的。因此不像通常的突变具有探索性那样，这里的算子更具有实效性。这种基于知识的算子被用到的概率很大。在 PPM 问题中定义了三种突变算子，如下所述，每一种算子以不同的概率得到应用。

(1) 突变-1　这里，在染色体内部，把对用(9.6)式计算得到的平方误差和起最大作用的基因与起次大作用的基因进行交换。这是一种确定性作法，而且总是在每次迭代时对群体中最好的染色体进行。它也以某个概率在其它染色体上实现。假设有以下收敛前一步的符号串：

1	2	10	4	5	6	7	8	9	3	11	12

显然在第 3 和第 10 位置上的基因需要进行交换以达到最优匹配。在这一步，用突变-1 就能有效地加以实现。与通常的突变方式不同，这种情况下，突变-1 算子以概率为 1 地使下一代收敛。

(2) 突变-2　第一个基因的选择与突变-1 相同。把对误差作用第三大的基因选作第二个基因，或从染色体符号串中随机拾取一个基因与第一个基因进行交换。在有必要把对误差作用最大的基因与不是次大作用的基因进行交换时，需要这样的交换方法。事实上，对于大的染色体，可以把这个原则扩展到对平方误差和作用大的多个基因组合。这样搜索可以更快一些。

(3) 突变-3　把对误差作用最大的基因与任何其它基因进行交换可能造成不适当的指派结果，从而显著增加总体的平方误差。这种情况使得上述两种突变都没有用处，而使解总是停留在局部最小值。突变-3 通过随机选择两个互换的基因来补救这种情况。它提供了比可用解空间更大的搜索空间。因此执行这种突变方式的概率要保持在很小数值。无论如何，由于它提供了从明显的局部最小值跳出的机制，所以可以起到重要的作用。

2. 均匀交叉

下面定义一种基于位置的均匀交叉，使在染色体间实现位置遗传信息(而不是阶保持的遗传信息)的交换，它是多点交叉概念的一般化[150],[153]。具体做法如下：

(1) 在规定概率的基础上，选择两个用于交叉的符号串。

(2) 对于第一个符号串的每一个基因位置，如果该基因对误差作用大，而在第二个符号串中相应基因的误差作用小，则把第二个符号串的该基因复制到第一个符号串的相应位置上；同时把该位置原来的基因复制到第一个符号串中与用于置换的基因相同的位置上。

(3) 对第二个染色体复重步骤(2)。

通过这种交叉，从两个父辈中产生的两个后代能保持好的基因位置，并能对坏的基因位置用从其它染色体的好基因位置的基因来改换。这种交叉非常有效，因为它能在不干扰好基因位置的情况下取代多个坏基因位置，从而使误差值决定性地得以降低。

3. 反转

反转算子对它所操作的染色体进行$(n+1)$补运算，从而完全改变了正在进行的搜索空间。例如，如果一个单元值的范围是从 0 到 n，则一个单元值为 x 的$(n+1)$补就是$(n+1-x)$。以下是当 $n=15$ 时，一个符号串及其$(n+1)$补的一个例子：

1	2	6	5	9	13	15	14	4	11	7	3
15	14	10	11	7	3	1	2	12	5	9	13

注意，反转操作使符号串产生了巨大的变化，因此实施反转操作的概率要比实施互换的概率小得多。最好随着代数的增长，使这种算子的概率为 0。只有在解看来陷入到局部最小值时，才进行这种操作。

9.3.6 模拟结果

试验结果表明进化编程技术在收敛速度和稳健性方面超过了其它通常算法。这是由于消除了其它算法所固有的无效计算环境。新的基于领域知识的互换和交叉算子是提高算法速度的起作用的因素。这里用三组点模式的结果对此加以说明。图 9.1(a)示出了模型点模式。考虑以下三种示例:①

例 1：观察模式＝T[模型模式]

例 2：观察模式＝T[模型模式]＋噪声

例 3：观察模式＝T[模型模式]＋噪声－n 个点，点 2，点 10 和点 11 丢失。

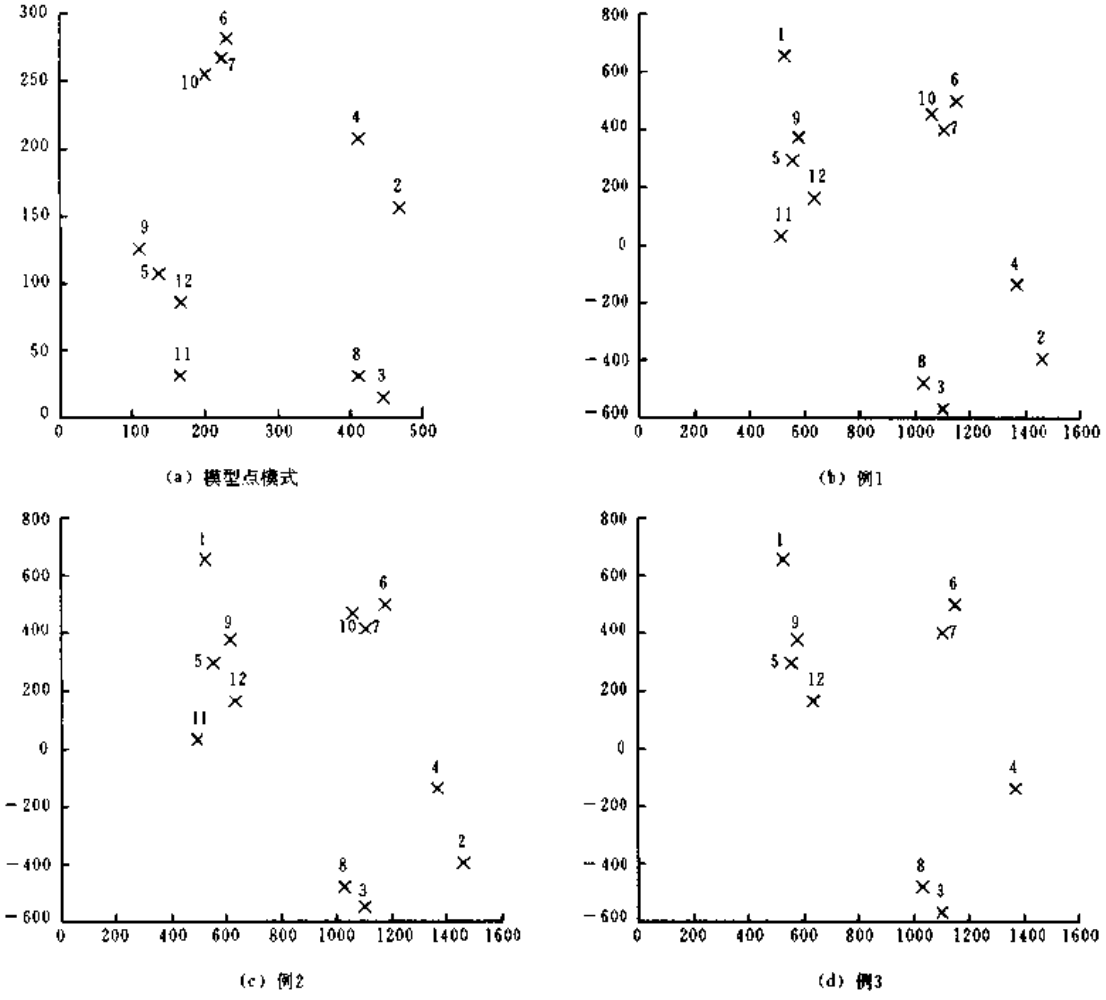


图 9.1 模型点模式和观察模式

例 1：T[模型点模式]，

例 2：T[模型点模式]加噪声，

例 3：T[模型点模式]加噪声并去掉 3 个点

① 第三种情况有三个丢失点，每一个符号串中赋有三个“0”。

$T[\cdot]$表示由旋转$\pi/4$,尺度因子为3,在x和y轴平移量分别为100和300所组成的相似变换。噪声为零均值,方差为9的高斯随机变量。例1,2,3三种示例分别示于图9.1(b),(c)和(d)。在几次迭代以后,算法就完成了匹配,图9.2给出了上述各种情况的收敛速率。表9.1总结了模拟结果。它表示的只是每种情况下的典型运行结果。

表 9.1 模拟结果(每例运行一次样本)

实　　验	收敛所要的代数	收敛时误差
例 1	8	0
例 2	6	43.62
例 3	12	90.67

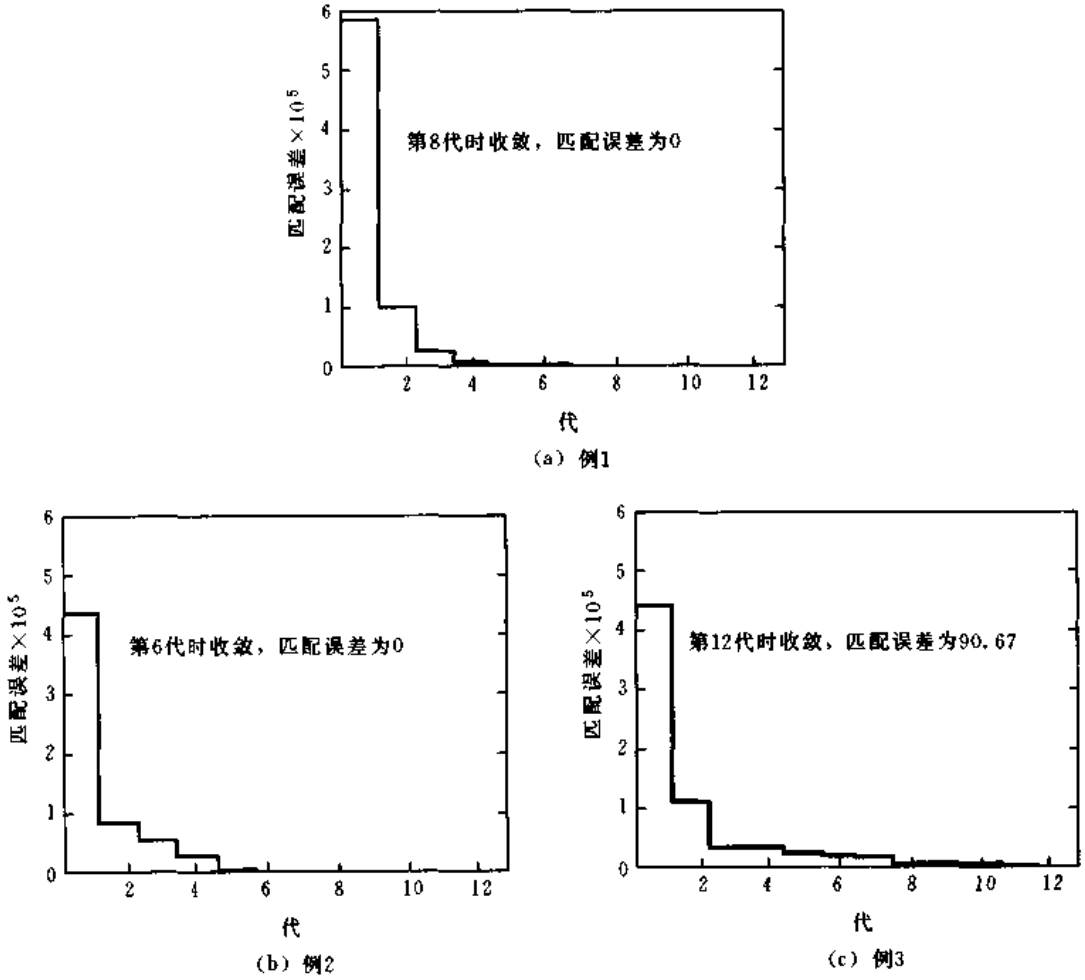


图 9.2 为得到正确匹配的进化程序设计的收敛率

例 1：T[模型点模式]，

例 2：T[模型点模式]加噪声，

例 3：T[模型点模式]加噪声并去掉 3 个点

9.4 总结

本章的主要目的是说明怎样把一个模式识别问题，例如 PPM 问题，映射到能用计算智能方法进行求解的框架中。在建立框架和选择设计参数时有许多自由度。本章所概述的过程可以作为解决其它模式识别问题的示例。

第10章　多处理器调度

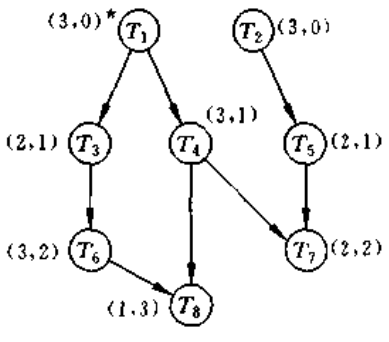
多处理器调度的一般问题可表述为在一个多处理器系统中调度一组部分有序的计算任务以使一组性能指标最优化。问题的难度在很大程度上取决于表示计算任务间先后次序关系的任务图的拓扑结构、多处理器系统的拓扑结构、并行处理器的个数、任务处理时间的均匀程度以及所选性能指标。一般来讲，多处理器调度问题(MSP)即使在简化了的假定前提下也很难解[57]，因此启发式算法被提出以求取MSP的最优解或次优解。

已有多种不同的MSP解法被提出[66][96]。卡沙拉(Kashara)和那瑞塔(Narita)[95][96]提出了一个启发式算法(关键路径/最直接节点优先)和一个最优化/近似算法(深度优先/隐式启发搜索)。陈(Chen)等[39]发展了一个利用从费南德(Fernandez)和布赛尔(Bussell)边界推导出的启发知识的MSP状态空间算法(A*算法)。海尔斯车姆(Hellstrom)和卡耐尔(Kanal)[76]将MSP映射到一个神经元网络模型——非对称均场网络。张(Zhang)等[178]开发了一个用霍普费尔德网络的均场退火求解MSP的方法。

在这一章中，介绍求解MSP的均场退火方法和遗传算法。这里所讨论的只是基于确定性模型的MSP，即计算任务的运行时间及其之间的关系是已知的。任务之间的先后关系由一个非循环有向图表示，且任务的运行时间可为非均匀的。同时，假定多处理器系统是均衡的、无优先权的，即每个处理器都是一样的，且一个处理器在完成当前任务前不会执行下一个任务。

10.1　模型与定义

一组偏序的计算任务可由一个非循环有向图表示，$TG=(V,E)$，其中V为一个节点的非空有限集，E为连接节点的有向连线的有限集。节点的集合$V=\{T_1,T_2,\cdots,T_m\}$表示要执行的一组计算任务，而有向连线的集合$E=\{e_{ij}\}$(e_{ij}表示从节点T_i到T_j的有向连线)则表示任务中的偏序或先后关系$\succ$。如果$T_i\succ T_j$，任务T_i必须在T_j开始执行之前完成。一个具有8个任务的简单任务图TG如图10.1所示。



*(执行时间，高度)

图10.1　任务图TG

一个具有p个多处理机系统的任务图最优化调度问题是要在保证前后关系的条件下将计算任务分配到各个处理机上，以使所有任务能在最短时间内完成。最后一个任务完成的时刻被称为调度的完成时间(FT)。图10.2描绘了一个以甘特图给出的两处理器的任务图TG示例。这一调度的完成时间为

11 个单位时间。临界路径长度对于调度的完成时间来说是一个重要下限。一个任务图的临界路径长度 t_{cp} 定义为完成任务图中所有任务所需的最短时间。

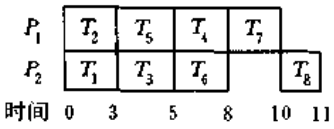


图 10.2 用甘特图表示的一个两处理器 TG 的调度

下列定义和术语将用于任务图 $TG=(V,E)$：

- 如果 $e_{ij}\in E$，则 T_i 是 T_j 的父节点，且 T_j 是 T_i 的子节点。
- 如果存在从 T_i 到 T_j 的序列有向连线，则 T_i 是 T_j 的前辈节点，且 T_j 是 T_i 的后辈节点。
- $PRED(T_i)$ 是 T_i 的前辈节点集；
- $SUCC(T_i)$ 是 T_i 的后辈节点集；
- $et(T_i)$ 是 T_i 的运行时间；
- 任务 T_i 的高度定义为

$$height(T_i)=\begin{cases}0 & \text{如果 } T_i \text{ 没有前辈节点}\\ 1+\max_{T_j\in PRED(T_i)} height(T_j) & \text{其它}\end{cases}$$

该高度公式以其特殊的方式给出了任务之间的先后关系。实际上，如果任务 T_i 是任务 T_j 的前辈节点(亦即 T_i 必须在 T_j 之前执行)，则一定会有 $height(T_i)<height(T_j)$。不过，如果两个任务之间没有路径，则它们之间没有先后关系，因而这两个任务的执行顺序是任意的。

10.2 均场退火

本节讨论利用霍普费尔德网络的均场退火求解 MSP 的基本方法。运用霍普费尔德神经元网络求解最优化问题的基本方法包括两个主要步骤：

(1) 为待解问题建立一个适当的霍普费尔德能量函数。

(2) 选取并设计一个最小化能量函数的递归机制。霍普费尔德能量函数的最小化可分别运用第 3 章讨论过的确定性更新规则、第 4 章介绍过的模拟退火方法以及第 5 章描述过的均场退火方法。

10.2.1 MSP 霍普费尔德能量函数

任务调度问题的霍普费尔德能量可用类似于第 5 章中介绍的方法推导出来以反映问题的代价和约束。这里让我们首先考虑所有任务的运行时间都相同的情况，然后再推广到不同任务具有不同运行时间的情况。

如图 10.3 所示的 MSP 可依下列约束被映射到处理器-时间段空间：

(1) 每个任务只占有一个时间段。

(2) 因为共有 P 个处理器，在同一时间段中最多可有 P 个任务被处理。

(3) 如果在当前时间段上运行的任务 T_i 前面有 k 个任务与其相连，这 k 个任务必须在前面的时间段上已被处理。

令

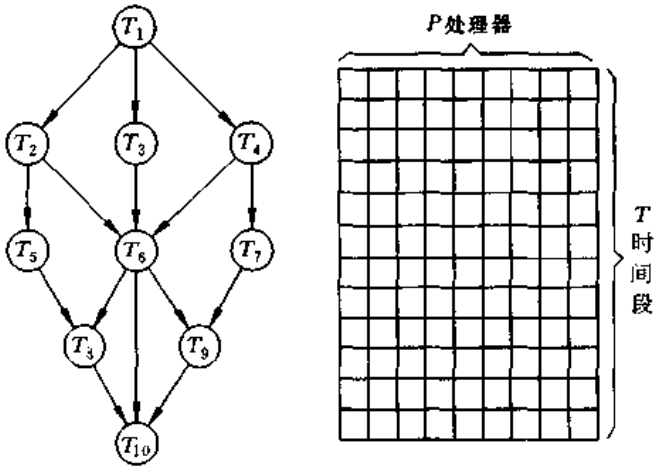


图 10.3 将任务 T 映射到处理器-时间段(p,t)坐标上去

$$s_{ia}=\begin{cases}1 & \text{如果任务 } T_i \text{ 在时间段 } a \text{ 运行}\\ 0 & \text{否则}\end{cases} \tag{10.1}$$

注意，对于 MSP，神经元变量取值 1 或 0 而不是 1 或 −1。定义任务 T_i 与任务 T_j 之间的连接性 c_{ij} 如下：

$$c_{ij}=\begin{cases}1 & \text{如果 } T_i \text{ 在 } T_j \text{ 之前且与之相连，} i\neq j\\ 0 & \text{否则}\end{cases} \tag{10.2}$$

前述三项约束可由下面能量函数相应表达：

$$E_1=\sum_i\sum_j\sum_a\sum_b\delta_{ij}(1-\delta_{ab})s_{ia}s_{jb}=\sum_i\sum_a\sum_{b\neq a}s_{ia}s_{ib} \tag{10.3}$$

$$E_2=\sum_a\Big(\Big(\sum_i s_{ia}-P\Big)^2\Big(\sum_j s_{ja}\Big)\Big) \tag{10.4}$$

$$E_3=\sum_i\sum_a s_{ia}\sum_{j=1}^{i-1}c_{ij}\Big(1-\sum_{b=1}^{a-1}s_{jb}\Big)^2 \tag{10.5}$$

(10.3)式对应于约束(1)。当 E_1 被最小化(理论上 $E_1=0$)时，$s_{ia}=s_{ib}=1(a\neq b)$的情况不存在。也就是说，每个任务的执行不超过一次。(10.4)式中的 E_2 用来施行约束(2)。一旦达到 E_2 的最小值，相应调度应为最有效率的，因为这一约束尽可能地使很多处理器被利用或使所有处理器在该单位时间闲置(未被任何处理器占用的单位时间不被计入总运行时间)。显然，(10.5)式中的 E_3 总为正值。如果约束(3)被满足，可得 E_3 的最小值。一个违反先后关系的调度的 E_3 值会较大。

另外，引入下列辅助函数可以加速收敛：

$$E_4=\sum_i\Big(1-\sum_a s_{ia}\Big) \tag{10.6}$$

$$E_5=\sum_i\sum_a s_{ia}(1-s_{ia}) \tag{10.7}$$

E_4 保证每个任务只在一个时间段里运行，而 E_5 将 s_{ia} 强行置为 0 或 1。

任务图的总能量 E_T 为

$$E_T=\alpha E_1+\beta E_2+\gamma E_3+\xi E_4+\lambda E_5 \tag{10.8}$$

其中 α,β,γ,ξ 和 λ 是用于为 E_1,E_2,E_3,E_4 和 E_5 加权的相应拉格朗日参数[105]。

在每个任务都可有不同的运行时间的情况下，任务图可分解为一个不同的任务图，其中每个新的任务可为原有任务图中任务的子任务。假定原有任务图中的每个任务都可以被分解为一些运行时间相等的子任务。在这种情况下，我们得到一个所有任务都具有同样运行时间的新的任务图，并可用上述公式求解。

10.2.2 均场近似

在 $s_i=1$ 或 0 的情况下，可以证明(见第 5 章)均场退火公式为

$$\begin{aligned} v_i &= (1+\mathrm{e}^{-u_i})^{-1} \\ &= \left[1+\mathrm{e}^{\left(\frac{\partial E}{\partial v_i}\frac{1}{T}\right)}\right]^{-1} \\ &= \frac{1}{2}\left[1+\tanh\left(-\frac{1}{2T}\frac{\partial E}{\partial v_i}\right)\right] \end{aligned} \tag{10.9}$$

10.2.3 MSP 的均场公式

从(10.3)～(10.7)式及(10.9)式可得

$$\frac{\partial E_1}{\partial v_{ia}} = \sum_{b\neq a} 2v_{ib} \tag{10.10}$$

$$\frac{\partial E_2}{\partial v_{ia}} = 2\left(\sum_j v_{ja} - P\right)\sum_j v_{jc} + \left(\sum_j v_{ja} - P\right)^2 \tag{10.11}$$

$$\frac{\partial E_3}{\partial v_{ia}} = \sum_j^{i-1} c_{ij}\left(1 - \sum_{b=1}^{a-1} v_{jb}\right)^2 \tag{10.12}$$

$$\frac{\partial E_4}{\partial v_{ia}} = 2\left(\sum_b v_{ib} - 1\right) \tag{10.13}$$

$$\frac{\partial E_5}{\partial v_{ia}} = 1 - 2v_{ia} \tag{10.14}$$

将(10.10)～(10.14)式代入(10.9)式，迭代更新公式变为：对所有 i 和 a，有

$$v_{ia} = \frac{1}{2}\left\{1+\tanh\left[-\frac{1}{2T}\left(\alpha\frac{\partial E_1}{\partial v_{ia}} + \beta\frac{\partial E_2}{\partial v_{ia}} + \gamma\frac{\partial E_3}{\partial v_{ia}} + \xi\frac{\partial E_4}{\partial v_{ia}} + \lambda\frac{\partial E_5}{\partial v_{ia}}\right)\right]\right\} \tag{10.15}$$

10.2.4 数值解法和仿真

在得到 MSP 的霍普费尔德能量函数和均场退火公式后，第 5 章给出的均场退火算法可用于求解 MSP。

一个由图 10.4 给出的 17-任务图被用于仿真求解以给出这一算法的示例。图 10.5(a)，(b)和(c)分别给出了这一任务图在 2，3，4 个处理器情况下的结果。在仿真计算中，(10.15)式的参数 α,β,γ,ξ 和 λ 分别为 7.5，3.5，1.0，1.0 及 1.0。上述仿真均获得最优解。

一个更复杂的 40-任务图示例由图 10.6 给出。这一拥有 40 个任务的 MSP 在 3，5，7 个处理器情况下的模拟退火调度分别为图 10.7(a)，(b)和(c)。这些解答的完成时间分别为 15，9，8，而最优解的完成时间相应为 14，9，7。

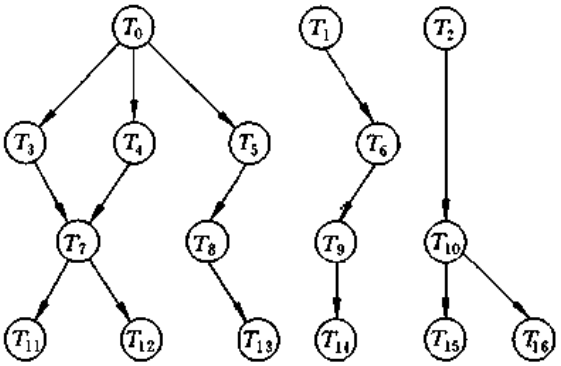


图 10.4　一个 17-任务图示例

T_0	T_1
T_2	T_3
T_4	T_6
T_6	T_9
T_7	T_8
T_{10}	T_{11}
T_{12}	T_{16}
T_{13}	
T_{14}	T_{15}

(a) 总执行时间

T_0	T_1	T_2
T_3	T_4	T_{10}
T_5	T_6	T_7
T_8	T_9	T_{16}
T_{14}	T_{15}	
T_{11}	T_{12}	T_{13}

(b) 总执行时间

T_0	T_1	T_2	
T_3	T_4	T_5	T_{10}
T_6	T_{15}	T_{16}	
T_7	T_8	T_9	
T_{11}	T_{12}	T_{13}	T_{14}

(c) 总执行时间

图 10.5　图 10.4 中任务图示例的均场退火调度

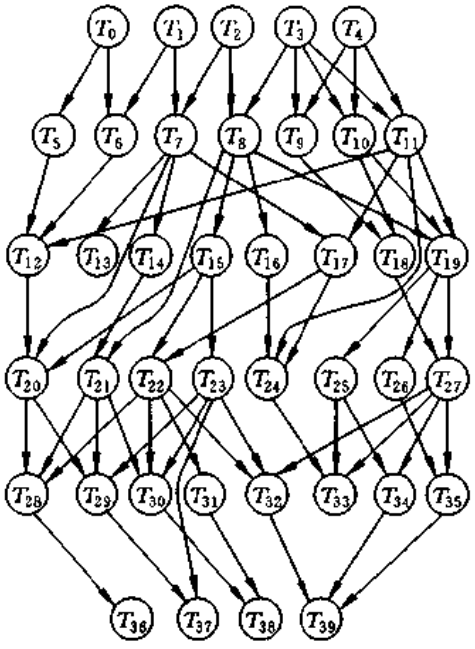


图 10.6　一个 40-任务图示例

T_0	T_1	
T_3	T_4	T_6
T_2	T_5	T_9
T_{10}	T_{11}	
T_8	T_{12}	T_{18}
T_7	T_{15}	T_{16}
T_{13}	T_{17}	T_{23}
T_{14}	T_{19}	T_{24}
T_{25}	T_{26}	T_{27}
T_{20}	T_{21}	T_{34}
T_{22}	T_{35}	
T_{30}	T_{31}	T_{32}
T_{29}	T_{38}	
T_{28}	T_{33}	
T_{36}	T_{37}	T_{39}

(a) 总执行时间

T_0	T_1	T_2	T_3	
T_4	T_5	T_6	T_7	
T_{10}	T_{11}	T_{13}	T_{14}	
T_8	T_9	T_{12}	T_{17}	
T_{15}	T_{16}	T_{18}	T_{21}	
T_{19}	T_{20}	T_{22}	T_{23}	T_{24}
T_{25}	T_{26}	T_{27}	T_{28}	T_{29}
T_{30}	T_{31}	T_{32}	T_{34}	T_{35}
T_{33}	T_{36}	T_{37}	T_{38}	T_{39}

(b) 总执行时间

T_0	T_1	T_2	T_3	T_4		
T_5	T_6	T_7	T_8	T_9	T_{10}	T_{11}
T_{13}	T_{14}	T_{15}	T_{16}			
T_{12}	T_{17}	T_{18}	T_{19}	T_{21}	T_{23}	
T_{20}	T_{22}	T_{24}	T_{25}	T_{26}	T_{27}	
T_{28}	T_{29}	T_{30}	T_{31}			
T_{32}	T_{33}	T_{34}	T_{35}			
T_{36}	T_{37}	T_{38}	T_{39}			

(c) 总执行时间

图 10.7　图 10.6 中任务图示例的均场退火调度

在这一节中，我们示范了用模拟退火求解 MSP 的基本方法。实验结果表明均场退火是求解 MSP 的一个有效方法。运用这一方法的难度仅在于将待解问题转化为均场退火可解的形式。

10.3　遗传算法

正如第 6 章所述，运用遗传算法的三个关键元素为

(1) 符号串表示。

(2) 适度函数。

(3) 遗传操作。

用 GA 求解 MSP，为任务调度所设计的表达形式须使遗传操作可以有效地检验新的调度并寻找最优调度。

10.3.1 符号串表示

符号串表示必须能够唯一地表示搜索空间中的所有搜索节点。对于多处理器调度问题，一个有效的搜索节点(一个调度)必须满足下列条件：

(1) 计算任务之间的先后关系。

(2) 完整性和唯一性条件(每个任务都在调度中且只出现一次)。

一种满足这两个条件的可行方法是将调度表示为一些计算任务的表。每个表对应于在一个处理器上运行的计算任务。图 10.8 给出了图 10.1 中调度的表表示。表中任务的顺序保持了在一个处理器上运行的任务的先后顺序(处理器内先后顺序)而忽略了在不同处理器上运行的任务的先后顺序(处理器间先后顺序)。这是因为处理器间先后顺序关系在实际计算调度的完成时间之前并不起作用。

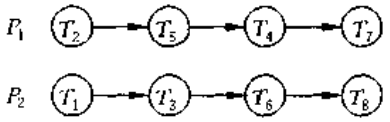


图 10.8 一个调度的表表示

10.3.2 起始群体

在遗传算法的每一次迭代中，必须维持一个符号串的群体。起始的符号串群体(MSP的调度)通常是随机生成的。群体的大小一般取决于问题本身，并需要在实验中确定。为了方便调度的生成和遗传操作的构造，引入了如下高度排序条件：在调度中每个处理器上的任务都按高度升序排列于表中。例如，在图 10.8 的处理器 P_1 中，$height(T_2) \leqslant height(T_5) \leqslant height(T_4) \leqslant height(T_7)$。可以证明一个满足高度排序条件的调度是一个有效调度，亦即先后关系不会被破坏[87]。

以图 10.1 中的任务图为例。任务 T_3(高度为 1)是 T_8(高度为 3)的一个前辈。如果它们两个都被分配给同一个处理器，则 T_3 依照高度排序在 T_8 的前面，从而保证了 T_3 在 T_8 之前在该处理器上得到执行。不过，如果在两个任务之间不存在先后关系，则不必采用高度排序。例如，任务 T_6(高度为 2)与 T_5(高度为 1)无关，所以它们可在同一处理器中以任意顺序执行。

高度排序条件只是一个必要条件，最优调度有可能并不满足这一条件。高度的定义可被修正以减小这种情况出现的可能性。任务 T_j 的新高度被定义为一个在 $1+\max height(T_i)$ 与 $-1+\max height(T_k)$ 之间的随机整数，其中 $T_i \in PRED(T_j)$，$T_k \in SUCC(T_j)$。在这一章的余下部分，$height$ 遵从新的定义。

对于一个具有 p 个处理器的多处理器系统，下面是一个可以生成满足高度排序条件的随机调度的算法。

调度生成算法 为一个具有 p 个处理器的多处理器系统生成满足高度排序条件的随机调度的算法。

(1) 计算 TG 中每个任务的高度 $height$。

(2) 通过将 TG 中的任务划分为不同的子集 $G(h)$，即高度为 h 的任务集，把任务按高度分开。

(3) 对前 $p-1$ 个处理器中的每一个处理器，进行以下操作：

① 对每一个子集 $G(h)$，置 $NG(h)$为 $G(h)$中的任务个数。

② 随机生成一个在 0 和 $NG(h)$之间的数字 r。

③ 从 $G(h)$中选取并移出 r 个任务，分配给当前处理器。

(4) 将所有子集中剩下的任务分配给最后一个处理器。

搜索节点的起始群体可由重复执行以上调度生成算法而得到。

10.3.3 适度函数

适度函数被用来评价搜索节点并控制遗传操作。对于 MSP，适度函数需要考虑吞吐量、完成时间以及处理器的使用等因素。在这里所介绍的遗传算法中，适度函数仅基于调度的完成时间。一个调度 S 的完成时间定义为任务图中所有任务执行完成所用的时间，并以 $FT(S)$表示。

因为我们所要求的是将完成时间最小化，而遗传操作之一(繁殖)总是试图将适度函数最大化，调度的适度值可被定义为 $C_{max}-FT(S)$，其中 C_{max}是在上一次迭代中获得的最大完成时间。因此，最优调度会有最小的完成时间和比其它调度大的适度值。

10.3.4 遗传操作

遗传操作的功能之一是在当前搜索节点群体基础上生成新的搜索节点。新的节点通过合并或重组旧的节点而构造出来的。就像在遗传学中一样，适当选择一个搜索节点的符号串表示，其特定结构可以表示该搜索节点的"优度"。因此，合并两个搜索节点的好的结构可能得到一个更好的搜索节点。将这一概念应用于 MSP，一个调度的特定部分可能属于最优调度。组合这些"最优"部分，我们也许会有效地发现最优调度。

对于 MSP，遗传操作必须确保调度中处理器内先后关系和任务的完整性和唯一性，正如 10.3.1 节讨论过的那样。这将保证新生成的符号串是表示有效搜索节点。下面我们介绍一个基于交叉概念的 MSP 遗传操作[64]。

1. 交叉

考虑图 10.1 所示任务图 TG。图 10.9 所示的两个符号串都是这一任务图对于两处理器的有效调度。新的符号串可由如下交换两个符号串的部分结构而得。

(1) 选择将表切断为两段的截点(交叉截点)，见图 10.9。

(2) 交换 P_1 的符号串 A 和符号串 B 的尾部。

(3) 交换 P_2 的符号串 A 和符号串 B 的尾部。

新生成的符号串如图 10.10 所示。注意，新的符号串之一 D 的完成时间短于原有的两个符号串。事实上，这是任务图 TG 在两处理器情况下的最优完成时间。上述操作可以很容易地扩展到 p 处理器的情况下并且十分有效。不过，我们还需要确定选取交叉截点的方法，并证明其生成的新符号串是有效的。

毫无疑问，被生成截点的有效性极大地取决于交叉截点的选择。注意在上面例子中交叉截点总是介于两个不同高度的任务之间($height(T_5)\neq height(T_8)$，$height(T_4)\neq height(T_6)$，等等)。这实际上是基于下面定理[87]。

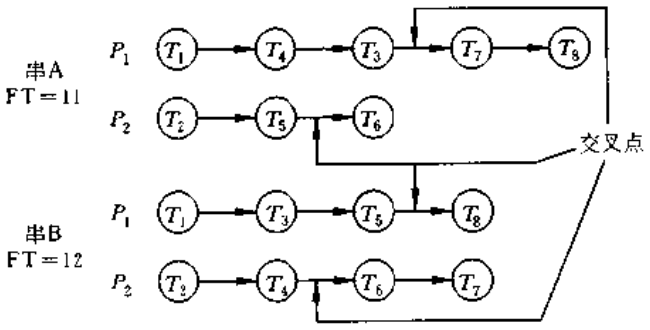


图 10.9　交叉操作的两个符号串

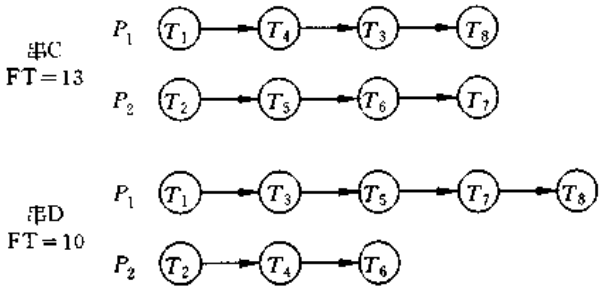


图 10.10　交叉操作生成的两个新符号串

定理 1　如果交叉截点的选取使得

(1) 每个交叉截点两侧的任务的高度都是不同的。

(2) 所有交叉截点前面最近任务的高度都是一样的。

则新生成的符号串一定有效。

交叉操作利用以上特性选择交叉截点以使新的符号串总为有效调度。这一过程被归纳为如下算法。

交叉算法　这一算法在两个符号串 A 和 B 上施用交叉操作并生成两个新符号串。

(1) 在 0 与任务图的最大高度之间随机生成数值 c。

(2) 选取交叉截点。

① 对符号串 A 和符号串 B 中的每个处理器 P_i，找出其中高度为 c 的最后一个任务 T_j，且令 T_k 为紧跟其后的任务，亦即 $c=height(T_j)<height(T_k)$，且对所有 i，$height(T_{j'})$ 均相同。

② 对符号串 A 和符号串 B 中的每个处理器 P_i，从交叉截点处交换处理器 P_i 中符号串 A 和符号串 B 的底半部。

虽然交叉操作十分有效，它在本质上是随机的并可能抹去最优解答。在一般情况下，它的使用要由交叉概率来控制，而交叉概率的值又通常是由实验中得到的。再进一步，我们总可以在下一代中保留已经发现的最优解答。

2. 繁殖

繁殖操作通过从旧的群体中选取适度值最大的符号串来构成新的群体。选取准则为：

具有较高适度值的符号串应有较高的机会在下一代中存活。这是因为"好的"符号串具有较高的适度值并应被保留到下一代。一般来说,一个偏置轮盘可用来实现繁殖。在这个轮盘上,群体中的每个符号串都占有面积与其适度值大小成正比的一席之地。随机生成的数值被用作轮盘上的指针以决定哪一个符号串会存活到下一代。因为适度值大的符号串占有面积较大,它们会有更大可能被选中并延续到下一代(见 6.2.1 节)。

对基本的繁殖操作稍加改动,即可将当前群体中最好的符号串繁衍到下一代中去。这一改动可以提高遗传算法的性能。下面算法归纳了繁殖操作。

繁殖算法 这一算法在符号串群体 *POP* 上施用繁殖操作并生成一个新的符号串群体 *NEWPOP*。令 *NPOP* 为 *POP* 中的符号串个数。

(1) 构造一个轮盘。令 *NSUM* 为 *POP* 中符号串适度值的总和;将轮盘划分为 *NSUM* 个格,然后根据适度值的大小将相应数目的格分配给符号串。

(2) 重复下面步骤 *NPOP*-1 次:生成一个在 1 和 *NSUM* 之间的随机数,并将它指到的格所对应的染色体加入 *NEWPOP*。

(3) 将 *POP* 中具有最大适度值的符号串加入 *NEWPOP*。

3. 突变

突变可视为以小概率发生的符号串值的随机转移。它也可被看作逃脱不成熟收敛的方式。对于 MSP,突变由随机交换两个高度相同的任务来实现。突变操作可归纳为如下算法。

突变算法 这一算法在一个符号串上施用突变操作以生成一个新的符号串。

(1) 随机选取一个任务 T_i。

(2) 比较高度。找出符号串中与之高度相同的任务 T_j。

(3) 交换任务。通过在调度中交换任务 T_i 与 T_j 来生成一个新的符号串。

一般来讲,施用突变操作的频率由突变概率控制,而突变概率的值通常又是由实验得到。

10.3.5 完整的算法

现在让我们来把上述单个算法集中到一起,构成 MSP 的完整遗传算法。

调度搜索算法 这一算法用于求解 MSP。

(1) 调用调度生成算法 *N* 次并将生成的符号串存入 *POP*。

(2) 重复下面步骤直至算法收敛:

① 计算 *POP* 中每个符号串的适度值。

② 调用繁殖算法。令 *BESTSTRING* 为 *POP* 中适度值最大的符号串。

③ 从 $i=1$ 到 $NPOP/2$,进行

- 从 *POP* 中取出两个符号串并以概率 p_c 调用交叉操作。
- 如果执行交叉操作,则将新生成的符号串加入 *TMP*;否则,将原符号串加入 *TMP*。

④ 对 *TMP* 中的每个符号串，以概率 p_m 调用突变算法。如果执行突变操作，则将新生成的符号串加入 *POP*；否则，将原符号串加入 *POP*。

⑤ 用 *BESTSTRING* 取代 *POP* 中适度值最小的符号串。

算法在满足收敛准则时终止，亦即经过连续数代群体中的最好解答不再变化。

10.3.6 仿真结果

我们对上节中讨论的遗传算法用随机选择的任务图实现并测试。这些任务图的最优调度都是已知的。随机生成的调度具有 20～90 个任务。每个任务的子节点个数是在 1 和 4 之间的随机整数，且每个任务的运行时间为 1 和 50 之间的随机整数。这些任务图同时也用列表调度算法进行了测试[91]。随机生成的任务图不是显式构造的，但在它们的构造过程中，可获取最优调度。仿真采用了下列参数：

- 群体大小＝10
- 交叉概率＝1.0
- 突变概率＝0.05
- 最大迭代次数＝1500

表 10.1～10.4 比较了在不同处理器个数情况下遗传算法和列表算法得到的完成时间以及随机任务图的最优调度。仿真是在一台 SUN4/490 工作站上完成的，且程序运行时间通常只有一两秒钟。在所有情况下，遗传算法在小于 1000 代之内收敛于一个解答。从表 10.1～10.4 可见，利用遗传算法得到的解答总是优于列表算法，且与最优解的误差不超过 10%。

表 10.1 两处理器情况下最优调度、遗传算法和列表算法用于各种任务图的比较

任务个数	最优调度(OPT)	遗传算法(GA)	列表算法(LA)	$\frac{GA-OPT}{OPT}\%$	$\frac{LA-OPT}{OPT}\%$
30	392	395	416	0.77	6.12
35	410	436	457	6.34	11.46
41	490	508	522	3.67	6.53
51	653	662	674	1.38	3.22
61	768	783	822	1.95	7.03

表 10.2 三处理器情况下最优调度、遗传算法和列表算法用于各种任务图的比较

任务个数	最优调度(OPT)	遗传算法(GA)	列表算法(LA)	$\frac{GA-OPT}{OPT}\%$	$\frac{LA-OPT}{OPT}\%$
31	260	266	280	2.31	7.69
36	295	305	366	3.39	24.07
42	352	378	393	7.39	11.65
53	434	451	454	3.92	4.61
68	561	584	608	4.1	8.38

表 10.3 四处理器情况下最优调度、遗传算法和列表算法用于各种任务图的比较

任务个数	最优调度(OPT)	遗传算法(GA)	列表算法(LA)	$\frac{GA-OPT}{OPT}\%$	$\frac{LA-OPT}{OPT}\%$
28	190	198	237	4.21	24.74
41	267	285	291	6.74	8.99
57	372	385	400	3.49	34.14
64	394	434	484	10.15	22.84
75	458	467	511	1.97	11.57

表 10.4 五处理器情况下最优调度、遗传算法和列表算法用于各种任务图的比较

任务个数	最优调度(OPT)	遗传算法(GA)	列表算法(LA)	$\frac{GA-OPT}{OPT}\%$	$\frac{LA-OPT}{OPT}\%$
29	147	153	186	4.08	26.53
42	220	232	268	5.45	21.82
56	280	305	329	8.93	17.50
67	346	357	363	3.18	4.91
77	383	407	421	6.27	9.92

10.4 习题

10.1 证明定理 10.1。

10.2 假定去掉高度排序条件且交叉操作与 6.1.2 节所给出的类似，

(1) 提出一个用于检查新生成调度的有效性的程序；

(2) 设计一个在调度无效时可以修正它的程序。

10.3 设计一个新的 MSP 交叉操作，使之总会生成有效调度。

第11章 作业调度

作业调度问题(JSP)是一个资源分配问题，只不过这里的资源被称作设备而已。这一问题的求解目标是要找到一个将一组作业安排到设备上去以使作业可被"最优"完成的方案。每个作业可由一些任务组成，而每个任务必须由特定的设备处理。一个调度是按先后顺序条件将所有任务安排到设备上的一种方案。通常，约束的数目很大，使得 JSP 成为一个非常难解的组合问题(NP 完全问题)[99]:57]。物流作业问题(FSP)则是具有更严格条件的 JSP 的特例，并可被约化为旅行商问题(TSP)[137]。

作业调度通常被称为 n/m JSP，其中 n 是作业个数，m 是设备个数。图 11.1 是一个 5/3问题。图中每一个格是一个操作，每一横行是一个作业。每个操作都有三个下标 i，j 和 k，其中 i 是作业序号，j 是操作序号，而 k 则是执行这一操作的设备的序号。每个格的长度表示完成该操作所需的时间。

作业 1	1,1,2	1,2,3	1,3,3	1,4,2	1,5,2
作业 2	2,1,1	2,2,2	2,3,1	2,4,2	
作业 3	3,1,1	3,2,3			
作业 4	4,1,1	4,2,2	4,3,1	4,4,1	4,5,3
作业 5	5,1,3	5,2,2	5,3,3		

图 11.1 一个 5/3 作业调度问题

在寻找上述问题最优调度的过程中，调度是在满足先后关系约束的前提下，对这些格子在每个设备上作出的安排。将有效的调度按照它们目标函数的相应得分排队，即可得到得分最高的调度为最优调度。图 11.2 示出了在目标函数为最短完成时间情况下前述 5/3 问题的最优调度。

设备 1	4,1,1	3,1,1	2,1,1	4,3,1	2,3,1	4,4,1	
设备 2	1,1,2	4,2,2	5,2,2	2,2,2	1,4,2	2,4,2	1,5,2
设备 3	5,1,3	1,2,3		1,3,3	3,2,3	5,3,3	4,5,3

图 11.2 图 11.1 中 5/3 问题的最优调度

已经有许多用于求解作业调度问题的最优化方法被提出，包括神经元网络[180]和拉格朗日松弛法[81][82]。不过，由于问题本身的难度很大，多数现有最优化算法都只适用于规模较小的问题。另一方面，许多工业界依靠计算机模拟生成可行调度，而这样得到的可行调度不一定保证好的性能。更进一步，评价调度算法的性能是一件很困难的事情，因为获取

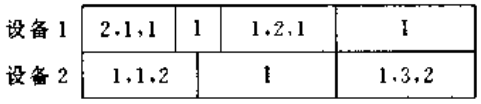
规模很大且最优调度已知的测试问题本身就很困难。

遗传算法(GA)在作业调度问题上的应用,包括物流作业调度问题,是最近才发展起来的研究方向(参见文献[31,43,49,88,103,112]),且研究的对象多为 FSP 或较小规模的 JSP。

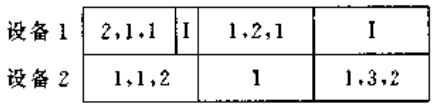
11.1 调度的分类

由于闲置时间可以无限多种方式插入任意给定调度,任意一个作业调度问题都可以有无数个调度。不过,具有闲置时间的调度并不在我们的兴趣范围之内,因为它们通常不能将目标函数最优化。在无限个数的调度中,特定的调度可视为主导调度。它们的定义为

• 半主动调度　在任意调度中将每个操作在不破坏操作顺序的条件下尽可能地向左移,即可获得半主动调度。图 11.3 所示就是一个 2/2 问题的半主动调度(图中 I 表示设备空闲)。



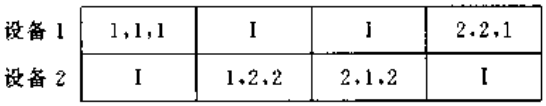
调度 1



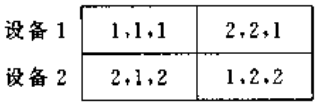
调度 2

图 11.3　调度 2 是从调度 1 得到的半主动调度

• 主动调度　如果一个操作的左侧有一个足够放下它的空位,它可以跳过其它操作而占据这一闲置时间。在一个半空调度中将每个操作在满足先后顺序约束的前提下尽可能地向左跳,即可获得主动调度。图 11.4 中的主动调度(调度 2)就是半主动调度(调度 1)中操作(2,1,2)跳过操作(1,2,2)所得。显然,最优调度一定是主动调度。



调度 1



调度 2

图 11.4　调度 2 是从调度 1 得到的主动调度

• 可安排操作　如果同一作业中排在一个操作前面的其它操作的总完成时间小于或等于 t,则称此操作在时间 t 设备 j 处可安排。如图 11.1 和 11.2 中的设备 1,在操作

(4,1,1)之后,操作(3,1,1)和(2,1,1)均对设备 1 可安排。

· 无延迟调度　在一个主动调度中,如果对每个闲置时间 t 和每个设备 j 都没有在时间 t 设备 j 处的可安排操作,则称此调度为无延迟调度。简而言之,没有在处理它的设备存在且闲置的情况下被延迟的作业。注意,无延迟调度是主动调度的一个子集,但是一个最优调度并不一定是无延迟调度。以图 11.5 给出的 2/2 问题为例。调度 1 不是一个无延迟调度,因为设备 2 在操作(2,1,2)可安排时处于闲置状态。相反,调度 2 则为一个无延迟调度,且调度 2 比调度 1 先完成。

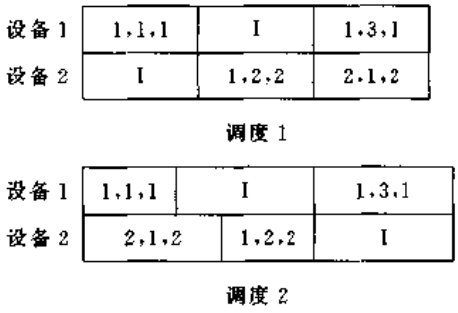


图 11.5　延迟的和无延迟的调度

在这一章中,只考虑主动无延迟调度的生成,以找到具有最短完成时间的调度。

11.2　JSP 的遗传算法

应用 GA 求解 JSP 与用其求解 MSP 类似。将 GA 应用于 JSP 中的关键是采用有效的编码方式以及适当的交叉、突变操作。在这一节中,我们介绍一些将 n/m 作业调度问题转换到 m 排列的新的编码方式。在这一编码方式中,会用到 PMX(部分匹配交叉),OX(有序交叉)以及 CX(循环交叉)等重新排序交叉操作(参见 6.4 节)。重新排序交叉操作适用于排列问题,因为它们在交叉后所生成的符号串一定还是有效的。

11.2.1　JSP 的编码方式

排列可表示为整数符号串。可利用一类从旧的排列产生新的有效排序的交叉操作,即重新排序交叉操作。这一技术已被成功地应用于求解那些可用排列表示的问题,诸如 TSP。利用 GA 求解 JSP 的基本概念是扩展这一方法,将调度表示为一组排列,其中每个排列对应于一个设备。重新排序操作被同时应用于所有排列并总是生成有效调度。

调度被表示为多个排列。在每个设备上都必须有特定数目的操作被执行。容量约束要求在同一时刻这些操作只有一个被执行。因此,每个操作的排列表示了一个设备上的操作处理序列,且所有设备上排列的组合表示一个可能的调度。

例如,图 11.1 给出了一个 5/3 JSP 的约束条件。如果每个设备上的操作序列都可表示如下:

设备 1	2,1,1	2,3,1	3,1,1	4,1,1	4,3,1	4,4,1	
	a_{11}	a_{12}	a_{13}	a_{14}	a_{15}	a_{16}	
设备 2	1,1,2	1,4,2	1,5,2	2,2,2	2,4,2	4,2,2	5,2,2
	a_{21}	a_{22}	a_{23}	a_{24}	a_{25}	a_{26}	a_{27}
设备 3	1,2,3	1,3,3	3,2,3	4,5,3	5,1,3	5,3,3	
	a_{31}	a_{32}	a_{33}	a_{34}	a_{35}	a_{36}	

则排列

设备 1 $[a_{14},a_{13},a_{11},a_{15},a_{12},a_{16}]$

设备 2 $[a_{21},a_{26},a_{27},a_{24},a_{22},a_{25},a_{23}]$

设备 3 $[a_{35},a_{31},a_{32},a_{33},a_{36},a_{34}]$

可被用来表示图 11.2 所示调度。

显然，并非所有排列都可以代表有效调度。例如，在设备 1 上，排列$[a_{14},a_{13},a_{12},a_{15},a_{11},a_{16}]$就不是一个有效调度，因为 a_{12}所表示的操作(2,3,1)不能在 a_{11}所表示的操作(2,1,1)之前进行。

11.2.2 调度的生成

对于多数实际目标函数来说，无延迟调度是一类非常重要的调度。生成无延迟调度也比生成任意的主动调度容易些。虽然无延迟调度不一定是最优的，但大量实验结果表明在多数情况下它们比其它主动调度要好。

可安排操作集是调度生成中的一个有用概念，并被简称为可安排子集。在任一时刻，可安排子集 S_a是那些在其前面的操作已被排好的所有操作。对于一个 n 作业 m 设备问题，S_a最初包括每个作业的第一个操作。一段时间之后，S_a的子集 S 被选中并移入调度。这时，如果存在的话，紧随 S 中每个操作之后的那些操作被加入 S_a中。这一过程重复进行，直到 S_a成为空集，调度完成。注意，当没有操作要在设备 j 上执行时，设备 j 闲置。设备 j 的可安排子集可定义为分配给设备 j 的操作中那些其前面的操作已被排好的操作集。利用可安排子集的概念，可以很容易地得到一个生成无延迟调度的程序。图 11.6 给出了调度程序的 C 语言伪编码。

```
调度
{
    初始化;
    while(not 所有作业都已完成)
    {
        时间推进;
        /* 检查每个设备的状态 */
        for(每个设备)
        {
            if(设备闲置或设备上当前操作已经完成)
                将此设备标为可用;
        }
```

```
        /* 在可用设备上排入新的操作 */
        for(每个设备)
        {
            if(这一设备已被标为可用)
            {
                计算这一设备的可安排子集;
                if(可安排子集非空)
                {
                    根据预先定义的选取准则从可安排子集中选出一个新的操作;
                    将新选定的操作分配给该设备;
                    将该设备标为不可用;
                }
            }
        }
    }
}
```

图 11.6 生成主动无延迟调度的程序

多种选取准则可用来构成不同的调度程序。调度的质量取决于选取准则从每个设备的可排子集中挑选操作的"智能"。下面列出了 7 个常用的选取准则[44]。它们被用于与 GA 比较。

(1) RANDOM：从每个设备的可安排子集中随机选取一个操作。

(2) MOPNR：从每个设备的可安排子集中选取一个操作，且该操作属于具有最大数目未完成操作的作业。

(3) MWKR-P：从每个设备的可安排子集中选取一个操作，且该操作对应于一个可安排操作之后的总处理时间最大的作业。

(4) MWKR/P：从每个设备的可安排子集中选取一个操作，且该操作对应于一个未安排操作的总处理时间与可安排操作本身的处理时间之比率最大的作业。

(5) SPT：从每个设备的可安排子集中选取一个处理时间最小的操作。

(6) MWKR：从每个设备的可安排子集中选取一个操作，且该操作对应于一个未安排操作的总处理时间最大的作业。

(7) LWKR：从每个设备的可安排子集中选取一个操作，且该操作对应于一个未安排操作的总处理时间最小的作业。

正如上节所述，表示调度的排列用作决定哪一个操作应先处理的优先次序。例如，设备 1 有一个排列$[a_{14},a_{13},a_{11},a_{15},a_{12},a_{16}]$，且此时设备 1 可用，并有可安排操作子集$(a_{11},a_{13},a_{12})$，则操作 a_{13}应选为下一个操作。

显而易见，排列与调度之间的映射关系不是唯一的，但这并不妨碍我们运用排列作为 GA 的编码方式。

11.2.3 遗传操作

本节讨论交叉和突变操作。突变操作相对简单一些，即随机选取一个设备及其排列中的两个元素，并交换这两个元素。不过，交叉操作就要稍微复杂一些。

因为一个 n/m JSP 调度是由 m 个排列表示的，对常用的 PMX，OX 和 CX 等重新排序操作（见 6.4 节）稍加改动即可加以使用。首先，重新排序操作必须作用于具有同样长度的排列。可是每个设备一般来说会有不同数目的操作。这使得交叉操作只能在同一设备的排列上使用。第二，这些重新排序操作的共同点在于它们对交叉截点的选择和对先后顺序的保护。因为设备并行运作，有必要使所有相关设备上的交叉截点相互关联。这可以通过在每个设备上选取起始和终止时间近乎相同的交叉截点来实现。这样，在这一时段上的先后顺序在交叉后仍可保持。

下面例子给出了上述交叉操作的示范。图 11.1 所示 5/3 JSP 的两个符号串被选来进行交叉。

符号串 A

设备 1 $[a_{14},a_{13},a_{11},a_{15},a_{12},a_{16}]$

设备 2 $[a_{21},a_{26},a_{27},a_{24},a_{22},a_{25},a_{23}]$

设备 3 $[a_{35},a_{31},a_{32},a_{33},a_{36},a_{34}]$

符号串 B

设备 1 $[a_{11},a_{14},a_{13},a_{15},a_{16},a_{12}]$

设备 2 $[a_{26},a_{27},a_{23},a_{24},a_{22},a_{25},a_{21}]$

设备 3 $[a_{35},a_{31},a_{33},a_{32},a_{34},a_{36}]$

假定对于符号串 A 设备 1 的排列随机选择了位置 2 和 4 作为两个交叉截点，则处于位置 2 的操作 a_{13}的起始时间和处于位置 4 的操作 a_{15}的终止时间可从调度中发现。对于符号串 A 的设备 2，两个相应的操作可据此找到，以使得第一个操作的起始时间非常接近于 a_{13}的起始时间，且第二个操作的终止时间非常接近于 a_{15}的终止时间。这两个操作将用作设备 2 的交叉截点。与此类似地，可找到设备 3 的交叉截点。若用符号“‖”来表示交叉截点，则有

符号串 A

设备 1 $[a_{14},\|a_{13},a_{11},a_{15},\|a_{12},a_{16}]$

设备 2 $[\|a_{21},a_{26},a_{27},\|a_{24},a_{22},a_{25},a_{23}]$

设备 3 $[a_{35},a_{31},\|a_{32},a_{33},\|a_{36},a_{34}]$

在上面交叉截点中，设备 1 的操作 a_{13}与设备 2 的操作 a_{21}和设备 3 的操作 a_{32}具有基本相同的起始时间；设备 1 的操作 a_{15}与设备 2 的操作 a_{27}和设备 3 的操作 a_{33}具有基本相同的终止时间。在确定交叉截点之后，重新排序操作施用于每个设备的排列上并生成两个相对排列。例如，在上述交叉截点处施用 OX 之后，设备 1 和设备 2 的新的调度为

符号串 A'

设备 1 $[a_{11},a_{14},a_{13},a_{15},a_{12},a_{16}]$

设备 2 $[a_{21},a_{26},a_{27},a_{24},a_{22},a_{25},a_{23}]$

设备 3 $[a_{35},a_{31},a_{32},a_{33},a_{36},a_{34}]$

符号串 B'

设备 1 $[a_{14},a_{13},a_{11},a_{15},a_{16},a_{12}]$

设备 2 $[a_{21},a_{26},a_{27},a_{24},a_{22},a_{25},a_{23}]$

设备 3 $[a_{35},a_{31},a_{32},a_{33},a_{34},a_{36}]$

11.3 仿真结果

我们对上面一节讨论的 GA 用不同规模的 JSP 进行了仿真，并将其结果与传统调度程序 RANDOM，MOPNR，MWKR-P，MWKR/P，SPT，以及 LWKR 等进行了比较。

表 11.1～11.2 是 GA 与传统调度程序的完成时间的对比。这里所用的交叉概率为 0.8，突变概率为 0.2，且群体大小为 80×(作业个数)。表 11.1 给出的是对于 8 个不同的 10/10 问题 GA 在 1000 代以上所得的最好调度与传统程序所得调度的比较。表 11.2 给出的是对于 8 个不同的 100/50 问题 GA 在 1000 代以上所得的最好调度与传统程序所得调度的比较。即使对于实际问题来说，100/50 问题的规模也是足够大的了。

表 11.1 10/10 问题的完成时间

	GA	RANDOM	MOPNR	MWKR-P	MWKR/P	SPT	MWKR	LWKR
1	64	66	69	83	88	70	67	86
2	71	76	80	84	73	79	83	89
3	64	66	74	74	70	74	75	91
4	73	77	83	87	86	81	84	99
5	83	86	97	98	114	113	105	114
6	83	83	85	96	96	98	84	100
7	89	97	99	114	123	120	105	134
8	87	90	102	116	116	119	97	102

表 11.2 100/50 问题的完成时间

	GA	RANDOM	MOPNR	MWKR-P	MWKR/P	SPT	MWKR	LWKR
1	549	574	537	614	727	607	541	718
2	550	579	557	672	691	616	558	708
3	563	591	565	604	663	625	555	704
4	548	567	539	640	685	603	551	669
5	547	576	553	617	679	652	549	694
6	543	572	544	650	664	603	541	702

续表

	GA	RANDOM	MOPNR	MWKR-P	MWKR/P	SPT	MWKR	LWKR
7	552	574	554	610	661	606	562	734
8	553	587	555	622	692	641	566	712

从这些比较可以看出，GA 所得结果相当不错。对于 10/10 问题，GA 的结果总是最好的；对于 100/50 问题，GA 的结果也在最好之列。相对于所有其它程序的平均水平，GA 总体上将 10/10 问题的解答改进了 17%，并将 100/50 问题改进了 14%。同时可以看到，传统程序的表现随测试问题不同有很大起伏，而 GA 在所有测试问题上的表现则相对稳定。

至于运行时间，GA 比多数传统程序的计算时间要长。这是因为 GA 是一个迭代算法，而其它算法则为一次性的。不过，GA 在本质上是并行的，并可在并行处理情况下获得很高效率。

11.4 习题

11.1 利用 11.2.3 节介绍的遗传算法和 OX 操作进行仿真，求解 JSP。

11.2 利用遗传算法和 PMX 操作进行仿真，求解 JSP。

11.3 利用遗传算法和 CX 操作进行仿真，求解 JSP。

11.4 比较 OX，PMX 和 CX 求解 JSP 的性能。

参 考 文 献

[1] Aazhang, B., B. P. Paris and G. C. Orsak, "Neural networks for multiuser detection in code-division-multiple-access communications," IEEE Trans. on Communications, vol. 40, no. 7, July 1992, pp. 1212—1222.

[2] Aarts, E. and J. Korst, Simulated Annealing and Boltzmann Machines. New York: Wiley & Sons, 1989.

[3] Aarts, E. H. L. and J. H. M. Korst, "Boltzmann machines for travelling salesman problems," European Journal of Operational Research, vol. 39, Issue 1, March 6, 1989, pp. 79—95.

[4] Aarts, E. and P. J. M. van Laarhoven, "A new polynomial time cooling schedule," Proc. IEEE Int. Conf. on Computer-Aided Design, Santa Clara, CA, November 1985, pp. 206—208.

[5] Abramson, D., "Constructing school timetables using simulated annealing sequential and parallel algorithms," Management Science, vol. 37, no. 1, January 1991, pp. 98—103.

[6] Abrishamkar, F. and Z. Siveski, "PCS global mobile satellites," IEEE Communications Magazine, vol. 34, no. 9, September 1996.

[7] T. L. Adams et al., "A comparison of list schedules for parallel processing systems," Communications of ACM, vol. 17, December 1974, pp. 685—690.

[8] Abu-Mostafa, Y. S. and J. St. Jacques, "Information capacity of the Hopfield model," IEEE Trans. on Information Theory, vol. 31, July 1985, pp. 461—464.

[9] Agrawal, A., N. Ansari and E. S. H. Hou, "Evolutionary programming for fast and robust point pattern matching," Proc. 1994 IEEE Conference on Neural Networks, Orlando, FL, June 28—July 2, 1994, pp. 1777—1782.

[10] Aho, A. V., J. E. Hopcroft and J. D. Ullman, The Design and Analysis of Computer Algorithms. New York: Addison Wesley, 1974.

[11] Aiyer, S. V. B., M. Niranjan and F. Fallside, "A theoretical investigation into the performance of the Hopfield model," IEEE Trans. on Neural Networks, vol. 1, June 1990, pp. 204—215.

[12] Akiyama, Y., A. Yamashita, M. Kajiura and H. Aiso, "Combinatorial optimization with Gaussian machines," Proc. International Joint Conference on Neural Networks, Washington, D. C, 1989, pp. I: 533—540.

[13] Anderson, J. A. and E. Rosenfeld, (eds.), Neurocomputing: Foundations of Research. Cambridge, MA: MIT Press, 1988.

[14] Anisimov, V. Y., "Parameter estimation in the case of fuzzy information on the observation conditions," Telecommunications and Radio Engineering, vol. 44, no. 5, May 1989, pp. 86—88.

[15] Ansari, N., A. Arulambalam and S. Balasekar, "Traffic management of a satellite communication network using stochastic optimization," IEEE Trans. on Neural Networks, vol. 7, May 1996, pp. 732—744.

[16] Ansari, N., M. Chen and E. S. H. Hou, "CHAPTER 13: A genetic algorithm for point pattern matching," in Dynamic, Genetic and Chaotic Programming—The Sixth-Generation (B. Soucek

ed.),pp. 353—371,New York: Wiley & Sons,1992.

[17] Ansari,N. and E. J. Delp,"Partial shape recognition: a landmarkbased approach," IEEE Trans. on Pattern Analysis and Machine Intelligence,vol. 12,no. 5,May 1990,pp. 470—483.

[18] Ansari,N. ,E. S. H. Hou and A. Agrawal,"Point pattern matching by simulated annealing,"Proc. 1993 IEEE Regional Conference on Control Systems,Newark,NJ,August 13—14,1993,pp. 215—218.

[19] Ansari,N. ,E. S. H. Hou and Y. Yu,"A new method to optimize the satellite broadcasting schedules using the mean field annealing of a Hopfield neural network,"IEEE Trans. on Neural Networks,vol. 6,no. 2,March 1995,pp. 470—483.

[20] Ansari,N. and K. Li,"Landmark-based shape recognition by a modified Hopfield neural network," Pattern Recognition,vol. 26,no. 4,April 1993,pp. 531—542.

[21] Ansari,N. and X. Liu,"Recognizing partially occluded objects by a bidirectional associative memory," Optical Engineering,vol. 32,no. 7,July 1993,pp. 1539—1548.

[22] Ansari,N. ,R. Sarasa and G. Wang,"An efficient annealing algorithm for global optimization in Boltzmann machines," Applied Intelligence,vol. 3,1993,pp. 177—192.

[23] Banerjee,P. ,M. H. Jones and J. S. Sargent,"Parallel simulated annealing algorithms for cell placement on hypercube multiprocessors,"IEEE Trans. on Parallel and Distributed Systems,vol. 1,no. 1,January 1990,pp. 91—106.

[24] Baram,Y. ,"On the capacity of ternary Hopfield networks,"IEEE Trans. on Information Theory, vol. 37,May 1991,pp. 528—534.

[25] Baram,Y. ,"Encoding unique global minima in nested neural networks,"IEEE Trans. on Information Theory,vol. 37,July 1991,pp. 1158—1162.

[26] Barr,A. and E. A. Feigenbaum,The Handbook of Artificial Intelligence,vol. 1. Los Altos,CA: Kaufmann,1981.

[27] Bartle,R. G. ,The Elements of Real Analysis,2nd Edition. New York: Wiley & Sons,1976.

[28] Basso,A. and M. Kunt,"Autoassociative neural networks for image compression," European Trans. on Telecommunications,vol. 3,no. 6,November 1992,pp. 593—598.

[29] Battiti,R. and G. Tecchiolli,"Simulated annealing and tabu search in the long run: a comparison on QAP tasks," Computers and Mathematics with Applications,vol. 28,no. 6,1994,pp. 1—8.

[30] Belew,R. K. and L. B. Booker,(eds.),Proc. of the Fourth International Conference on Genetic Algorithms. San Mateo,CA: Morgan Kaufmann Publishers,Inc. ,1991.

[31] Biegel,J. E. and J. J. Davern,"Genetic algorithms and job shop scheduling,"Computers industrial Engineering,vol. 19,nos. 1—4,1990,pp. 81—91.

[32] Bilbro,G. L. ,W. E. Snyder and J. W. Gault,"Mean field annealing: a formalism for constructing GNC-like algorithms,"IEEE Trans. on Neural Networks,vol. 3,no. 1,1992,pp. 131—138.

[33] Boissin,N. and J. -L. Lutton,"A parallel simulated annealing algorithm,"Parallel Computing,vol. 19,no. 8,August 1993,pp. 859—872.

[34] Bourret,P. ,S. Goodall and M. Samuelides,"Optimal scheduling competitive activation: application to the satellite antennas scheduling problem,"Proc. IJCNN' 89, Washington, D. C. , 1989, pp. I565—572.

[35] Bourret,P. ,F. Rem and S. Goodall,"A special purpose neural network for scheduling satellite broadcasting times,"Proc. IJCNN' 90,Washington,D. C. ,1990,pp. II535—538.

[36] Chao, D. Y. and D. T. Wang, "Enhancement of memory capacity of neural networks," Proc. 1992 IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems, Raleigh, NC, July 7—10, 1992, pp. 519—526.

[37] Chang, C. and C. Wu, "Optimal frame pattern design of a TDMA mobile communication system using a simulated annealing algorithm," IEEE Trans. on Vehicular Technology, vol. 42, May 1993, pp. 205—211.

[38] Chang, P. -R. , and B. -C. Wang, "Adaptive decision feedback equalization for digital satellite channels using multilayer neural networks," IEEE Journal on Selected Areas in Communications, vol. 13, no. 2, February 1995, pp. 316—324.

[39] Chen, C. L. , C. S. G. Lee, and E. S. H. Hou, "Efficient scheduling algorithms for robot inverse dynamics computation on a multiprocessor system," IEEE Trans. on Systems, Man and Cybernetics, vol. 18, no. 5, December 1988, pp. 729—743.

[40] Chen, S. , B. Mulgrew and P. M. Grant, "A clustering technique for digital communications channel equalization using radial basis function networks," IEEE Trans. on Neural Networks, vol. 4, no. 4, July 1993, pp. 570—579.

[41] Chen, X. and I. M. Leslie, "Neural adaptive congestion control for broadband ATM networks," IEE Proc. I, Communications, Speech, and Vision, vol. 139, no. 3, pp. 233—240.

[42] Chou, L. -D. and J. -L. C. Wu, "Parameter adjustment using neural-network-based genetic algorithms for guaranteed QOS in ATM networks," IEICE Trans. on Communications, vol. 78, no. 4, April 1995, pp. 572—579.

[43] Cleveland, G. A. and S. F. Smith, "Using genetic algorithms to schedule flow shop release," Proc. 3rd Int. Conf. on Genetic Algorithms and their Applications, Arling, VA, 1989, pp. 160—169.

[44] Conway, R. W. , W. L. Maxwell, and L. W. Miller, (eds.), Theory of Scheduling. Reading, MA: Addison-Wesley, 1967.

[45] Cook, J. , "The mean-field theory of a Q-state neural network model," Journal of Physics A, vol. 22, no. 12, June 21, 1989, pp. 2057—2068.

[46] Cook, S. A. , "The complexity of theorem-proving procedures," Proc. 3rd Ann. ACM Symp. on Theory of Computing, ACM, 1971, pp. 151—158.

[47] Cooper, P. G. , "Neural networks enable radio wave propagation modeling," Signal, vol. 47, no. 6, February 1993, pp. 29—31.

[48] Davenport Jr. , W. B. , Probability and Random Process. New York: McGraw-Hill, 1970.

[49] Davis, L. , "Job shop scheduling with genetic algorithms," Proc. of the First International Conference on Genetic Algorithms and Their Applications, Carnegie-Mellon University, Pittsburgh, PA, July 24—26, 1985, pp. 136—140.

[50] Davis, L. , "Applying adaptive algorithm to epistatic domains," Proc. of the International Joint Conference on Artificial Intelligence, 1985, pp. 162—164.

[51] Duque-Anton, M. , D. Kunz and B. Ruber, "Channel assignment for cellular radio using simulated annealing," IEEE Trans. on Vehicular Technology, vol. 42, no. 1, February 1993, pp. 14—21.

[52] Fischer, M. J. and T. C. Harris, "A model for evaluating the performance of an integrated circuit and packet-switched multiplex structure," IEEE Trans. on Communications, vol. 24, February 1976, pp. 195—202.

[53] Foo, Y. P. S. and Y. Takefuji, "Stochastic neural networks for solving job-shop scheduling," Proc.

IEEE IJCNN 88,1988,pp. 275—290.

[54] Foo,Y. P. S. and Y. Takefuji,"Integer-neural programming neural networks for job-shop scheduling,"Proc. IEEE IJCNN 88,1988,pp. 341-348.

[55] Friesz,T. ,H. J. Cho and J. N. Mehta,"Simulated annealing approach to the network disign problem with variational inequality constraints,"Transportation Science,vol. 26,no. 1,February 1992, pp. 18—26.

[56] Fu,Y. and P. W. Anderson,"Application of statistical mechanics to NP-complete problems in combinatorial optimization,"Journal of Physics A,vol. A19,1986,pp. 1605—1620.

[57] Garey,M. R. and D. S. Johnson,Computers and Intractability. New York: W. H. Freeman and Company,1979.

[58] Gelfand,S. B. ,Analysis of Simulated Annealing for Optimization,Massachusetts Institute of Technology,Ph. D. Dissertation,1987.

[59] Geman,S. and D. Geman,"Stochastic relaxation,Gibbs distribution and Baysian restoration in images,"IEEE Trans. on Pattern Analysis and Machine Intelligence,vol. 6,November 1984,pp. 721—741.

[60] Gelfand,S. B. and S. K. Mitter,"Weak convergence of Markov chain sampling methods and annealing algorithms to diffusion,"Journal of Optimization Theory and Applications,vol. 68,no. 3,March 1991,pp. 483—498.

[61] Gidas,B. ,"Nonstationary Markov chains and convergence of the annealing algorithm,"Journal of Statistical Physics,vol. 39,1985,pp. 73—131.

[62] Glauber,R. J. ,"Time-dependent statistics of the Ising model,"Journal of Mathematical Physics, vol. 4,1963,pp. 294—307.

[63] Goffe,W. L. ,G. D. Ferrier and J. Rogers,"Simulated annealing: an initial application in econometrics,"Computational Economics,vol. 5,no. 2,May 1992,pp. 133—146.

[64] Goldberg,D. E. ,Genetic Algorithms in Search,Optimization,and Machine Learning. New York: Addison Wesley,1989.

[65] Goldberg,D. E. ,and R. Lingle,"Alleles,loci,and the traveling salesman problem,"Proc. of an International Conference on Genetic Algorithms and Their Applications,1985,pp. 154—159.

[66] Gonzalez,M. J. ,"Deterministic processor scheduling,"Computing Surveys,vol. 9,no. 3,September 1977,pp. 173— 204.

[67] Grefenstette,J. J. ,(ed.),Genetic Algorithms and Their Applications: Proc. of the Second International Conference on Genetic Algorithms. Hillsdale,NJ: Lawrence Erlbaum Assoc. ,Publishers, 1987.

[68] Guo,H. ,M. Zuckermann and R. Harris,"A fast algorithm simulated annealing,"Physica Scripta, vol. 38,1991,pp. 40.

[69] Habib,I. W. ,A. A. Tarraf and T. N. Saadawi,"Intelligent traffic control for ATM broadband networks,"IEEE Communications Magazine,vol. 33,no. 10,October 1995,pp. 76—82.

[70] Hajek,B. ,"Cooling schedules for optimal annealing,"Mathematics of Operations Research,vol. 13,1988,pp. 311—329.

[71] Handley,S. ,"The genetic planner: the automatic generation of plans for a mobile robot via genetic programming,"Proc. of 8th IEEE International Symposium on Intelligent Control,1993.

[72] Harrington,E. A. ,"Voice/data integration using circuit switched networks,"IEEE Trans. on Com-

munications, vol. 28, June 1980, pp. 781—793.

[73] Hart, P. E., N. J. Nilsson, B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," IEEE Trans. on SSC, vol. SCC-4, 1968, pp. 100—107.

[74] Hattori, S., J. Mizusawa and K. Murakami, "User programmability of communication network services with associative-memory neural networks," Electronics & Communications in Japan, Part 1., vol. 75, no. 11, November 1992, pp. 1—13.

[75] Haykin, S., Neural Networks: A Comprehensive Foundation. New York: Macmillan, 1994.

[76] Hellstrom, B. and L. Kanal, "Asymmetric mean-field neural networks for multiprocessor scheduling," Neural Networks, vol. 5, 1992, pp. 671—686.

[77] Hinton, G. E. and T. J. Sejnowski, "Learning and relearning in Boltzmann machines," in Parallel Distributed Processing, Volume 1 (Rumelhart, McClelland, and the PDP Group ed.), Cambridge, MA: MIT Press, pp. 282—317.

[78] Hiramatsu, A., "Integration of ATM call admission control and link capacity control of distributed neural networks," IEEE Journal on Selected Areas in Communications, vol. 9, no. 7, September 1991, pp. 1131—1138.

[79] Hiramatsu, A., "Training techniques for neural network applications in ATM," IEEE Communications Magazine, vol. 33, no. 10, October 1995, pp. 58—67.

[80] Hiriyannaiah, H. P., G. L. Bilbro and W. E. Snyder, "Restoration of piecewise-constant images by mean-field annealing," Journal of the Optical Society of America, vol. 6, no. 12, December 1989, pp. 1901—1912.

[81] Hoitimt, D. J., P. B. Luh, and K. R. Pattiati, "Job shop scheduling," First International Conference on Automation Technology, Taipei, Taiwan, July 1990, pp. 565—574.

[82] Hoitimt, D. J., P. B. Luh, and K. R. Pattiati, "A practical approach to job-shop scheduling problems," IEEE Transaction on Robotics and Automation, vol. 9, no. 1, February 1993, pp. 1—13.

[83] Holland, J., Adaptation in Natural and Artificial Systems. Ann Arbor: The University of Michigan Press, 1975.

[84] Hopfield, J. J., "Neural networks and physical systems with emergent collective computational abilities," Proc. of the National Academy of Sciences of the U. S. A., vol. 79, April 1982, pp. 2554—2558.

[85] Hopfield, J. J., "Neurons with graded response have collective computational properties like those of two-state neurons," Proc. of the National Academy of Sciences of the U. S. A., vol. 81, May 1984, pp. 3088—3092.

[86] Hopfield, J. J. and T. W. Tank, "Neural computation of decisions in optimization problems," Biological Cybernetics, vol. 52, 1985, pp. 141—152.

[87] Hou, E. S. H., N. Ansari and H. Ren, "A genetic algorithm for multiprocessor scheduling," IEEE Trans. Parallel and Distributed Systems, vol. 5, no. 2, February 1994, pp. 110—111.

[88] Hou, E. and D. Liu, "N/M job-shop scheduling with a genetic algorithm," in Intellient Automation and Soft Computing: Trends in Research, Development and Applications (M. Jamshildi ed.), New Mexico: TSI Press, vol. 1, 1994, pp. 511—516.

[89] Hsu, J. C. and A. U. Meyer, Modern Control Principles and Applications. New York: McGraw-Hill, 1968.

[90] Huttenlocker, D. P. and S. Ullman, "Object recognition using alignment," Proc. IEEE First Interna-

tional Conference on Computer Vision, London, England, 1987, pp. 102—111.

[91] Hwang, K. and F. A. Briggs, Computer Architecture and Parallel Processing. New York: McGraw Hill, 1984.

[92] Ingber, L., "Very fast simulated re-annealing," Mathematical and Computer Modeling, vol. 12, no. 8, 1989, pp. 967—973.

[93] Jeong, C. S. and M. H. Kim, "Fast parallel simulated annealing for traveling salesman problem on SMD machines with linear interconnections," Parallel Computing, vol. 17, no. 2/3, June 1991, pp. 221—228.

[94] Karp, R. M., "Reducibility among combinatorial problems," in Complexity of Computer Computations (R. E. Miller and J. W. Thatcher eds.), New York: Plenum Press, 1972.

[95] Kasahara, H. and S. Narita, "Practical multiprocessing scheduling algorithms for efficient parallel processing," IEEE Trans. on Computers, vol. C-33, no. 11, November 1984, pp. 1023—1029.

[96] Kasahara, H. and S. Narita, "Parallel processing of robot-arm control computation on a multimicroprocessor system," IEEE Journal of Robotics and Automation, vol. RA-1, no. 2, June 1985, pp. 104—113.

[97] Kim, S. -S. and C. -M. Kyung, "Module orientation algorithm using reconstruction of nets and mean field annealing," Electronics letters, vol. 27, no. 13, June 20, 1991, pp. 1198—1199.

[98] Kirkpatrick, S., C. D. Gelatt Jr. and M. P. Vecchi, "Optimization by simulated annealing," Science, vol. 220, 1983, pp. 671—680.

[99] Lenstra, J. K., A. H. G. Rinnooy Kan, and P. Bruckner, "Complexity of machine scheduling problems," Annals of Discrete Mathematics, vol. 7, 1977, pp. 343—362.

[100] Leong, H. W., D. G. Wong and C. L. Liu, "A simulated annealing channel router," Proc. IEEE international Conference on Computer-Aided Design, Santa Clara, CA, November 1985, pp. 226—229.

[101] Lin, F. T., C. Y. Kao and C. C Hsu, "Applying the genetic approach to simulated annealing in solving some NP-hard problems," IEEE Trans. on Systems, Man, and Cybernetics, vol. 23, no. 6, November 1993, pp. 1752—1767.

[102] Lin, S., "Computer solutions of the traveling salesman problem," Bell System Technical Journal, vol. 44, 1965, pp. 2245—2269.

[103] Liepin, G. E. and M. R. Hilliard, "Greedy genetics," Proc. 2nd Int. Conf. on Genetics Algorithms and Their Applications, Cambridge, MA, 1987, pp. 90—99.

[104] Lockwood, C. and T. Moore, "Harvest scheduling with spatial constraints: a simulated annealing approach," Canadian Journal of Forest Research, vol. 23, no. 3, March 1993, pp. 468—478.

[105] Luenberger, D. G., Linear and Nonlinear Programming. Reading, MA: Addison Wesley, 1984.

[106] Malek, M., M. Guruswamy and M. Pandya, "Serial and parallel simulated annealing and tabu search algorithms for the traveling salesman problem," Annals of Operations Research, vol. 21, no. 1/4, November 1989, pp. 59—84.

[107] McEliece, R. J., E. C. Posner, E. R. Rodemich and S. S. Venkatesh, "The capacity of the Hopfield associative memory," IEEE Trans. on Information Theory, vol. 33, July 1985, pp. 461—482.

[108] Metropolis, N. A., A. Rosenbluth, M. Rosenbluth, A. Teller and E. Teller, "Equation of state calculations by fast computing machines," Journal of Chemical Physics, vol. 21, 1953, pp. 1087—1092.

[109] Michalewicz, Z., Genetic Algorithms + Data Structures = Evolution Programs. Berlin Heidelberg: Springer-Verlag, 1992.

[110] Miller, R. E. and J. W. Thatcher, Complexity of Computer Computations. New York: Plenum Press, 1972.

[111] Morris, R. J. T. and B. Samadi, "Neural network control of communications systems," IEEE Trans. on Neural Networks, vol. 5, no. 4, July 1994, 639—650.

[112] Nakano, R. and T. Yamada, "Conventional genetic algorithms for job shop problems," Proc. 4th Int. Conf. On Genetics Algorithms and Their Applications, San Diego, CA, 1991, pp. 474—479.

[113] Ndousse, T. D. "Fuzzy neural control of voice cells in ATM networks," IEEE Journal on Selected Areas in Communications, vol. 12, no. 9, December 1994, pp. 1488—1495.

[114] Neves, J. E., M. J. Leitao and L. B. Almeida, "Neural networks in BISDN flow control: ATM traffic prediction or network modeling?" IEEE communications magazine, vol. 33, no. 10, October 1995, pp. 50—57.

[115] Nilar, S. H., "Applications of the simulated annealing method to intermolecular interactions," Journal of Computational Chemistry, vol. 12, no. 8, October 1991, pp. 1008—1013.

[116] Nilsson, N. J., Learning Machines: Foundations of Trainable Pattern Classifiers. New York: McGraw-Hill, 1965; also republished as The Mathematical Foundations of Learning Machines. San Mateo, CA: Morgan Kaufmann Publishers, 1990.

[117] Nilsson, N. J., Problem-solving Methods in Artificial Intelligence. New York: McGraw-Hill, 1971.

[118] Nilsson, N. J., Principles of Artificial Intelligence. Palo Alto, CA: Tioga Publishing Company, 1980.

[119] Nobakht, R. A., S. H. Ardalan and D. E. Van den Bout, "Adaptive filtering of nonlinear systems with memory by quantized mean field annealing," IEEE Trans. on Signal Processing, vol. 41, no. 2, February 1993, pp. 913—925.

[120] Nobakht, R. A., D. E. Van den Bout and J. K. Townsend, "Optimization of transmitter and receiver filters for digital communication systems using mean field annealing," IEEE Journal on Selected Areas in Communications, vol. 8, no. 8, October 1990, pp. 1472—1480.

[121] Nordstrom, E., J. Carlstrom and L. Asplund, "Neural networks for adaptive traffic control in ATM networks," IEEE Communications Magazine, vol. 33, no. 10, October 1995, pp. 43—49.

[122] Oliver, I. M., D. J. Smith, J. R. C. Holland, "A study of permutation crossover operators on the traveling salesman problem," Proc. of the Second International Conference on Genetic Algorithms, Cambridge, MA, July 28—31, 1987, pp. 224—230.

[123] Otten, R. H. J. M. and L. P. P. P. van Ginneken, The Annealing Algorithm. Boston, MA: Kluwer Academic Publishers, 1989.

[124] Ozcelik, T., J. C. Brailean and A. Katsaggelos, "Image and video compression algorithms based on recovery techniques using mean field annealing," Proc. of the IEEE, vol. 83, no. 2, Feb. 1995, pp. 304— 316.

[125] Papadimitriou C. and K. Steiglitz, Combinatorial Optimization: Algorithms and Complexity. New York: Prentice Hall, Inc., 1982.

[126] Park, Y.-K. and G. Lee, "Applications of Neural Networks in high-speed communication networks," IEEE Communications Magazine, vol. 33, no. 10, October 1995, pp. 68—75.

[127] Pearl, J., Heuristics: Intelligent Search Strategies for Computer Problem Solving. NY: Addison-Wesley Publishing Company, 1984.

[128] Peterson, C. and J. R. Anderson, "A mean field theory learning algorithm for neural networks," Complex Systems, vol. 1, 1987, pp. 995—1019.

[129] C. Peterson and J. R. Anderson, "Neural networks and NP-complete optimization problems: a performance study on the graph bisection problem," Complex Systems, vol. 2, 1988, pp. 59—89.

[130] Peterson, C. and J. R. Anderson, "Applications of mean field theory neural networks," Dept. of Theoretical Physics, Technical Report CS-1153, Univ. of Lund, August 1989, pp. 1—27.

[131] Peterson, C. and E. Hartman, "Explorations of the mean field theory learning algorithm," Neural Networks, vol. 2, August 1989, pp. 475—494.

[132] Peterson, C. and B. Soderberg, "A new method for mapping optimization problems onto neural networks," International Journal of Neural Systems, vol. 1, no. 1, 1989, pp. 3—22.

[133] Pospichal, J. and V. Kvasnicka, "Fast evaluation of chemical distrane by simulated-annealing algorithm," Journal of Chemical Information and Computer Science, vol. 33, no. 6, November 1993, pp. 879—885.

[134] Pritchard, W. L., H. G. Suyderhoud and R. A. Nelson, Satellite Communication Systems Engineering, 2nd Edition. Englewood Cliffs, NJ: Prentice Hall, 1993.

[135] Pratt, W. K., Digital Image Processing, 2nd Edition. New York: Wiley & Sons, 1991.

[136] Ramamoorthy, C. V. et al., "Optimal scheduling strategies in a multiprocessor system," IEEE Trans. Computers, vol. C-21, February 1972, pp. 137—146.

[137] Reddi, S. S. and C. V. Ramamoorthy, "On the folw-shop sequencing problem with no waiting process," Operational Research Quarterly, vol. 23, no. 3, 1972, pp. 323—331.

[138] Rong, L. and L. Ze-min, "Parameters rules of the Hopfield/Tank model on solving TSP," Proc. 1992 IEEE Conference on Neural Networks, Baltimore, MD, June 7—11, 1992, pp. IV. 492—497.

[139] Rose, C., "Low mean internodal distance network topologies and simulated annealing," IEEE Trans. on Communications, vol. 40, no. 8, August 1992, pp. 1319—1326.

[140] Rosenfeld, A., "Image Analysis and Computer Vision: 1993," Computer Vision, Graphics and Image processing, vol. 59, no. 3, May 1994, pp. 367—405.

[141] Rosenfeld, A., "Image Analysis and Computer Vision: 1994," Computer Vision and Image Understanding, vol. 62, no. 1, July 1995, pp. 90—133.

[142] Rosenfeld, A., "Image Analysis and Computer Vision: 1995," Computer Vision and Image Understanding, vol. 63, no. 3, May 1996, pp. 568—627.

[143] Schaffer, J. D., Proc. of the Third International Conference on Genetic Algorithms. San Mateo, CA: Morgan kaufmann Publishers, Inc., 1989.

[144] Schneider, C. R. and H. C. Card, "CMOS mean field learning," Electronics Letters, vol. 27, no. 19, September 12 1991, pp. 1702—1703.

[145] Schwartz, M., Telecommunication Networks: Protocols, Modeling and Analysis. Reading, MA: Addison-Wesley, 1987.

[146] Sechen, C., Placement and Global Routing of Integrated Circuits Using the Simulated Annealing Algorithm, University of California at Berkeley, Ph. D. Dissertation, 1986.

[147] Sengoku, M., K. Nakano and Y. Yamaguchi, "Channel assignment in a cellular mobile communication system and an application of neural networks," Electronics & Communications in Japan, Part

1,vol. 75,no. 4,April 1992,pp. 24—36.

[148] Skea,D. ,I. Barrodale,R. Kuwahara and R. Poecker,"A control point matching algorithm,"Pattern Recognition,vol. 26,no. 2,February 1993,pp. 269—276.

[149] Snyder,W. ,A. Logenthiran and P. Santago,"Segmentation of magnetic resonance images using mean field annealing,"Image and Vision Computing,vol. 10,no. 6,July 1992,pp. 361—368.

[150] Spears,W. M. and K. A. De Jong,"On the virtues of parameterized uniform crossover,"Proc. of the Fourth International Conference on Genetic Algorithms,pp. 230—236,1991. San Mateo,CA: Morgran Kaufman Publishers.

[151] Special Issue on Computational and Artificial Intelligence in High Speed Networks,IEEE Journal on Selected Areas in Communications,vol. 15,no. 2,February,1997.

[152] Sridhar,J. and C. Rajendran,"Scheduling in a cellular manufacturing system: a simulated annealing approach,"International Journal of Production Research,vol. 31,no. 12,December 1993,pp. 2927--2946.

[153] Syswerda,G. ,"Uniform crossover,"Proc. of the Third International Conference on Genetic Algorithms,1989,pp. 2—9.

[154] Staniforth,P. R. ,"Store-and-forward satellite communications system on UOSAT-2,"The Journal of the Institution of Electronic and Radio Engineers,vol. 57,January 1987,p. 43.

[155] Szu,H. H. and R. Hartley,"Fast simulated annealing,"Physics Letters A,vol. 122,June 1987,pp. 157—162.

[156] Szu,H. H. and R. Hartley,"Nonconvex optimization by fast simulated annealing,"Proc. of the IEEE,vol. 75,November 1987,pp. 1538—1540.

[157] Szykman,S. and J. Cagan,"A simulated annealing-based approach to three-dimensional component packing,"Journal of Mechanical Design,vol. 117,no. 2,June 1995 pp. 308—314.

[158] Thouless,D. J. ,P. W. Anderson and R. G. Palmer,"Solution of 'solvable model of a spin glass',"Philosophical Magazine,vol. 35,no. 3,1977,pp. 593—601.

[159] Tan,H. L. ,S. B. Gelfand and E. J. Delp,"A cost minimization approach to edge detection using simulated annealing,"IEEE Trans. on Pattern Analysis and Machine Intelligence,vol. 14,no. 1,January 1992,pp. 3—18.

[160] Tanaka,Y. and S. Hosaka,"Fuzzy control of telecommunications networks using automatic learning technique,"Electronics & communications in Japan—Part1,vol. 76,no. 12,December 1993,pp. 41—51.

[161] Umeyama,S. ,"Parameterized point pattern matching and its application to recognition of object families,"IEEE Trans. on Pattern Analysis and Machine Intelligence,vol. 15,no. 2,February 1993,pp. 136—144.

[162] Van den Bout,D. E. and T. K. Miller,III,"Graph partitioning using annealed networks,"IEEE Trans. on Neural Networks,vol. 1,no. 2,1990,pp. 192—203.

[163] Van Laarhoven,P. J. M. and E. H. L. Aarts,Simulated Annealing: Theory and Applications. Dordrecht,Holland: D. Reidel Publishing Company,1987.

[164] Vecchi,M. P. and S. Kirkpatrick,"Global wiring by simulated annealing,"IEEE Trans. on Computer-Aided Design,vol. 2,1983,pp. 215—222.

[165] Wang,G. and N. Ansari,"A neural network approach to broadcast scheduling in multi-hop radio networks,"Proc. 1994 IEEE Conference on Neural Networks,Orlando,FL. ,June 28-July 2,

1994,pp. 4699—4703.

[166] Wang,G. and N. Ansari,"Maximizing data throughput in an integrated TDMA communication system using mean field annealing,"Proc. 1994 IEEE Conference on Global Telecommunications, San Francisco,CA. ,November 28—December 2,1994,pp. 329—333.

[167] Wang,G. and N. Ansari,"Optimal broadcast scheduling in packet radio networks using mean field annealing,"IEEE Journal on Selected Areas in Communications,vol. 15,no. 2,February,1997, pp. 250—266.

[168] Wayman,J. L. ,"Optimization of signal distribution networks using simulated annealing,"IEEE Trans. on Communications,vol. 40,no. 3,March 1992,pp. 465—471.

[169] Witte,E. E. ,R. D. Chamberlain and M. A. Franklin,"Parallel simulated annealing using speculative computation,"IEEE Trans. on Parallel and Distributed Systems,vol. 2,no. 4,October 1991, pp. 483—494.

[170] Wolberg,G. and T. Pavlidis,"Restoration of binary images using stochastic relaxation with annealing,"Pattern Recognition Letters,vol. 3,1985,pp. 375—388.

[171] Wong,K. P. and Y. W. Wong,"Genetic and genetic/simulated-annealing approaches to economic,"IEE Proc. —C,Generation,Transmission,and Distribution,vol. 141,no. 5,September 1994, pp. 507—513.

[172] Woodruff,D. L. ,"Simulated annealing and tabu search: Lessons from a line search,"Computers and Operations Research,vol. 21,no. 8,1994,pp. 823—840.

[173] Wu,G. and J. W. Mark,"Capacity allocation for integrated voice/data transmission at a packet switched TDM,"IEEE Trans. on Communications,vol. 40,June 1992,pp. 1059—1069.

[174] Yip,P. P. C and P. -H. Pao,"Combinatorial optimization with use of guided evolutionary simulated annealing,"IEEE Trans. on Neural Networks,vol. 6,no. 2,March 1995,pp. 290—295.

[175] Yuhas,B. and N. Ansari,(eds.),Neural Networks in Telecommunications. Norwell,MA: Kluwer,1994.

[176] Yuille,A. L. ,D. Geiger and H. H. Bulthoff,"Stergo integration,mean field theory and psychophysics,"Network: Computation in Neural Systems,vol. 2,no. 4,November 1991,pp. 423—442.

[177] Zerubia,J. and R. Chellappa,"Mean field annealing using compound Gauss-Markov random fields for edge detection and image estimation,"IEEE Trans. on Neural Networks,vol. 4,no. 4,July 1993,pp. 703—709.

[178] Zhang,Z. Z. ,N. Ansari,E. Hou and P. Yi,"Multiprocessor scheduling by mean field theory," Proc. of the IJCNN-92,Vol. IV,June 1992. pp. 582—587.

[179] Zhao,M. ,N. Ansari and E. S. H. Hou,"Mobile manipulator path planning by a genetic algorithm,"Journal of Robotic Systems,vol. 11,no. 3,pp. 143—153,1994.

[180] Zhou,D. N. ,V. Cherkassky,T. R. Baldwin,and D. E. Olson,"A neural network approach to job-shop scheduling,"IEEE Trans. on Neural Networks,vol. 2,no. 1,January 1991,pp. 175—179.

名词术语中英文对照表(以汉语拼音排序)

A

鞍点	saddle point
鞍点逼近	saddle-point approximation
鞍点展开	saddle-point expansion

B

玻耳兹曼常数	Boltzmann constant
玻耳兹曼分布	Boltzmann distribution
本征值	eigenvalue
本征向量	eigenvector
本征值分析	eigenvalue analysis
本征空间	eigenspace
玻尔兹曼机	Boltzmann machine
白噪声	white noise

D

代数重数	algebraic multiplicity
多项式时间	polynomial time
动态系统	dynamic system
动力学	dynamics
定长边界	fixed length boundary
等位基因	alleles
代	generation
单调增函数	monotonically increasing function
多处理器调度问题	multiprocessor scheduling problem
点模式匹配	point pattern matching
调度	schedule
电信业务	telecommunications services
多项式时间约化	polynomial time reduction

E

二值硬限器	binary hardlimiters
二值符号串	binary string
二维点模式	two-dimensional (2-D) point pattern

F

放大器	amplifier
非对称均场网络	asymmetric mean-field network
复杂度	complexity
复杂度函数	complexity function
反馈电路	feedback circuit
繁殖	reproduction
分次神经元	graded neuron
返回神经网络	recurrent neural network
分时多路存取	time-division multiple access(TDMA)

G

概率密度函数	probability density function
归一化吞吐量	normalized throughput
高斯机	Gaussian machine
更新	updating

H

回溯	backtracking

J

接受概率	acceptance probability
接受比	acceptance ratio
激活	activation
激活能	activation potentials
接受规则	acceptance rule
均场退火	mean field annealing
渐近时间/空间复杂度	asymptotic time/space complexity
计算复杂度	computational complexity

计算智能	computational intelligence
计算机辅助设计	computer-aided design
计算机视觉	computer vision
晶体	crystalline
距离矩阵	distance matrix
距离参数	distance parameter
几何重数	geometric multiplicity
结构单元	building blocks
基因	gene
交叉	crossover
局部搜索	local search
局部最小	local minima
均场逼近	mean field approximation
均场方程	mean field equation
均场网络	mean field net
均场变量	mean field variable
均场向量	mean field vector
决策问题	decision problems
计算复杂性理论	theory of computational complexity
均匀分布	uniform distribution

K

空间复杂度	space complexity
可按内容检索的记忆	content addressable memory(CAM)
柯西机	Cauchy machine
可满足性问题	satisfiability problem

L

连接矩阵	connection matrix
临界温度	critical temperature
累积高斯分布函数	cumulative Gaussian distribution
拉格朗日参数	Lagrange parameter
旅行商问题	traveling salesman problem(TSP)

M

模拟退火	simulated annealing
马尔可夫链	Markov chain
模式识别	pattern recognition

N

N-皇后问题	N-Queen problem
NP 完全理论	theory of NP-completeness

P

泊松过程	Poisson process

Q

群体	population
全局最小	global minima
启发函数	heuristic function
启发搜索	heuristic search
群体空间	population space

R

扰动规则	pertubation rule
染色体	chromosome
锐化	sharpening
软限幅器	soft-limiter
任务图	task graph
热力学	thermodynamics

S

算法	algorithm
双极	bipolar
数据吞吐量	data throughput
数据结构	data structure
适度值	fitness value
适度函数	fitness function
S 形函数	Sigmoid function
随机噪声	random noise
随机数	random number
随机数发生器	random number generator
随机点过程	random point process
随机过程	random process
随机搜索	random search
随机变量	random variable
收敛速率	rate of convergence
随机激活函数	stochastic activation function

随机方程	stochastic equations
随机机	stochastic machine
随机神经元	stochastic neurons
随机松弛法	stochastic relaxation
扫描	sweep
随机更新	stochastic updating
尝试法	trial and error

T

退火	annealing
退火过程	annealing process
退火流程	annealing schedule
突变	mutation
统计力学	statistical machanics
统计物理学	statistical physics
停止准则	stopping criterion
停止参数	stopping parameter
同步更新	synchronous updating
特征方程	characteristic equation
特征多项式	characteristic polynomial

W

卫星广播调度	satellite broadcast scheduling(SBS)
稳定性	stability
稳定构形	stable configuration
稳定点	stable point
稳定状态	stable state
稳态	steady state

X

选择规则	selection rules
形状特征	shape features
形状识别	shape recognition
相似变换	similarity transformation

Y

异步传递方式	asynchronous transfer mode
有色噪声	colored noise
一致性函数	consensus function
遗传算法	genetic algorithms
异联想	hetero-associative
硬限器	hardlimiter
异步更新	asynchronous updating

Z

自联想	auto-associative
指数函数	exponential functions
指数时间	exponential time
帧格式	frame format
帧模式	frame pattern
直方图	histograms
作业调度问题	job shop scheduling problem
最小生成树	minimum spanning tree
最优化问题	optimization problem
最优调度	optimal schedule
终止准则	termination criteria