第一章　引言

1.1　最优化问题

最优化(optimization)是一个具有悠久研究历史的数学问题。在17世纪——英国伟大科学家Newton发明微积分的时代，就对极值问题进行了研究。后来，出现了Lagrange乘数法。1847年，法国数学家Cauchy提出函数值最速下降算法。1939年,前苏联数学家Л. В. Канторович提出了用于解决下料和运输等线性规划问题的计算方法。从20世纪40年代以来,随着电子计算机、复杂的经济管理系统、大型工程设计及军事科技的发展，最优化理论和方法进入了新的发展时期，成为数学等科学领域中一门迅速发展的独立学科，并在许多领域，特别是军事领域获得了广泛应用，成为军事运筹学的一个重要基础。

1.1.1　最优化问题的定义、数学模型和解的分类

日本茨木俊秀教授给出最优化问题一个较简单的定义[1]：

最优化问题一般是指在某种状态下作出最好的决策，或者从几个候选决策中选出最优的决策。

最优化问题通常还可用数学模型描述为在给定的约束条件(constraint)下,求出一个决策变量(decision variable)的值,使得目标函数(objective function)为最小或最大值。因为求目标函数最大值等价于把它乘以 -1 后求最小值，所以为便于讨论起见，本书主要介绍求目标函数最小值的情况。最优化问题可用

需要全本请访问 www.ertongbook.com

下式表示：

$$\min_{x \in s} f(x)$$
$$\text{s.t.}\ g(x) \geqslant 0 \qquad (1.1)$$

其甲：

x 为决策变量或决策状态变量（以下简称为变量或状态），可表示为一个 n 维向量 $x=(x_1,x_2,\cdots\cdots,x_n)$；

S 为决策变量 x 的定义域，它是 n 维实数 R^n 上的有界子集合，S 可用与变量 x 相关的等式及不等式表示；如果难以用数学公式表示，也可采用图形等方法表示；

$f(x)$ 为目标函数，是定义在 S 的 n 维实数值函数；

$g(x)$ 是约束条件，如果难以用数学公式表示它，还可采用关联图或网络图等方法给出。

如果 x 满足约束条件 $x \in S$，则称它为式（1.1）所示最优化问题的可行解（feasible solution）。因此，S 也称为可行域（feasible region），或者称为可行解空间，甚至也可简称为解空间。

如果最优化问题的可行解 x^* 满足：

$$f(x^*) \leqslant f(x) \quad (x \in S) \qquad (1.2)$$

则称它为此问题的最优解（optimal solution）。

如果在包含可行解 $x^* \in S$ 的适当邻域 $U(x^*)$ 里，满足：

$$f(x^*) \leqslant f(x) \quad (x \in (S \cap U(x^*))) \qquad (1.3)$$

则称它为此问题的局部最优解（local optimal solution），也可简称为优化解或近似最优解。为区别式（1.2）所表示的最优解与式（1.3）所表示的局部最优解，也可将最优解称为全局最优解（global optimal solution）。

许多最优化问题决策变量 x 的可行域非常庞大，有时在寻找问题解时还可能面临搜索空间‘爆炸’的困难 也有许多最优

化问题的目标函数或约束条件很复杂，人们常常是花费了很长时间，却只得出局部最优解甚至仅为可行解，要找出全局最优解非常困难。

本书的重要议题就是使用新方法高效率地求得最优化问题的可行解、局部最优解及全局最优解。

1.1.2 最优化问题的分类

根据变量、目标函数及约束条件的类型，可将最优化问题分类[1]。

按变量的连续或离散，可将最优化问题分为变量取连续实数的连续最优化问题（continuous optimization problem）以及取整数或者仅取 0 或 1 的离散最优化问题（discrete optimization problem）。后者因多用组合方法在决策变量的定义域中找到最优解，常涉及分组、排序、筛选等问题，也称为组合优化问题（combinatorial optimization problem），组合优化对象是有限解空间内的离散变量。本书将重点讨论组合优化问题求解的新方法及其军事应用。

连续最优化问题可分为两类：

1. 线性规划问题（linear programming problem）

如果最优化问题的目标函数是线性函数、约束条件是线性方程组或(和)不等式组，则此问题为线性规划问题。这是一种较易解的最优化问题。

2. 非线性规划问题（nonlinear programming problem）

如果最优化问题的目标函数和约束条件不都是线性的，则此问题为非线性规划问题。这是一种较难解或难解的最优化问题。

离散最优化问题主要根据其约束条件的类型分类。例如，变量为整数的问题称为整数规划问题（integer programming

problem)。变量的值为 0或 1 的问题称为 0 - 1 规划问题(0 - 1 programming problem)。另外，约束条件由关联图或网络图给出的问题也很多。例如，处理网络流程的各种网络流问题 (network flow problem)，尤其是属于组合优化问题的旅行商问题 (TSP, Traveling Salesman Problem),它可以表述为 给定 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,使每个城市只经过一次且总的旅行距离最短。本书中将多次提及 TSP。

1.2 计算复杂性

计算复杂性(computing complexity)理论研究计算方法的有效性和问题求解难度[1,2,3,4] ,是最优化理论的基础。例如,无法找到一个计算程序,可在任意一种输入下,判定该程序是否会停止，这就是著名的停机理论(Halt Theorem) 。某些问题虽然从理论上讲可以用算法解决，但是用当今最快的计算机也需要数个世纪的计算时间,实际上也是无法解决的。因此,在求解一个最优化问题之前，了解它是否为难解问题及设计求解问题算法的复杂性是很重要的。对于难解问题，通常采用近似计算方法及新的最优化方法。

有的新最优化方法具有独特的长处，例如神经网络具有自学习功能,在求解较复杂、不一定是难解的最优化问题时,也可以酌情选用。

1.2.1 计算复杂性的基本概念

1965 年,J. Edononds 提出了以计算时间作为算法复杂性评价标准及多项式时间算法[5]，这是计算复杂性的一个基础理论。

1.计算复杂性

一个问题通常由无数个问题实例(problem instances)组成。例如,TSP 就是由给定节点数、各节点间距离及对应解的无数问题实例集合组成。求解某个问题的算法,应能正确求解该问题的任何实例。处理数字的算法一般由算术和逻辑运算等基本步骤组成(以下所述算法均是指此类算法)。解决某个问题所用算法需要执行的基本步骤数量称为时间复杂性(time complexity),简称为时间量;该算法为了保存中间结果所占用的存储空间称为空间复杂性(space complexity)。这二者统称为算法的计算复杂性,也简称为计算复杂性或计算量。

通常采用两种方法来计算一个算法的基本步骤数量:

(1)单位成本(unit cost)法

把每一次加、减、乘、除等运算都作为一次基本运算。

(2)对数成本(logarithmic cost)法

把运算分解成二进制数的位运算形式,每一次位运算作为一次基本运算。在自然数 x 化为二进制数时,其位数大致为 $\log_2 x$,对数成本法的命名由此而来。

采用上述两种方法所得结果因所处理数 x 的计算机字长(二进制数的位数)不同而不同。例如,在 n 位的两个数求和情况下,采用单位成本法时其计算量是 1,而采用对数成本法的计算量约为 $c_1 + c_2, c_1$ 和 c_2 为特定的常数。当利用一定的计算机字长即能以足够精度表示所有处理的数时,通常采用单位成本法 当不同的数字需要不同位数才能以足够的精度表示时,采用对数成本法比较合适。

2. 多项式阶

算法 A 的时间复杂性可用函数 $T_A(n)$表示,其中 n 为整数,亦称为问题规模。例如,TSP 的城市数 n 即为问题规模。$T_A(n)$ 可以用多项式函数 $p(n)$表示,还可以用多项式主要项的阶 $O(p(n))$ 来表示,亦称为多项式阶(polynomial order)。例如,

需要全本请访问 www.ertongbook.com

虽然 $p(n)$可分别表示为 $0.5n^2$,$100n+5n^2$及 $50-10^4n+10^{-3}n^2$等不同形式，但是其多项式阶均为$O(n^2)$,均可表示为 $T_A(n)=O(n^2)$。

在评价算法 A 的时间复杂性时，常用方法就是看 $O(p(n))$ 是否为多项式阶。如果存在某个正常数 n_0和 c,当 $n\geqslant n_0$ 时,有$T_A(n)\leqslant c\,p(n)$,则称 $O(p(n))$为多项式阶,可表示为 $T_A(n)=O(p(n))$。例如，$T_A(n)=n+5n^2$,存在正常数 $n_0=0$ 和 $c=6$,当 $n\geqslant 0$ 时,有 $T_A(n)\leqslant 6n^2$,则称 $O(n^2)$ 为多项式阶，记为 $T_A(n)=O(n^2)$。用整数 k 表达的 $O(n^k)$是多项式阶,像 $O(n^2\log n)$ $(\leqslant O(n^3))$这样含有 $\log n$ 的也是多项式阶，而 $O(2^n)$,$O(n^{\log n})$,$O(n!)$,$O(n^n)$等均不是多项式阶。

3. 多项式时间算法和指数时间算法

如果 $O(p(n))$为多项式阶,则称算法 A 为多项式时间算法(polynomial time algorithm) 如果 $O(p(n))$ 不是多项式阶，则称为指数时间算法 (index time algorithm)[4,13]。通常把存在多项式时间算法的问题看作是易解问题，把没有找到多项式时间算法的问题看作是难解问题。

4. 评价算法计算量的标准

评价算法计算量通常采用如下两项标准：

(1)最坏计算量(worst case complexity)

在所有问题实例中所需要的最大计算量。

(2)平均计算量(average complexity)

利用问题实例各自发生的概率来计算所有问题实例平均计算量。

由于求最坏计算量在数学处理上较容易，所以通常用此法来评价算法计算量。但是，如果问题只有少数实例需要异常大计算量，采用它会对算法计算量得出不符实际的过低评价。

5. 算法计算量随问题规模变大的增长率

问题规模增大时，通常算法计算量也会相应地增大。因此，现在还将算法占用时间随问题规模变大的增长率作为评价算法复杂性的重要指标。随着问题规模的增大，指数函数值要比多项式函数值的增长速度快得多。

常见算法时间复杂性函数，按复杂性程度由小到大排列如下：

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

以使用每秒运算 10 亿次的计算机为例，表 1.1 列出上述复杂性函数在问题规模变大时的计算量需用时间增长情况[13]。

表 1.1 常见算法时间复杂性函数在问题规模变大时计算量需用时间增长情况

时间复杂性函数	问题规模 n				
	10	20	30	40	100
n	10ns	20ns	30ns	40ns	100ns
$n\log n$	10ns	26ns	44.3ns	64.1ns	200ns
n^2	100ns	400ns	900ns	1.6μs	10μs
2^n	1.0μs	1.0ms	1.1s	18.3min	4.0 世纪
$n!$	3.6ms	77.1 年	8.4×10^{13} 世纪	2.6×10^{29} 世纪	3.0×10^{139} 世纪

1.2.2 P, NP, NP-C 和 NP-hard 问题

无法用多项式时间算法求解的最优化问题，被称为难解问题。1971 年，S. A. Cook 提出了 NP 完全问题理论，用于确定问

题的难解程度及问题解之间在难易程度方面的相互关系[6]。自从他提出 NP 完全问题理论以来,人们发现了数以千计的 NP 完全问题。它们大多是难解问题,分布于逻辑、代数、数论、自动机、图论、网络理论、组合优化及军事运筹等许多领域。NP 完全问题理论已成为计算复杂性的一个基础理论,与它有关的基本概念简要介绍如下。

1. 判定问题

判定问题只有‘是’或‘否’两类答案。虽然最优化问题表面看来不一定是判定问题,但是可以很容易地把它转化为判定问题,从而简化该问题的求解。例如,可以将 TSP 转化为‘是否有长度不超过一个自然数 k 的旅行路线使得从某城市出发走遍所有的城市后回到出发地’的判定问题。为了求解此问题,可以先估计最短旅行路线的范围。例如,它不会短于两城市间最短路径长度 a,也不会超过所有两城市间路径长度之和的两倍 b。取 $k=(a+b)/2$,解相应判定问题。若答案是肯定的,那么可判定最短旅行路线长在 $[a,(a+b)/2]$ 之中。若答案是否定的,那么可判定最短旅行路线长在 $[(a+b)/2,b]$ 之中。这样,至多解 $[\log_2(b-a)]<b-a$ 次相应判定问题,就可以找到原问题的解。

由于用原问题的解给出相应判定问题的解,或者根据相应的判定问题的解来确定原问题的解,两者难度一样。所以,当论述求解某最优化问题时,通常可简化为论述求解其相对应的判定问题。

2. P 问题

一个判定问题一般可以有多个算法,当该问题至少存在有一个多项式时间算法时,称之为多项式问题,简称为 P 问题(polynomial)。通常把属于 P 的问题看作是易解问题。例如,线性规划问题、2 次规划问题、最大流问题及最小费用流问题等

均属于 P 问题。

3. NP 问题

把找不到多项式时间算法的判定问题称为非多项式确定问题,简称为 NP 问题(NP,nondeterministic polynomial)。

NP 问题的定义如下[3,4]:

若存在一个多项式函数 $g(x)$ 和一个验证算法 H,对一类判定问题 A 任何一个“是”的判定实例 I 都存在一个字符串 S 是 I 的“是”答案,满足其输入长度 $d(S)$ 不超过 $g(d(I))$,其中 $d(I)$ 为 I 的输入长度,并且验证算法验证 S 为 I “是”答案的计算时间 $f(H)$ 不超过 $g(d(I))$,则称判定问题 A 为非多项式确定问题,简记为 NP。

由上述定义可知,判定问题是否属于 NP 的关键在于对“是”的判定实例,是否存在满足上述定义的一个字符串和算法,该字符串可理解为判定问题的一个解。由于定义中没有强调字符串和算法是如何得到的,可见,$P \subset NP$。

4. 多项式时间归约

1971 年,R. Karp 提出了多项式归约(reduction)的概念[7]。当问题 A_1 可在多项式时间内化为问题 A_2 的一个实例时,称 A_1 可多项式归约于 A_2。如果 A_1 可在多项式时间内归约于 A_2,那么 A_1 的解可以通过如下途径找到:先把 A_1 化为 A_2,对 A_2 求解,此解正对应于 A_1 的解。如果可将 A_1 在多项式时间内归约化为 A_2,那么当 A_2 有多项式时间算法时,A_1 也一定有。反之,若 A_1 没有多项式时间算法,则 A_2 也一定没有。这从某种程度上反映了 A_1 和 A_2 问题解之间的难易关系。

多项式归约的定义如下[3,4]:

对于给定的判定问题 A_1 和 A_2,如果存在多项式函数 $g_1(x)$ 和 $g_2(x)$,使得对 A_1 的任何一个实例 I,在多项式时间内构造 A_2 的一个实例,其输入长度不超过 $g_1(d(I))$,并对 A_2 的任何一个

算法 H_2,都存在问题 A_1 的一个算法 H_1,使得 H_1调用 H_2且计算时间 $f_{H1}(d(I)) \leqslant g_2(f_{H2}(g_1(d(I))))$,则称判定问题 A_1 可多项式归约为 A_2。

5. NP – C 问题

可以利用多项式时间算法归约化为 NP 问题的判定问题,称为 NP 完全问题(NP – C,NP – Completeness)。NP – C 是否难解至今仍然没有答案,成为一个十分重要的待解决问题。通常认为如果一个问题属于 NP – C,则它多半是难于求解的。这就迫使人们转向研究近似算法、启发式优化算法及其他新方法。目前已证明 TSP、背包问题、装箱问题、车间作业调度问题等许多问题都是 NP – C[3,4,8]。

6. NP – hard 问题

可以利用多项式时间算法归约化的判定问题,称为 NP 难解问题(NP – Hardness,简记为 NP – hard)。特别要说明的是,它包括两类问题:一类是 NP – C,即可以利用多项式时间算法归约化为 NP 问题的判定问题;另一类虽然是可以利用多项式时间算法归约,但不能归约化为 NP 问题的判定问题。人们也在研究用启发式优化算法及其他新方法求解此类问题。

图 1.1 表示了 P 问题、NP 问题、NP – C 问题和 NP – hard 问题等四类问题的从属关系。

综上所述及如图 1.1 所示,可以归纳出如下 4 点结论:

(1)P 问题是 NP 问题的子集,即 $P \subset NP$,它是易解问题;

(2)在 NP 问题中,除了所包含的 P 问题和 NP – C 问题外,还有部分问题的复杂性是未知的;

(3)NP – C 问题是 NP 问题与 NP – hard 问题的交集,其中大多是难解问题;

(4)NP – hard 问题包含 NP – C 问题,即 $NP - C \subset NP - hard$,其中大多是难解问题。

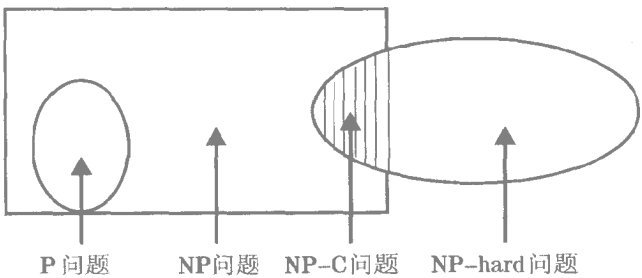


图 1.1 P 问题、NP 问题、NP－C 问题和 NP－hard 问题的从属关系

1.2.3 军事最优化问题的计算复杂性

在求解复杂、大规模的军事最优化问题之前，应该了解它是否为难解问题及设计求解该问题算法的计算复杂性。求解军事最优化问题常见的困难之一是必须搜索大量的候选决策状态，即可行解空间。随着问题规模的增大，计算量常常急剧增大，甚至发生所谓待搜索空间的“组合爆炸”，造成问题难解。因为本书讨论的重点是用新方法求得军事最优化问题的优化解，而不是对这些问题的算法复杂性函数、基本运算步骤数量、计算量随问题规模变大的增长率进行详细分析，所以下面仅根据需要检测的状态数量来简要介绍在本书中将重点讨论的若干军事最优化问题计算复杂性。

1. 空战战役兵力计划优化问题

假设空战的红、蓝双方各有M_r及M_b种飞机，双方每种飞机各有N_r及N_b种可能的状态，则需要检测的状态数量为$M_r^{N_r} \times M_b^{N_b}$个[8]。如果交战双方各有10种飞机，每种飞机各有10种

需要全本请访问 www.ertongbook.com

可能的状态，则需要检测的状态数量高达 10^{20}。此问题很可能是一个难解问题，以下简称为“兵力使用问题”(Force Allocation Problem)，它的复杂性函数为指数函数，类似于 1.2.1 节所述的算法复杂性函数 $O(n^n)$。

2. 多机空战目标分配方案优化问题

假设红、蓝双方参战的飞机分别为 M 架和 N 架，则双方攻击目标分配的候选决策方案总数为 $M^N \times N^M$。如果是 2 对 2 空战，则可以有 16 种分配方案，而 4 对 4 就急剧增加到 65536 种，32 对 32 则为 2^{320}种[10]。此问题很可能也是一个难解问题，以下简称为“目标分配问题”(Target Allocation Problem)，它的复杂性函数也类似于指数函数 $O(n^n)$.

3. 飞机规避来袭导弹及导弹突防策略优化问题

在空战中，飞机为了能规避来袭导弹，在一段时间内，需要不断地优化飞机机动策略，力求摆脱导弹的攻击。由于导弹飞行速度很快，报警时间很短，仅由飞行员选择机动动作来规避导弹是很困难的。目前，美军正在大力研制各种无人驾驶飞机，研究无人机在机载计算机和雷达密切配合下的自动规避导弹方法。解决无人机控制中候选决策空间呈现指数级“爆炸”的难题，是其中的重要课题[11]。

设问题规模为 1 架飞机规避 1 枚导弹，作者在空战仿真中发现：

(1)设空战持续时间为 3min，仿真软件运行的时间步长为 0.1s。由于飞机速度可达 600m/s 左右，导弹速度可达 1500m/s 左右，飞机与导弹之间的距离在 0.1s后的变化可达 200 多米，所以仿真软件在 3min 内至少需要为裁决空战结果进行 1800 次决策。

(2)设飞机可选择 16 种基本战术动作、5 种推力、8 种坡度及 6 种俯仰角，它与敌方飞机的相对态势关系分为 91 种[10]，则

仿真软件需要为它检测 349440 种候选决策。

(3) 以美国的 AIM－120 先进中程空空导弹为例，其火控雷达有 4 种工作状态,导弹有 3 种发射方式、3 种中制导方式及 3 种末制导方式。仿真软件需要为导弹检测 108 种攻击目标机的候选决策。

如果交战双方飞机分别为 M 架和 N 架,每 1 架飞机需要规避多枚导弹，随着问题规模增大，求解的计算量将会剧增。此问题也是一个难解问题，以下简称为“规避导弹问题”，它的复杂性函数也类似指数函数 $O(n^n)$.

目前，美国正在大力推行“战区导弹防御系统”(TMD)和“国家导弹防御系统”(NMD)计划。作为对美国的回应，俄罗斯宣布其白杨－M 导弹采用变轨机动飞行的突防策略可突破 NMD 拦截。与飞机规避来袭导弹问题相似，弹道导弹突防策略优化问题也可能是难解问题。

4. 卫星问题

(1) 人造卫星调度问题

根据文献[12]可知军事侦察卫星网络与地面站之间的通信调度管理优化问题是一个 NP 难解问题。例如,设有 4 颗卫星、3 个地面站和 9 个通信时间段,如果 4 颗卫星对 3 个地面站各分配 2 个时间段，则需要检测的决策状态数量为 $2\times2\times2\times2\times3=48$ 个。如果 4 颗卫星对 3 个地面站分别分配 9、8、7、6 个时间段，则需要检测的决策状态数量为 $9\times8\times7\times6\times3=9072$ 个[13]。随着卫星、地面站和通信时间段数量增加，即问题规模增大，决策状态数量急剧增大，则该问题变成一个难解问题。

(2) 由地球、月亮和人造地球卫星（或宇宙飞船）构成的三体问题

三体问题是 n 体问题的简化。所谓 n 体问题,就是研究 n 个物体之间的引力问题。自从 Newton 发现万有引力定律及

Newton 运动第二定律之后，Newton 本人和 17、18 世纪的数学家、物理学家就早已解决了二体问题。他们利用包括六个二阶方程的常微分方程组，求解了太阳和地球的运动问题，获得了精确的解析解。但是，人们至今仍无法求出由地球、月亮和人造地球卫星（或宇宙飞船）构成的三体问题精确的解析解，即使求得其近似解也需要使用电子计算机来完成很大的计算量。三体问题至今仍然是一个难解问题。

5. 特定作战效能指标对武器使用的影响问题

这是在作战计划决策中很常见的一类较难解的优化问题，以下简称为“效能指标的影响问题”，它可以进一步划分为如下几个类似的问题：

（1）为达到一定毁伤效果，给出不同的武器编配方案，以下简称为“武器编配问题”（Weapon Mix Problem）[14]；

（2）为达到一定毁伤效果，计算所需要的最少弹药，以下简称为“最少弹药问题”；

（3）上一问题可以在将约束条件放松后变成“为达到一定毁伤效果，如何节约弹药问题”，以下简称为“节约弹药问题”。该问题不是求最优解，而是求近似最优解。

例如，在研究优化红军利用反舰导弹打击蓝军水面舰艇编队的海战计划中，就需要根据红军打击蓝军水面舰艇编队毁伤指标来确定最优的反舰导弹到达分布，这是制定作战计划的重要依据。与通常作战仿真系统采用“制定作战计划→为武器平台分配任务→计算武器到达分布→计算打击效果”的计算流程方向正相反，解决上述问题需要采用“达到预定打击效果→优化武器到达分布→减少武器平台数量→制定／改进作战计划”的计算流程。进行相反方向计算的主要困难是在每一个阶段必须检测大量决策状态，搜索庞大的决策空间。当攻击武器及发射平台种类和数量增大时，由于许多不确定因素和武器数据很

难确定，问题求解空间将急剧增大，从而很难得到问题的最优解。

6. 大型军用通信网络的管理问题

这是在军事指挥通信系统中常见的一个优化问题。它要求动态、合理地为通信网络各节点及线路分配负荷，能自动处理通信节点或线路的故障、加入新的节点及线路，尽可能为大量的通信用户选择最短通信路径，减少通信阻塞及呼叫失败的次数。为了减少问题的计算复杂性，假定通信网络节点之间的连线具有无限的容量，不考虑连线的节点之间的地理距离，只把网络节点容量作为唯一的通信瓶颈。在满足上述假设条件并拥有 n 个节点的通信网络中，如果每一个节点有 k 个相邻节点，最多可同时接通 m 个呼叫，则每一个节点有 $(n-1)\times k\times m$ 种状态，整个网络的通信至少约有 $n\times(n-1)\times k\times m$ 种状态[15]。当战场情况突变导致用户数量急剧增加时，解决通信网络管理问题将因搜索的决策状态数量急剧增大而变成一个难解问题。

在本书中，将详细讨论如何使用启发式优化算法及其他新方法求解各种难解或较难解的军事最优化问题。

1.3 启发式优化算法

1997 年，印度军事运筹学专家 N. K. Jaiswal 在其《军事运筹学 定量决策》一书中介绍了包括神经网络、遗传算法和模拟退火在内的启发式优化算法(Heuristic Algorithm)[16]。1999 年，邢文训在《现代优化计算方法》一书中介绍的启发式优化算法不仅包括神经网络、遗传算法、模拟退火算法，还包括拉格朗日松弛算法及禁忌搜索算法[3]。2000 年，王凌在《智能优化算法及其应用》一书中将神经网络、混沌、进化计算、遗传算法、进化规划、模拟退火、禁忌搜索及其混合优化策略统称为智能优化算法(Intelligent Optimization Algorithms)，或称为现代启发式优化算

法[4]。现在，蚁群算法也成为一种新的启发式优化算法[17]。

20 世纪 60 年代以来，人们发现很多优化问题，如 TSP、图的划分问题及运输调度问题等，随着问题规模增大，计算量将按指数方式增大，发生所谓的“组合爆炸”。启发式优化算法是解决此类问题的有效算法。

启发式优化算法可以定义为：一种基于直观或经验构造的算法，在可接受的计算时间、占用资源等条件下，给出难解最优化问题的一个优化解或可行解。它不能保证找到全局最优解，有时甚至不能估计该优化解或可行解与最优解的偏离程度[3,4]。

启发式优化算法是相对于最优算法提出的。20 世纪 70 年代以来，出于求解实际问题的需要，随着计算复杂性理论的完善，人们不再强调一定要求最优解。这是促使遗传算法和神经网络等启发式算法，及其他许多新优化算法迅速发展的一个重要原因。从 1993 以来,作者在多机空战仿真研究中,在求解多目标分配、考虑多因素的机动动作选择、在空战各个时间段中飞机多批次出动计划等问题时，就遇到了随着飞机数量增加计算时间增长率过大的难题。作者尝试利用遗传算法优化多机空战目标分配及飞机规避导弹策略，利用神经网络优化空战机动动作选择，取得了令人满意的优化解和较快的计算速度。实践证明神经网络、遗传算法、模拟退火、蚁群算法等启发式优化算法不仅在军事运筹学和作战仿真中具有良好的应用前景，还可在其他众多军事领域,例如决策支持、通信、情报侦察、电子战、指挥系统、导弹攻防、航天、军用卫星、飞机车辆和舰艇驾驶等领域中,获得更广泛的应用。

近年来，启发式优化算法已成为许多决策支持系统（DSS, Decision Support System）使用的一种新方法。决策支持系统是由美国 M. S. Scott Morton 于 20 世纪 70 年代初在《管理决策系

统》一文中提出的。早期的决策支持系统多使用模型库、数据库和信息检索技术,采用结构化知识及传统的数值算法,追求问题解的精确结果和最优解。由于在许多实际决策问题中，决策者的主要困难是必须检测大量的候选决策状态，搜索庞大的决策状态空间，所以后来发展的新型决策支持系统使用了专家系统、知识库和非结构化知识，采用了模糊逻辑和概率推理技术，采用了神经网络、遗传算法等启发式算法及其他许多新方法。图 1.2 是决策支持系统使用的主要方法及发展趋势示意[18]。

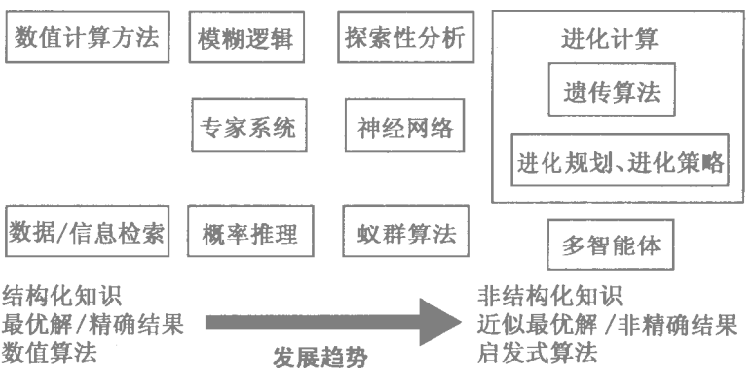


图 1.2 决策支持系统使用的主要方法及发展趋势示意图

1.4 从神经网络和遗传算法到计算智能

神经网络（以下简记为 NN, Neural Network）和遗传算法（以下简记为 GA, Genetic Algorithm）是计算智能这门 21 世纪新兴学科的基础,是生命科学、计算机科学、人工智能、认知科学和数理科学等的交叉学科。文献[4]将 NN 和 GA 等启发式优化算法又称为智能优化算法。

需要全本请访问 www.ertongbook.com