Management Science

Publication details, including instructions for authors and subscription information:
http://pubsonline.informs.org

Deterministic Production Planning: Algorithms and Complexity

M. Florian, J. K. Lenstra, A. H. G. Rinnooy Kan,

To cite this article:
M. Florian, J. K. Lenstra, A. H. G. Rinnooy Kan, (1980) Deterministic Production Planning: Algorithms and Complexity. Management Science 26(7):669-679. http://dx.doi.org/10.1287/mnsc.26.7.669

MANAGEMENT SCIENCE
Vol 26, No. 7, July 1980
Printed in U.S.A

DETERMINISTIC PRODUCTION PLANNING: ALGORITHMS AND COMPLEXITY*

M. FLORIAN,† J. K. LENSTRA‡ AND A. H. G. RINNOOY KAN§

A class of production planning problems is considered in which known demands have to be satisfied over a finite horizon at minimum total costs. For each period, production and storage cost functions are specified. The production costs may include set-up costs and the production levels may be subject to capacity limits. The computational complexity of the problems in this class is investigated. Several algorithms proposed for their solution are described and analyzed. It is also shown that some special cases are *NP*-hard and hence unlikely to be solvable in polynomial time.

(INVENTORY/PRODUCTION—DETERMINISTIC MODELS; INVENTORY/PRODUCTION—SURVEYS; DYNAMIC PROGRAMMING—DETERMINISTIC, DISCRETE TIME)

1. Introduction

We consider a class of production planning problems, in which a facility manufactures a single product to satisfy known integer demands over a finite planning horizon of n periods. For each period, production and storage cost functions are specified. The production cost functions may include set-up costs and the amount produced in each period may be subject to an integer capacity limit. The problem is that of determining the amounts to be produced in each period in order to supply each demand on time (no backlogging) and to minimize the total costs of production and storage.

In this paper we investigate the computational complexity of these problems for various types of cost functions, set-up costs and capacity limits. The Appendix provides a brief introduction to the relevant concepts of computational complexity theory; for more extensive expositions the reader is referred to [14], [18], [10].

In §2 we consider the standard dynamic programming approach for the most general problem in our class. Its running time is $O(R_n C_n)$ or $O(nR_n^2)$, where R_n is the total demand and C_n is the total capacity over the entire interval. This algorithm is pseudopolynomial in the sense that its running time is exponential in the problem size under a binary representation of the numerical data, but polynomial in the data themselves. We also establish *NP*-hardness for the problem, even for the special case in which all demands are equal, all storage costs are zero, and the production cost functions can be interpreted as being either concave with arbitrary capacity limits or convex with additional unit set-up costs. Hence, it is very unlikely that these restricted versions of the problem allow solution in truly polynomial time. These are, of course, worst-case results, and the average problem instance encountered may be solved quite quickly by enumerative methods.

In §3 we investigate problems with concave cost functions. We recall results of

*Accepted by Bennett L. Fox; received November 23, 1978. This paper has been with the authors 6 months for 2 revisions.

†Université de Montréal.
‡Mathematisch Centrum, Amsterdam.
§Erasmus University, Rotterdam.

0025-1909/80/2607/0669$01.25

Wagner and Whitin [24] and Florian and Klein [8], who characterized the structure of optimal production plans and presented polynomial algorithms for the special cases of infinite and equal capacities. These algorithms can be implemented to run in $O(n^2)$ and $O(n^4)$ time, respectively. Further, we discuss some enumerative methods for the case of arbitrary capacities, as possible alternatives to the general dynamic programming approach.

In §4 we turn to problems with convex cost functions. For the case that no additional set-up costs are specified, results of Johnson [12] and Veinott [23] yield a simple pseudopolynomial algorithm, which runs in $O(nR_n)$ time. However, the recent development of a polynomial algorithm for linear programming [15] implies that this case is solvable in strictly polynomial time.

The primary contribution of this paper lies in establishing the sharp borderline between well-solved and probably intractable deterministic production planning problems, as indicated by Table 1: our *NP*-hardness results exactly complement the polynomial algorithms.

TABLE 1

Summary of Complexity Results

arbitrary cost functions		capacity limits			convex cost functions
		infinite	equal	arbitrary	
set-up costs	zero	† (2)	† (2)	† (2)	* (4)
	equal	† (2)	† (2)	† (2)	† (2)
	arbitrary	† (2)	† (2)	† (2)	† (2)
concave cost functions		* (3)	* (3)	† (2)	

*solvable in polynomial time

†solvable in pseudopolynomial time, and *NP*-hard even for equal demands and zero storage costs

(x) see §x

2. General Case

We start by introducing some notation. For period i ($i = 1, \ldots, n$), let r_i be the demand, b_i the production set-up cost, c_i the production capacity limit and x_i the production amount, and let $R_i = \sum_{j=1}^{i} r_j$, $C_i = \sum_{j=1}^{i} c_j$ and $X_i = \sum_{j=1}^{i} x_j$, so that R_i, C_i and X_i are the cumulative demand, capacity and production level up to period i. The cost of producing an amount x_i in period i is given by $p_i(x_i)$, with $p_i(0) = 0$ and $p_i(x) = b_i + p_i'(x)$ for $x > 0$, where p_i' is a continuous and nondecreasing function with $p_i'(0) = 0$. The cost of storing an inventory $I_i = X_i - R_i$ from period i to period $i + 1$ is given by $h_i(I_i)$, where h_i is a continuous and nondecreasing function with $h_i(0) \geq 0$.

The production planning problem is that of determining amounts $x_1, \ldots, x_n$ that minimize the total costs of production and storage:

$$\sum_{i=1}^{n} (p_i(x_i) + h_i(I_i)), \tag{1}$$

subject to the conditions of satisfying each demand on time and observing the capacity limits:

$$I_i \geq 0 \qquad (i = 1, \ldots, n), \tag{2}$$

$$0 \leq x_i \leq c_i \qquad (i = 1, \ldots, n). \tag{3}$$

A positive initial inventory I_0 could be handled by appending a period 0 in which $x_0 = I_0$ is produced at suitably small costs. The final inventory I_n can be assumed to be equal to zero without loss of generality [24]; hence, we also require that

$$I_n = 0. \tag{4}$$

Note that the inventory constraints correspond to flow conservation equations on an appropriately defined network.

Feasibility is assured by the assumption that $R_i \leqslant C_i$ $(i = 1, \ldots, n)$. Under the assumptions that all r_i and c_i are integers and that all p_i' and h_i are linear functions between successive integer values of the argument, we may restrict our attention to integer valued $x_1, \ldots, x_n$. In analyzing the complexity of problems and algorithms, we assume that all p_i and h_i can be evaluated in unit time for any value of the argument.

The standard way to solve problems of this type is by means of *dynamic programming*. Let $D_i(X)$ be the cost of an optimal production plan over periods $1, \ldots, i$ subject to $X_i = X$, let $\mathfrak{X}_i$ be the set of feasible cumulative production levels in period i, and let $\mathfrak{X}_i(X)$ be the set of feasible production amounts in period i subject to $X_i = X$. It is clear that

$$D_0(X) = \begin{cases} 0 & (X = 0), \\ \infty & (X \neq 0), \end{cases}$$

$$D_i(X) = \left\{ \begin{array}{ll} \min_{x \in \mathfrak{X}_i(X)} \{ D_{i-1}(X - x) + p_i(x) \} + h_i(X - R_i) & (X \in \mathfrak{X}_i) \\ \infty & (X \notin \mathfrak{X}_i) \end{array} \right\} \quad (i \geqslant 1). \tag{5}$$

The cost of an optimal production plan over the entire interval is equal to $D_n(R_n)$, which is calculated according to the above forward recursion; the corresponding values of $x_1, \ldots, x_n$ are obtained by standard backtracing techniques. A backward recursion could be formulated just as easily.

To estimate the running time of the dynamic programming algorithm, we note that $\mathfrak{X}_i \subseteq \{R_i, R_i + 1, \ldots, R_n\}$ and $\mathfrak{X}_i(X) \subseteq \{0, 1, \ldots, c_i\}$. Hence, for fixed i, all $D_i(X)$ are determined in $O(R_n c_i)$ time. It follows that the complete recursion requires $O(R_n C_n)$ time. If no capacity limits are specified, we have $\mathfrak{X}_i(X) \subseteq \{0, 1, \ldots, R_n - R_{i-1}\}$, and an $O(nR_n^2)$ running time results.

Thus, dynamic programming provides a *pseudopolynomial* algorithm. We shall now present strong circumstantial evidence that a truly *polynomial* algorithm for several special cases of the production planning problem will probably never be found.

PROPOSITION 1. *The production planning problem with equal demands and zero storage costs is NP-hard.*

PROOF. We have to show that some known *NP*-complete problem is reducible to the simplified production planning problem. Our starting point will be the following *NP*-complete problem [13]:

KNAPSACK: Given positive integers $a_1, \ldots, a_t$, A, does there exist a subset $S \subset T = \{1, \ldots, t\}$ such that $\sum_{i \in S} a_i = A$?

Given any instance of KNAPSACK, we define the following instance of the production planning problem, where, for notational convenience, the periods are

numbered from 0 to n:

$$n = t;$$
$$r_i = A \qquad (i = 0, \ldots, t);$$
$$b_0 = 1,\ c_0 = tA,\ p_0(0) = 0,\ p_0(x) = 1 \qquad (0 < x \leqslant c_0);$$
$$b_i = 1,\ c_i = a_i,\ p_i(0) = 0,\ p_i(x) = 1 + \frac{a_i - 1}{a_i}x \quad (0 < x \leqslant c_i) \quad (i = 1, \ldots, t);$$
$$h_i(I) = 0 \qquad (I \geqslant 0) \qquad (i = 0, \ldots, t).$$

The production cost functions are illustrated in Figure 1. We claim that KNAPSACK has a solution if and only if there exists a feasible production plan with total costs at most equal to $A + 1$.

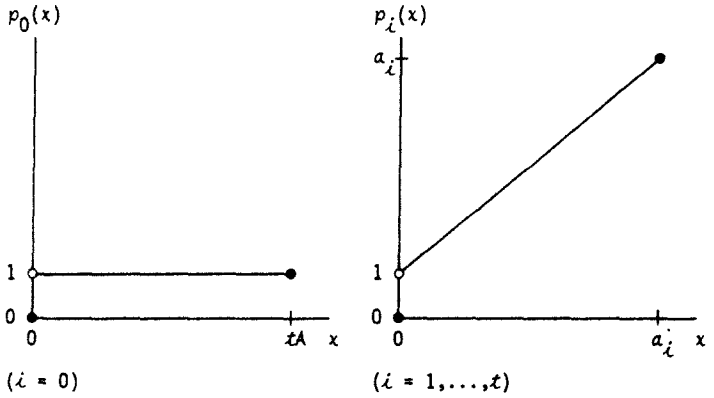

FIGURE 1. The Production Cost Functions in the Reduction.

Since $x_0 > 0$ in any feasible plan and $p_0(x)$ is constant for $0 < x \leqslant tA$, we may assume that the production in period 0 is at capacity and supplies the demands in periods $0, \ldots, t-1$. The production in periods $1, \ldots, t$ has to supply the demand in period t only. Therefore, we may restrict our attention to production plans defined by

$$x_0 = tA, \quad \sum_{i \in T} x_i = A, \quad 0 \leqslant x_i \leqslant a_i \quad (i \in T).$$

Since for all $i \in T$

$$p_i(x) = x \quad \text{for } x = 0 \text{ and } x = a_i,$$
$$p_i(x) > x \quad \text{for } 0 < x < a_i,$$

the total costs of such a plan are at least equal to $A + 1$:

$$\sum_{i=0}^{t} p_i(x_i) = 1 + \sum_{i \in T} p_i(x_i) > 1 + \sum_{i \in T} x_i = A + 1.$$

Moreover, they are exactly equal to $A + 1$ if and only if $x_i \in \{0, a_i\}$ for all $i \in T$, i.e., if and only if there exists a subset $S \subset T$ such that $\sum_{i \in S} a_i = A$. This establishes the desired result. Q.E.D.

PROPOSITION 2. *The production planning problem with equal demands and zero storage costs remains NP-hard in each of the following cases*:

(a) *arbitrary cost functions, no set-up costs, no capacity limits*;

(b) *concave cost functions, no set-up costs, arbitrary capacity limits*;

(c) *convex cost functions, unit set-up costs, no capacity limits*.

PROOF. The production cost functions involved in the above *NP*-hardness proof are both concave and convex, with additional unit set-up costs and arbitrary capacity limits. It is easily seen that concave cost functions can be adapted to incorporate set-up costs in such a way that they are still linear between successive integer points. Similarly, convex cost functions can be adapted to effectuate capacity limits by making the costs sufficiently high beyond these limits. Q.E.D.

Proposition 2 leaves only a few possibilities for truly polynomial algorithms. These will be considered in §§3 and 4.

3. Concave Costs

Let us assume that all p_t and h_i are concave functions, possibly including set-up costs. In this section, we first review the known polynomial algorithms for special cases of this problem, and then discuss solution methods for the general case, which is *NP*-hard.

Wagner and Whitin [24] studied the case in which no capacity limits are specified; their characterization of the structure of optimal production plans immediately yielded an $O(n^2)$ algorithm. Florian and Klein [8] characterized the structure of optimal production plans for the case of arbitrary capacities and obtained an $O(n^4)$ algorithm for the special case of equal capacities. These results are summarized below. It should also be mentioned that for the related problem in which upper bounds on inventory rather than on production are specified, Love [19] developed an $O(n^3)$ algorithm.

Recall the formulation of the problem in §2. The constraints (2)–(4) define a closed bounded convex set. The objective function (1) is concave and hence its minimum value is achieved at one of the extreme points of this set. The special structure of the set allows a simple characterization of the production plans corresponding to its extreme points. It was shown in [8] that such plans consist of a sequence of subplans in which

(a) the inventory is strictly positive in every period, except the last, where it is zero, and

(b) the production is either zero or at full capacity in every period, except for at most one period, which will be called the *fractional period*.

Thus, we are led to consider $\frac{1}{2}n(n+1)$ subproblems P_{lm} $(0 \leq l < m \leq n)$ in which our objective is to minimize

$$\sum_{i=l+1}^{m} (p_i(x_i) + h_i(I_i))$$

subject to

$$I_l = I_m = 0,$$

$$I_i > 0 \qquad (i = l+1, \ldots, m-1),$$

$$0 \leq x_i \leq c_i \qquad (i = l+1, \ldots, m),$$

$$0 < x_f < c_f \quad \text{for at most one } f \qquad (l+1 \leq f \leq m).$$

Let E_{lm} be the optimal solution value to problem P_{lm} and D_m^* the cost of an optimal production plan over periods $1, \ldots, m$. Given $\frac{1}{2}n(n+1)$ values E_{lm} $(0 \leqslant l < m \leqslant n)$, we solve the original problem by calculating D_n^* as follows:

$$D_0^* = 0,$$
$$D_m^* = \min_{0 \leqslant l < m} \{ D_l^* + E_{lm} \} \qquad (m = 1, \ldots, n).$$

This recursion can be carried out in $O(n^2)$ time.

In the case that $c_i = \infty$ $(i = 1, \ldots, n)$, an optimal solution to problem P_{lm} is clearly given by $x_{l+1} = R_m - R_l$, $x_i = 0$ $(i = l+2, \ldots, m)$, so that

$$E_{lm} = p_{l+1}(R_m - R_l) + \sum_{i=l+1}^{m} h_i(R_m - R_i)$$

(cf. [24]). All E_{lm} can be determined in $O(n^2)$ time. It follows that the original problem is solved in $O(n^2)$ time.

In the case that $c_i = c$ $(i = 1, \ldots, n)$, problem P_{lm} can be solved in the following way (cf. [8]). For notational convenience, we assume that $l = 0$. Dividing total demand by the capacity, we find $R_m = kc + \epsilon$, where k is the number of periods in which the production will be at capacity and ϵ $(0 \leqslant \epsilon < c)$ is the amount to be produced in the fractional period. In order to apply the dynamic programming recursion (5), we observe that the sets $\mathcal{X}_i$ of feasible cumulative production levels in period i are given by

$$\mathcal{X}_i = \{ X \mid X \in \{0, \epsilon, c, c+\epsilon, \ldots, kc, kc+\epsilon\}, \quad R_i < X \leqslant ic \} \qquad (i = 1, \ldots, m-1),$$
$$\mathcal{X}_m = \{ R_m \},$$

and the sets $\mathcal{K}_i(X)$ of feasible production amounts in period i subject to $X_i = X$ by

$$\mathcal{K}_i(jc) = \{0, c\}, \quad \mathcal{K}_i(jc+\epsilon) = \{0, \epsilon, c\} \quad (j = 0, \ldots, k; \quad i = 1, \ldots, m).$$

We solve P_{0m} by calculating $E_{0m} = D_m(R_m)$.

With respect to the running time of the algorithm, we note that $|\mathcal{X}_i| = O(i)$ and $|\mathcal{K}_i(X)| \leqslant 3$ $(X \in \mathcal{X}_i)$. Hence, for fixed i, all $D_i(X)$ are determined in $O(i)$ time. It follows that E_{0m} is found in $O(m^2)$ time and that the original problem is solved in $O(n^4)$ time.

In the case that the c_i need not be equal, the problem is NP-hard. The general dynamic programming approach solves the problem in $O(R_n C_n)$ time. The question arises, however, if the available information on the structure of optimal production plans is useful in deriving an efficient algorithm. We can solve problem P_{0m} by successively fixing the period f that is allowed to be fractional. Let $P_{0m}^{(f)}$ denote the subproblem under the additional restriction that $x_i \in \{0, c_i\}$ $(i \neq f)$, and let $E_{0m}^{(f)}$ be the optimal solution value to this problem. The sets of feasible cumulative production levels for $P_{0m}^{(f)}$ satisfy the following restrictions:

$$\mathcal{X}_0 = \{0\},$$
$$\mathcal{X}_i = \{ X \mid X \in \{X', X' + c_i \mid X' \in \mathcal{X}_{i-1}\},$$
$$\max\{R_i + 1, R_m - (C_m - C_i)\} \leqslant X \leqslant R_m \} \qquad (i = 1, \ldots, f-1),$$
$$\mathcal{X}_i = \{ X \mid X \in \{X', X' - c_{i+1} \mid X' \in \mathcal{X}_{i+1}\}, \quad R_i + 1 \leqslant X \leqslant C_i \} \qquad (i = f, \ldots, m-1),$$
$$\mathcal{X}_m = \{ R_m \}.$$

Thus, we generate the sets $\mathcal{X}_0, \mathcal{X}_1, \ldots, \mathcal{X}_{f-1}$ according to a forward recursion and the sets $\mathcal{X}_m, \mathcal{X}_{m-1}, \ldots, \mathcal{X}_f$ according to a backward recursion. Considering $\mathcal{X}_{f-1} \times \mathcal{X}_f$ we need only retain those pairs (X_{f-1}, X_f) for which $0 \leqslant X_f - X_{f-1} \leqslant c_f$. Therefore, we may subsequently carry out a backward and a forward recursion to reduce the size of $\mathcal{X}_i$ for $i < f-1$ and $i > f$ respectively even further. Defining the sets of feasible production amounts by

$$\mathcal{X}_i(X) = \{0, c_i\} \quad (i = 1, \ldots, f-1, f+1, \ldots, m), \qquad \mathcal{X}_f(X) = \{0, 1, \ldots, c_f\},$$

we solve $P_{0m}^{(f)}$ by calculating $E_{0m}^{(f)} = D_m(R_m)$ according to the recursion (5).

With respect to the running time, we note that $|\mathcal{X}_i| \leqslant R_m$, $|\mathcal{X}_i(X)| = 2$ $(i \neq f)$ and $|\mathcal{X}_f| = c_f + 1$. Hence, $E_{0m}^{(f)}$ is determined in $O(mR_m + R_m c_f)$ time. It follows that $E_{0m} = \min_{1 \leqslant f \leqslant m} \{E_{0m}^{(f)}\}$ is found in $O(m^2 R_m + R_m C_m)$ time and that the original problem is solved in $O(n^4 R_n + n^2 R_n C_n)$ time. In terms of worst-case running time, this approach is inferior to standard dynamic programming.

This analysis has no implications for the relative efficiency of both algorithms in an average-case sense. Computational experiments have indicated that, when the average demand is equal to 20, our approach is ten times faster than the standard approach on test problems with up to 12 periods and twice as fast on problems with 24 periods; when the average demand is equal to 200, the differences are even more impressive, as was to be expected. Further speed-up may be possible by applying "reaching" techniques as in [6].

Under certain circumstances, pseudopolynomial algorithms may be outperformed by tree search methods which have an even more forbidding worst-case behavior. For example, the KNAPSACK problem of § 2 can be solved by dynamic programming in $O(tA)$ time [2], but for large values of A branch-and-bound tends to be more efficient [20].

For the production planning problem with concave costs and arbitrary capacities, several tree search methods have been suggested. We mention the work of Chen [4] and Lambrecht and Vander Eecken [16] and, in particular, the algorithm recently proposed by Baker, Dixon, Magazine and Silver [1]. The latter authors considered the special case in which $p_i'(x) = \bar{p}x$, $h_i(I) = \bar{h}I$ $(i = 1, \ldots, n)$. (This problem is NP-hard, even for $\bar{p} = \bar{h} = 0$, as can be proved by a slight modification of the proof of Proposition 1.) They found that for each subproblem P_{lm} only the first period can be fractional, so that $E_{lm} = E_{lm}^{(l+1)}$. It follows that in this case the above dynamic programming approach requires only $O(n^3 R_n + nR_n C_n)$ time.

Computational experiments have shown that our approach and the algorithm of Baker *et al.* require roughly the same amount of time on test problems with up to 24 periods.

4. Convex Costs

Let us now assume that all p_i and h_i are convex functions. Thus, they do not include set-up costs, but they can easily be adapted to enforce capacity limits.

Veinott [23] has shown that an optimal production plan can be obtained by satisfying each unit of demand in turn as cheaply as possible. An algorithm based on this rule has $O(nR_n)$ running time. It generalizes an algorithm due to Johnson [12] for the case of linear storage costs; Johnson's method essentially yields an initial solution to a linear transportation problem, which also happens to be optimal. As pointed out earlier, the problem can be formulated as a minimum convex cost network flow

problem, and Veinott's work can also be interpreted as the use of the "incremental" algorithm [3; 11] for solving such problems.

Examples can be constructed to show that the above rule cannot be stretched to allocate more than one unit of demand at a time. However, the problem is not harder than linear programming [21], and therefore Khachian's recent polynomial algorithm for linear programming [15] provides at least a formal proof that the problem is solvable in strictly polynomial time. For practical purposes, the Johnson-Veinott rule will rightly remain to be considered the most attractive approach.

In the case that additional set-up costs b_i are specified, the problem is *NP*-hard, even if $b_i = 1$ $(i = 1, \ldots, n)$. As pointed out in § 3, a branch-and-bound approach may offer a practical alternative to the general dynamic programming recursion. Such an approach could be based on a fixed charge network flow formulation (cf. [22]).

5. Summary

We have analyzed the computational complexity of a class of deterministic production planning problem for various types of cost functions, set-up costs and capacity limits. We hope that the results, as summarized in Table 1, have once again demonstrated the usefulness of arguments from complexity theory, when potential algorithmic improvements for a combinatorial optimization problem are being considered.

Appendix. A Brief Introduction to Computational Complexity Theory

The inherent computational complexity of a combinatorial problem obviously has to be related to the computational behavior of algorithms designed for its solution. This behavior is usually measured by the *running time* of the algorithm (i.e., the number of elementary operations such as additions and comparisons) as related to the *size* of the problem (i.e., the number of bits occupied by the data).

If a problem of size n can be solved by an algorithm with running time $O(p(n))$[1] where p is a *polynomial* function, then the algorithm may be called "good" and the problem "well solved." These notions were introduced by Edmonds [7] in the context of the matching problem; his algorithm can be implemented to run in $O(n^3)$ time on graphs with n vertices. Polynomial algorithms have been developed for a wide variety of combinatorial optimization problems [17]. On the other hand, many such problems can only be solved by enumerative methods which may require *exponential* time.

When encountering a combinatorial problem, one would naturally like to know if a polynomial algorithm exists or if, on the contrary, any solution method must require exponential time in the worst case. Results of the latter type are still rare, but it is often possible to show that the existence of a polynomial algorithm is at the very least extremely unlikely. One may arrive at such a result by proving that the problem in question is *NP-complete* [5; 13]. According to the formal definition given below, the *NP*-complete problems are equivalent in the sense that none of them has been well solved and that, if one of them would be well solved, then the same would be true for all of them. Since all the classical problems that are notorious for their computational intractability, such as traveling salesman, job shop scheduling and integer program-

[1] The notation "$q(n) = O(p(n))$" means that there exists a constant $c > 0$ such that $|q(n)| \leq c \cdot p(n)$ for all $n > 0$.

ming problems, are known to be *NP*-complete, the polynomial-time solution of such a problem would be very surprising indeed. For practical purposes, this implies that in solving those problems one may just as well accept the inevitability of a bad (superpolynomial) *optimization* algorithm or resort to using a good (polynomial) *approximation* algorithm.

The theory of *NP*-completeness deals primarily with *recognition problems*, which require a yes/no answer. An example of a recognition problem is the following:

KNAPSACK:

instance: a set $T = \{1, \ldots, t\}$, positive integers $a_1, \ldots, a_t, A$;

question: does there exist a subset $S \subset T$ such that $\sum_{i \in S} a_i = A$?

An instance of a recognition problem is *feasible* if the question can be answered affirmatively. Feasibility is usually equivalent to the existence of an associated *structure* which satisfies a certain property.

A recognition problem belongs to the class $\mathcal{P}$ if, for any instance of the problem, its feasibility or infeasibility can be determined by a polynomial algorithm. It belongs to the class $\mathcal{NP}$ if, for any instance, one can determine in polynomial time whether a given structure affirms its feasibility. For example, KNAPSACK is a member of $\mathcal{NP}$, since for any $S \subset T$ one can test whether $\sum_{i \in S} a_i = A$ in $O(t)$ time. It is obvious that $\mathcal{P} \subseteq \mathcal{NP}$.

Problem P' is said to be *reducible* to problem P (notation: $P' \propto P$) if for any instance of P' an instance of P can be constructed in polynomial time such that solving the instance of P will solve the instance of P' as well. Informally, the reducibility of P' to P implies that P' can be considered as a special case of P, so that P is at least as hard as P'.

P is called *NP-hard* if $P' \propto P$ for every $P' \in \mathcal{NP}$. In that case, P is at least as hard as any problem in $\mathcal{NP}$. P is called *NP-complete* if P is *NP*-hard and $P \in \mathcal{NP}$. Thus, the *NP*-complete problems are the most difficult problems in $\mathcal{NP}$.

A polynomial algorithm for an *NP*-complete problem P could be used to solve all problems in $\mathcal{NP}$ in polynomial time, since for any instance of such a problem the construction of the corresponding instance of P and its solution can be both effected in polynomial time. We note the following two important consequences.

(i) It is very unlikely that $\mathcal{P} = \mathcal{NP}$, since $\mathcal{NP}$ contains many notorious combinatorial problems, for which in spite of a considerable research effort no polynomial algorithms have been found so far.

(ii) It is very unlikely that $P \in \mathcal{P}$ for any *NP*-complete P, since this would imply that $\mathcal{P} = \mathcal{NP}$ by the earlier argument.

The first *NP*-completeness result is due to Cook [5]. He designed a "master reduction" to prove that every problem in $\mathcal{NP}$ is reducible to the so-called SATISFIABILITY problem. Starting from this result, Karp [13] and many others [10] identified a large number of *NP*-complete problems in the following way. One can establish *NP*-completeness of some $P \in \mathcal{NP}$ by specifying a reduction $P' \propto P$ with P' already known to be *NP*-complete: for every $P'' \in \mathcal{NP}$, $P'' \propto P'$ and $P' \propto P$ then imply that $P'' \propto P$ as well.

As far as *optimization problems* are concerned, one usually reformulates, say, a minimization problem as a recognition problem by asking for the existence of a feasible solution with value at most equal to a given threshold (cf. the proof of Proposition 1). When this recognition problem can be proved to be *NP-complete*, the corresponding optimization problem might be called *NP-hard* in the sense that the existence of a polynomial algorithm for its solution would imply that $\mathcal{P} = \mathcal{NP}$.

Let us finally examine the KNAPSACK problem in more detail. The problem size is $O(t \log A)$ when the numerical data are represented in a reasonable way, e.g., in a binary or decimal encoding. KNAPSACK has been proved to be *NP*-complete [13]. However, it can be solved by dynamic programming in $O(tA)$ time [2]. This algorithm is *exponential* in the problem size, but it might be called *pseudopolynomial* [9] in the sense that it is polynomial in the data themselves. A similar combination of *NP*-hardness and solvability in pseudopolynomial time occurs in the case of the production planning problem. For other problems, which are apparently even more difficult, the existence of a pseudopolynomial algorithm can be excluded (unless $\mathcal{P} = \mathcal{NP}$) by proving that they are *strongly* [9] or *unary* [18] *NP-complete*.[2]

[2]The authors are grateful to M. L. Fisher and B. J. Lageweg for their useful comments and to J. Hooft and E. Blanc for their assistance in the computational experiments. This research was partially supported by the National Research Council of Canada and by NATO Special Research Grant 9.2.02 (SRG.7).

References

1. Baker, K. R., Dixon, P., Magazine, M. J. and Silver, E. A., "An Algorithm for the Dynamic Lot-Size Problem with Time-Varying Production Capacity Constraints," *Management Sci.*, Vol. 24 (1978), pp. 1710–1720.
2. Bellman, R. E. and Dreyfus, S. E., *Applied Dynamic Programming*, Princeton Unlv. Press, Princeton, N.J., 1962.
3. Busacker, R. G. and Gowen, P. J., "A Procedure for Determining a Family of Minimal-Cost Network Flow Patterns," Technical Paper 15, Operations Research Office, John Hopkins Univ., 1961.
4. Chen, K.-Y., "A Network Approach to the Capacitated Lot-Size Problem," unpublished manuscript, The Wharton School, University of Pennsylvania, Philadelphia, 1976.
5. Cook, S. A., "The Complexity of Theorem-Proving Procedures," *Proc. 3rd Annual ACM Symp. Theory of Computing*, 1971, pp. 151–158.
6. Denardo, E. V. and Fox, B. L., "Shortest-Route Methods: 1. Reaching, Pruning, and Buckets," *Operations Res.*, Vol. 27 (1979), pp. 161–186.
7. Edmonds, J., "Paths, Trees, and Flowers," *Canad. J. Math.*, Vol. 17 (1965), pp. 449–467.
8. Florian, M. and Klein, M., "Deterministic Production Planning with Concave Costs and Capacity Constraints," *Management Sci.*, Vol. 18 (1971), pp. 12–20.
9. Garey, M. R. and Johnson, D. S., " "Strong" NP-Completeness Results: Motivation, Examples and Implications," *J. Assoc. Comput. Mach.*, Vol. 25 (1978), pp. 499–508.
10. ——— and ———, *Computers and Intractability: a Guide to the Theory of NP-Completeness*, Freeman, San Francisco, Calif., 1979.
11. Hu, T. C., "Minimum-Cost Flows in Convex-Cost Networks," *Naval Res. Logist. Quart.*, Vol. 13 (1969), pp. 1–9.
12. Johnson, S. M., "Sequential Production Planning over Time at Minimum Cost," *Management Sci.*, Vol. 3 (1957), pp. 435–437.
13. Karp, R. M., "Reducibility among Combinatorial Problems," in: R. E. Miller, J. W. Thatcher (eds.), *Complexity of Computer Computations*, Plenum Press, New York, 1972, pp. 85–103.
14. ———, "On the Computational Complexity of Combinatorial Problems," *Networks*, Vol. 5 (1975), pp. 45–68.
15. Khachian, L. G., "A Polynomial Algorithm in Linear Programming," *Soviet Math. Dokl.*, Vol. 20 (1979), pp. 191–194.
16. Lambrecht, M. and Vander Eecken, J., "A Capacity Constrained Single-Facility Dynamic Lot-Size Model," *Eur. J. Operational Res.*, Vol. 2 (1978), pp. 132–136.
17. Lawler, E. L., *Combinatorial Optimization: Networks and Matroids*, Holt, Rinehart and Winston, New York, 1976.
18. Lenstra, J. K. and Rinnooy Kan, A. H. G., "Computational Complexity of Discrete Optimization Problems," *Ann. Discrete Math.*, Vol. 4 (1979), pp. 121–140.

19. LOVE, S. F., "Bounded Production and Inventory Models with Piecewise Concave Costs," *Management Sci.*, Vol. 20 (1973), pp. 313–319.
20. MARTELLO, S. AND TOTH, P., "An Upper Bound for the Zero-One Knapsack Problem and a Branch and Bound Algorithm," *Eur. J. Operational Res.*, Vol. 1 (1977), pp. 169–175.
21. MEYER, R. R., "A Class of Nonlinear Integer Programs Solvable by a Single Linear Program," *SIAM J. Control Optimization*, Vol. 15 (1977), pp. 935–946.
22. RECH, P. AND BARTON, L. G., "A Non-Convex Transportation Algorithm," in: E. M. L. Beale (ed.) *Applications of Mathematical Programming Techniques*, American Elsevier, New York, 1970, pp. 250–260.
23. VEINOTT, A. F., JR., "Production Planning with Convex Costs: A Parametric Study," *Management Sci.*, Vol. 10 (1964), pp. 441–460.
24. WAGNER, H. M. AND WHITIN, T. M., "Dynamic Version of the Economic Lot Size Model," *Management Sci.*, Vol. 5 (1958), pp. 89–96.
25. ZANGWILL, W. I., "A Deterministic Multiperiod Production Scheduling Model with Backlogging," *Management Sci.*, Vol. 13 (1966), pp. 105–119.