JOURNAL OF SCHEDULING
J. Sched. **1**, 3–14 (1998)

THE MYSTICAL POWER OF TWONESS: IN MEMORIAM EUGENE L. LAWLER

JAN KAREL LENSTRA*

Eindhoven University of Technology; CWI, Amsterdam

ABSTRACT

This paper reviews the work of Eugene L. Lawler, one of the early investigators of combinatorial optimization and the architect of deterministic machine scheduling theory. After sketching his research career and examining the development of his interests, I describe some of his seminal results in scheduling theory and recall one of his contributions to scientific journalism. © 1998 John Wiley & Sons, Ltd.

KEY WORDS: combinatorial optimization; machine scheduling; history

1. INTRODUCTION

Eugene L. Lawler (1933–1994) was one of the earliest researchers who concentrated on combinatorial optimization as a field of investigation. His textbook on networks and matroids is a classic, and several of his papers became benchmark references. His research in sequencing and scheduling was instrumental in stimulating and unifying an area that, prior to his work, was a rather unsystematic hodgepodge. He also made significant contributions to algorithmic graph theory, complexity theory, and computational biology. In addition, he was a phenomenal expositor, and he was the social conscience of the Computer Science Division at Berkeley.

In this paper I attempt to review his work, with an emphasis on his research in deterministic machine scheduling. The paper is based on a lecture given at the 16th International Symposium on Mathematical Programming in Lausanne on 25 August 1997. It supplements the personal reminiscences that he wrote in 1991 [1] and an obituary that appeared in 1994 in *Optima*, the newsletter of the Mathematical Programming Society [2]. A selection of his publications [3] and two issues of *Mathematical Programming* containing 14 papers dedicated to his memory [4] will appear shortly.

2. AN OVERVIEW IN SIX DIMENSIONS

Gene Lawler went to college in Tallahassee, Florida. He began graduate study at Harvard in 1954. After interruptions by law school, the army and employment at a real job shop, he obtained his

* Correspondence to: Jan Karel Lenstra, Department of Mathematics and Computing Science, Eindhoven University of Technology, P.O. Box. 513, 5600 MB Eindhoven, The Netherlands. E-mail: jkl@win.tue.nl

Contract/grant sponsor: NATO; Contract/grant number: CRG 971550

CCC 1094–6136/98/010003–12$17.50

Ph.D. in Applied Mathematics in 1962. He then joined the faculty of Electrical Engineering at the University of Michigan in Ann Arbor. In 1970 he moved to the EECS Department at the University of California in Berkeley. He died on 2 September 1994.

Plate 1 is a map of his scientific life. *Time* runs vertically. His *career* is represented by the vertical bar at the left-hand side. Plate 1(a) lists his publications, with algorithmic or analytical *approaches* indicated by capitals and *problem areas* by colours. Plate 1(b) classifies the publications according to *style*. Plate 1(c) gives a further partitioning of the papers on *scheduling problems*.

Some caveats are in order. I have applied a fair amount of judgemental rounding in classifying the publications, especially in deciding about thé algorithmic approach taken. Several D papers also deal with B or E, and the last scheduling paper is not only A but also E. I have included only final publications and ^deleted prepublications as well as translations (into Czech, Dutch, Hungarian, and Japanese).

Plate 1(a) shows that Gene, after his early work in switching theory, became interested in combinatorial optimization. Most of his work is concerned with networks and matroids or sequencing and scheduling, perhaps with a gradual shift from the former area to the latter. In the 1990s, he turned to combinatorial problems in the new field of computational biology.

The algorithmic approaches also show a clear picture. He started by studying enumerative methods, first branch-and-bound and then dynamic programming. Under the influence of complexity theory, he became interested in polynomial-time optimization and, later and to a lesser extent, in performance guarantees for polynomial-time approximation algorithms. I will return to this development in Section 3.

Plate 1(b) highlights two books: his textbook on networks and matroids [5] and an edited collection of chapters on the travelling salesman problem [6]. Especially the first book had a pronounced impact and is as useful today as it was in 1976. He contributed a number of influential surveys, most notably on branch-and-bound (see Section 3), well-solvable cases of the TSP [7], and scheduling [8–10]. His lucid writing style and innate wit made him a superb scientific journalist. The Sputnik paper [11] will serve as an example in Section 6.

The classification given in Plate 1(c) will be discussed in Section 5.

3. EARLY WORK

I will explore the development of Gene's research interests by examining nine publications from the period 1962–1975.

E. L. Lawler. *Some aspects of discrete mathematical programming.* Ph.D. thesis, August 1962; Report BL-31, Computation Laboratory, Harvard University, Cambridge, MA (1963). —Gene's thesis deals with integer programming. He applies local search *avant la lettre* and branch-and-bound in the footsteps of Willard Eastman. To my knowledge, Eastman's thesis of 1958 [12] is the earliest publication on branch-and-bound in the operations research literature, predating the celebrated paper by Little, Murty, Sweeney and Karel by five years. The majority of Gene's work in switching theory is also concerned with enumerative methods based on tree search.

E. L. Lawler. The quadratic assignment problem. *Management Science 9* (1963), 586–599. —Gene's first paper in combinatorial optimization is still an important reference. Some later papers on special cases of the quadratic assignment problem propose lower bounds that are dominated by the bound developed here.

E. L. Lawler, D. E. Wood. Branch-and-bound methods: a survey. *Operations Research 14* (1966), 699–719. —This is a great survey. It unifies approaches that originated in distinct areas. The LP-based integer programming methods of Land, Doig and Dakin, Balas' additive algorithm for the same problem, the synthesis and covering algorithms for switching circuits proposed by Roth, Karp and Lawler, the game trees from artificial intelligence, and the backtrack methods from combinatorics are all put in the branch-and-bound framework. The paper became a citation classic [13].

At this point, Gene had paid his dues to branch-and-bound. His research was moving to more structured forms of enumeration. I remember that, at Bernard Roy's 1974 Versailles Summer School on Combinatorial Programming, where we first met, he tallied the papers presented and commented that there was too much branch-and-bound to his liking. It is not that he came to dislike the approach. He admired his younger colleagues who turned it from an art into a science and who used Lagrangian relaxation and polyhedral combinatorics in replacing branch-and-bound by continue-to-bound-and-avoid-to-branch. But he followed his own gifts and redirected his interests to pseudopolynomial and truly polynomial algorithms.

E. L. Lawler, J. M. Moore. A functional equation and its application to resource allocation and sequencing problems. *Management Science 16* (1969), 77–84. —Gene became the wizard of finite-state dynamic programming recursions. Among his many papers on the approach, the most salient ones are the Lawler–Moore paper and his 1977 paper on a pseudopolynomial-time algorithm for minimizing total tardiness on a single machine [14].

E. L. Lawler. Optimal cycles in doubly weighted directed linear graphs. In: P. Rosenstiehl (editor). *Theory of Graphs, International Symposium.* Gordon and Breach, New York, NY (1967), pp. 209–213.

E. L. Lawler. A solvable case of the traveling salesman problem. *Mathematical Programming 1* (1971), 267–269.

E. L. Lawler. Optimal sequencing of a single machine subject to precedence constraints. *Management Science 19* (1973), 544–546. —Three short papers presenting polynomial-time solutions for the ratio cycle problem in graphs, a special case of the TSP, and a minmax scheduling problem. Gene's interest in well-solvable TSP's led to his co-authorship of Chapter 4 of the TSP book [7], which spawned much research on the subject. The scheduling paper will be discussed in Section 5.

E. L. Lawler. Matroid intersection algorithms. *Mathematical Programming 9* (1975), 31–56. —As Gene has commented elsewhere [1], his work on matroid optimization was inspired by the insights of Jack Edmonds. It culminated in this landmark paper, which gave a polynomial-time algorithm for finding an optimum intersection of two matroids. It is undoubtedly the most important 'mystical 2' in his work. Gene was not the only one who found out that often 2 is easy while 3 seems undoable.

R. M. Karp. Reducibility among combinatorial problems. In: R. E. Miller, J. W. Thatcher (editors). *Complexity of Computer Computations.* Plenum Press, New York, NY (1972), pp. 85–103. —An explanation was provided by Dick Karp's famous paper on computational complexity, in which he established NP-completeness of 21 fundamental problems. Dick attributes two reductions to Gene: from Vertex Cover to Directed Hamiltonian Circuit and from Exact Cover to 3-Dimensional Matching. The latter result gives another mystical 2: 2-dimensional (or bipartite) matching is easy but 3-dimensional matching is hard. Gene would invariably comment that this is why a world with two sexes has been devised.

A quite different mystical 2 in his life was his habit of having second thoughts about the topic of his talks. His preferred subject was what he had been working on the night before. At the 1974 Versailles conference he was asked to talk about the quadratic assignment problem. He suggested complexity but was not allowed any second thoughts. He then gave, after hours, an illicit talk about NP-completeness.

Eventually, Gene was not primarily interested in what makes a problem hard but in what makes it well solvable, or approximable within a decent bound. In other words, he was concerned with the mystical 2 rather than the magical 3. At the same time, he knew the limitations of worst-case guarantees on running time or solution quality. Probabilistic analysis may not have been his favourite approach, but he was fully convinced of the importance and computational power of superpolynomial techniques such as branch-and-bound and local search.

4. SCHEDULING THEORY IS SOMETHING OF A JUNGLE

There is a bewildering variety of problem types in deterministic machine scheduling theory. The investigation of their complexity status is not as simple as '2 is easy, 3 is hard', and may surprise and confuse the analyst. Let me give an example.

A single machine needs to process a finite set N of jobs. The machine is available from time 0 onwards and can process at most one job at a time. Each job $j \in N$ requires processing during an uninterrupted period of length p_j; it is convenient to write $P = \sum_{j\in N} p_j$. We wish to find a schedule of minimum length. This is trivial: any schedule without machine idle time has length P and is optimal. When we introduce a second machine, the problem becomes NP-hard: a schedule of length $P/2$ exists if and only if the set of processing times can be split into two subsets of equal sum. We now allow job preemption; a job may be split but cannot be processed by both machines at the same time. The problem becomes easy again: if there is a job that is longer than the others together, it gets a machine on its own and the other jobs go on the second machine; otherwise, we schedule the jobs in arbitrary order on one machine, at time $P/2$ we interrupt the job that the machine is working on, we start its remainder at time 0 on the second machine, and continue with the jobs that are left. Returning to the single-machine problem, we now introduce job release dates and stipulate that a job cannot start before it is released. The problem remains easy: just schedule the jobs in order of nondecreasing release dates. We then add job deadlines, by which the jobs must be finished. A simple application of the equal-split argument tells us that the problem has become NP-hard.

We discard release dates and deadlines but change the objective. We wish to find a schedule that minimizes the sum of the job completion times, not their maximum. This problem is solved by the SPT rule: put the jobs in order of nondecreasing processing times. When we add a second machine, the problem remains easy (SPT still works), but when we add release dates instead, it becomes NP-hard (by a less obvious reduction). This is the reverse of what we saw for the makespan objective.

Figure 1 depicts the situation in English and in what Gene called Schedulese, the language proposed by Graham *et al.* [8] as a revision of the classification system devised by Conway, Maxwell and Miller in their book *Theory of Scheduling* (1967).

The above example illustrates a game that Dick Karp, Ben Lageweg, Gene and I played on an afternoon in September 1975 in the old Mathematisch Centrum in Amsterdam, amid its characteristic smell of overworked copying machines and stale beer. (The Amstel brewery was next door.) We soon realized that we needed the aid of a computer. Ben wrote a program that generated 4536 problem types, involving a single machine, identical, uniform and unrelated parallel machines, open shops, flow shops, job shops, various sorts of job characteristics such as preemption, precedence constraints, release dates and deadlines, and eight optimality criteria. With the known results

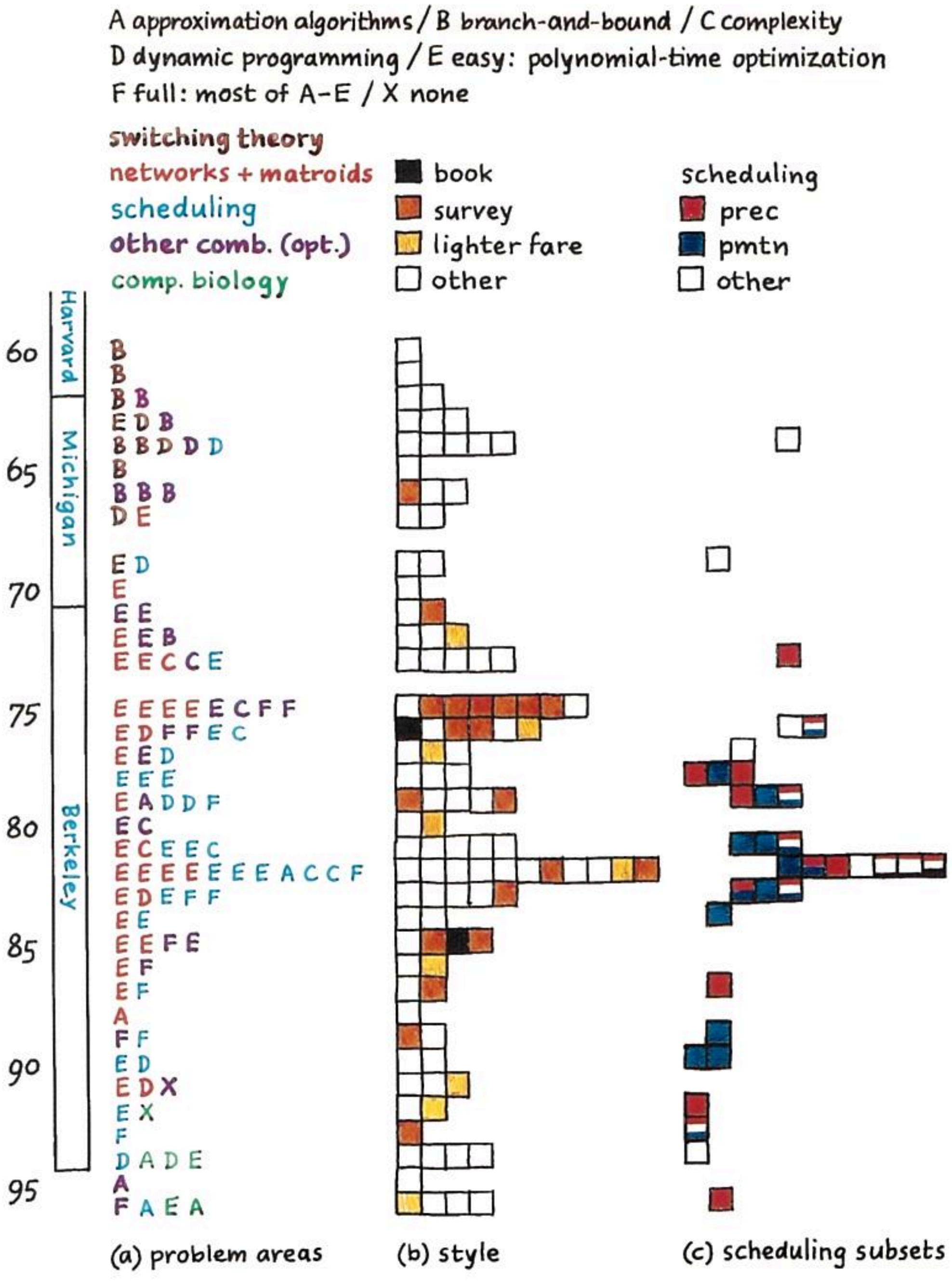

Plate 1. A research map

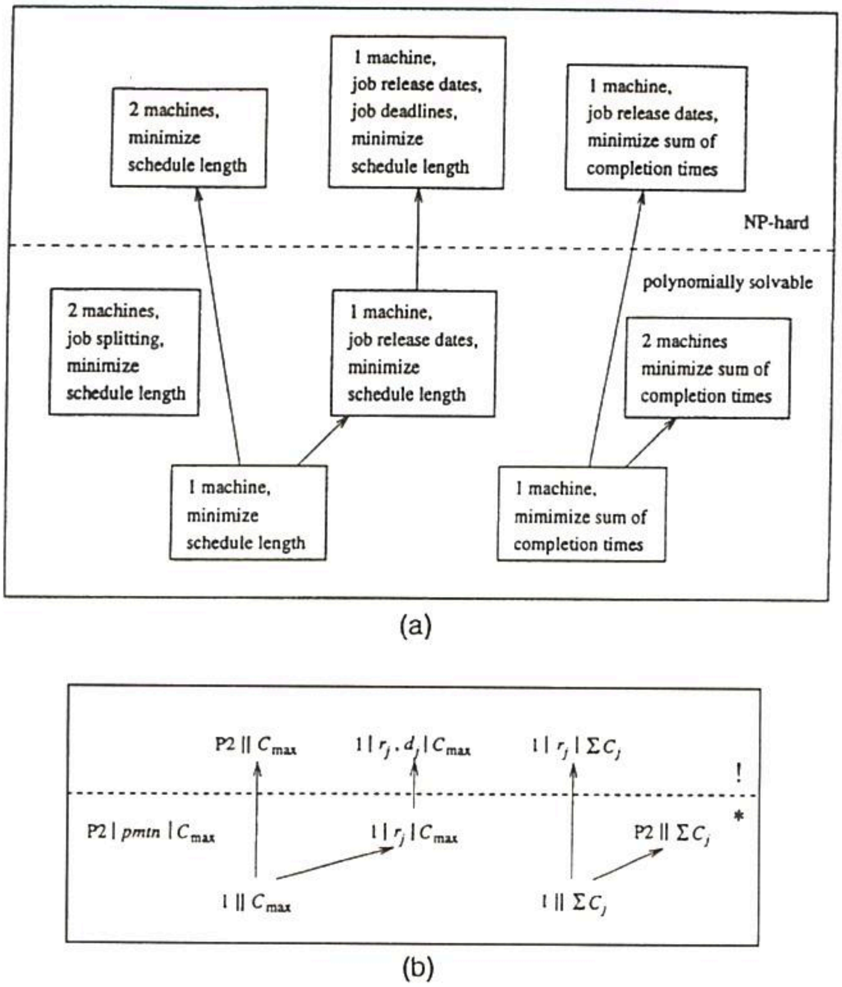
2 machines, minimize schedule length
1 machine, job release dates, job deadlines, minimize schedule length
1 machine, job release dates, minimize sum of completion times
NP-hard
polynomially solvable
2 machines, job splitting, minimize schedule length
1 machine, job release dates, minimize schedule length
2 machines minimize sum of completion times
1 machine, minimize schedule length
1 machine, mimimize sum of completion times
(a)
$P2 \| C_{\max}$
$1 | r_j, d_j | C_{\max}$
$1 | r_j | \Sigma C_j$
!
*
$P2 | pmtn | C_{\max}$
$1 | r_j | C_{\max}$
$P2 \| \Sigma C_j$
$1 \| C_{\max}$
$1 \| \Sigma C_j$
(b)

Figure 1. Reducibility among scheduling problems: (a) in English; (b) in Schedulese

as input, the program classified each problem as solvable in polynomial time, NP-hard, or open. The most useful part of the output was given by the borderlines: the maximal easy problems, the minimal and maximal open ones, and the minimal hard ones. Each new listing of open problems contained targets that we had overlooked in the previous round.

The computer-aided game started when we decided on a surprise for Alexander Rinnooy Kan on the occasion of his thesis defense. A listing of all scheduling problems with their complexity status seemed a good complement to his thesis. The surprise to us was that Ben's program proved to be a much more useful research tool than we had anticipated [15, 16]. This does not imply that all of those 4536 problems are for real. Many of them are quite artificial and exist only because a computer has generated them.

The automatic classification system is presently maintained by Peter Brucker and Sigrid Knust at the Universität Osnabrück for an extended class of deterministic machine scheduling problems; see `http://www.mathematik.uni-osnabrueck.de/research/OR/class/`.

5. PRECEDENCE CONSTRAINTS AND PREEMPTION

When we were generating random walks in scheduling space in 1975, Gene's main interest was not in NP-hardness proofs but in polynomial-time algorithms. His principal concern was the algorithmic treatment of precedence constraints and preemption. Plate 1(c) supports this observation: almost 40% of his scheduling work dealt with precedence constraints and over 40% with preemption. Gene's last manuscript on scheduling [17] considers precedence constraints with communication delays; see Section 7. His last research discussion was about staircase algorithms for handling preemption on parallel processors.

In Section 3 his *Management Science* paper of 1973 was cited. It gives a solution to the following problem. A finite set N of jobs is to be processed on a single machine. Each job $j \in N$ has a processing time p_j and a nondecreasing cost function f_j; if job j is completed at time t, a cost $f_j(t)$ is incurred. Furthermore, a digraph G with vertex set N is given; if G contains a path from job j to job k, then k cannot start before j has been completed. We wish to find a schedule that satisfies the precedence constraints and minimizes the maximum job completion cost.

We need some notation: $L \subset N$ is the set of jobs without successors in G, $P = \sum_{j \in N} p_j$ is the schedule length, and $f^*(S)$ is the cost of an optimal schedule for the job set $S \subset N$. The *least cost last* rule selects a job $l \in L$ satisfying $f_l(P) = \min_{j \in L} f_j(P)$, puts job l in the last position of the schedule, and repeats. Why does this work? We clearly have $f^*(N) \geqslant f_l(P)$ and $f^*(N) \geqslant f^*(N \setminus \{l\})$, and hence

$$f^*(N) \geqslant \max\{f_l(P), f^*(N \setminus \{l\})\}.$$

But the right-hand side of this inequality is precisely the cost of an optimal schedule under the condition that l is scheduled last.

The original proof uses a job interchange argument. Gene's above more elegant proof extends to the case that there are release dates and preemption is allowed [18]. Without release dates, there is no use to preemption; with release dates, the nonpreemptive case is NP-hard.

A problem that occupies a central position in Gene's work is the minimization of total weighted completion time on a single machine subject to precedence constraints. Each job $j \in N$ now also has a weight w_j and, if C_j denotes its completion time, we wish to find a schedule minimizing $\sum_{j \in N} w_j C_j$. In the absence of precedence constraints, Smith's ratio rule [19] uses a simple job interchange relation to show that the jobs must be scheduled in order of nondecreasing values p_j/w_j. Various special types of precedence constraints can still be solved in $O(n \log n)$ time, typically by a decomposition approach that is based on a string interchange relation. Horn [20] solved the case of *tree*-type constraints, Lawler [21] and Monma and Sidney [22, 23] handled *series–parallel* constraints. Very recently, Goemans and Williamson [24] gave a new proof for Gene's algorithm, using a linear programming formulation of the precedence-constrained single-machine problem due to Queyranne and Wang [25], which completely describes the scheduling polyhedron in the case of series–parallel constraints. Gene's NP-hardness proof for *general* precedence constraints [21] was a breakthrough at the time.

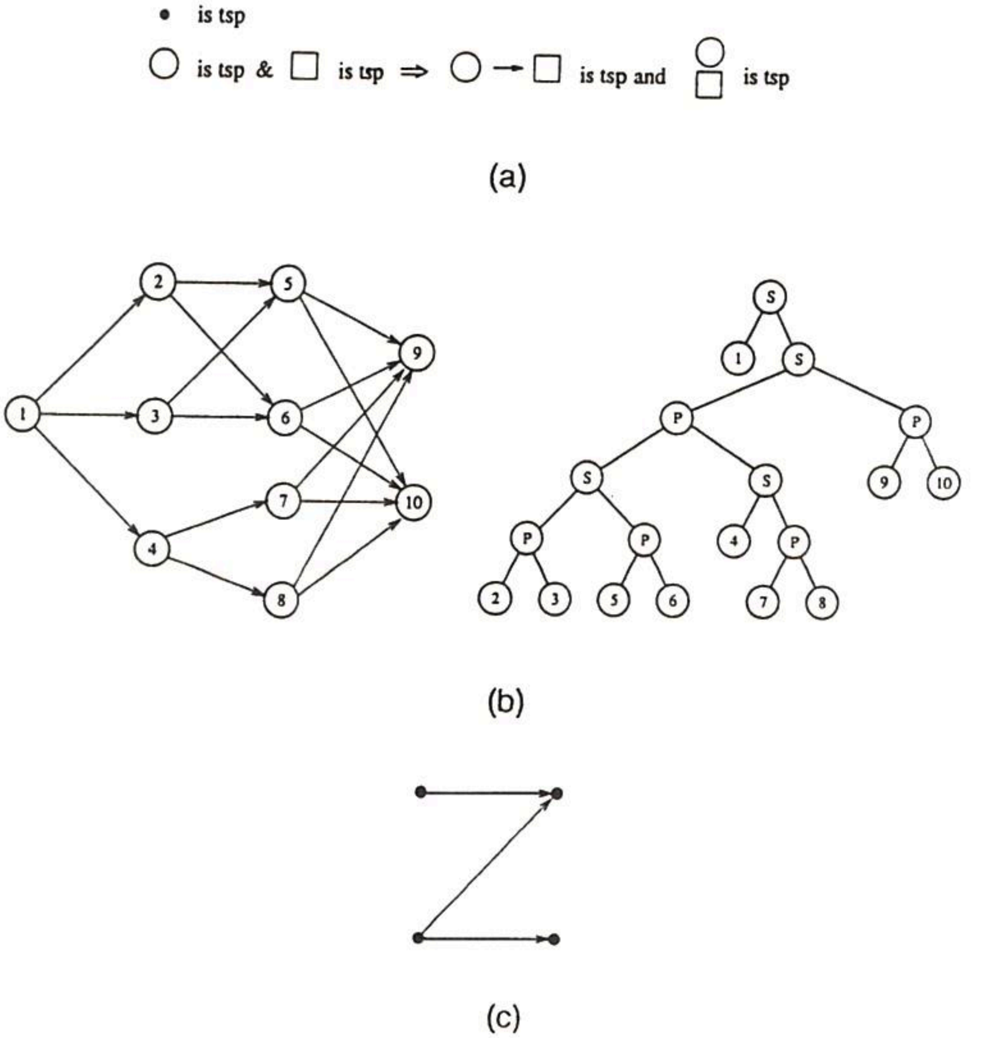

Figure 2. Series–parallel precedence constraints: (a) composition of transitive series–parallel (tsp) digraph; (b) decomposition of series–parallel digraph; (c) smallest non-series–parallel digraph

Series–parallel digraphs already occur in Gene's very first paper [26]. A digraph is series–parallel if its transitive closure is *transitive series–parallel*, which means that it can be recursively composed as follows (cf. Figure 2(a)): the digraph with a single vertex and no arcs is transitive series–parallel, and if the digraphs (V_1, A_1) and (V_2, A_2) are transitive series-parallel with $V_1 \cap V_2 = \emptyset$, then so are the series composition $(V_1 \cup V_2, A_1 \cup A_2 \cup (V_1 \times V_2))$ and the parallel composition $(V_1 \cup V_2, A_1 \cup A_2)$. Series–parallel digraphs can be recognized in linear time [27]; the smallest digraph that is not series–parallel is the Z-digraph shown in Figure 2(c).

The structure of a series-parallel digraph is displayed by a decomposition tree (cf. Figure 2(b)). Its leaves are the vertices, a P-node represents parallel composition, and an S-node represents series composition with the convention that left precedes right. The scheduling algorithms work their way up the tree and construct an optimal schedule for a parent node out of the schedules for its children.

Let us now turn to preemption. Gene often cited an experimental law, which states that a preemptive problem should not be harder that its nonpreemptive counterpart. Indeed, in many cases, the former problem is easier than the latter, and the distinction has the same flavour as that between linear and integer programming. But experimental laws have experimental exceptions. Here I know of 1·5 exceptions.

One exception occurs in job shop scheduling. If no preemption is allowed, there are two nice mystical 2's: the 2-job problem is easy [28], the 2-machine problem with a fixed number of jobs is easy [29], and the 3-machine 3-job problem is NP-hard [30]. If preemption is allowed, the 2-machine 3-job problem turns out to be NP-hard [31], a surprising and unsettling result that oversteps the law.

Half an exception occurs in scheduling unrelated parallel machines. Here each of n jobs is to be processed by one of m machines, and putting job j on machine i takes time p_{ij}. If we wish to minimize schedule length, everything is as it should be. The nonpreemptive case is NP-hard, because it is a generalized packing problem; the preemptive case can be solved by an LP-based algorithm due to Lawler and Labetoulle [32]. If the objective is the sum of the completion times, the nonpreemptive case is a bipartite matching problem: allocating job j to the kth last position on machine i gives a contribution of kp_{ij} to the objective [33, 34]. The preemptive total completion time problem, however, is one of the more vexing open questions.

Gene made important contributions to preemptive scheduling: the LP-based algorithm cited above, the staircase algorithm for minimizing makespan on uniform parallel machines subject to release dates with Jacques Labetoulle and others [35], and the polymatroidal flow algorithm for the same problem with release dates and deadlines with Chip Martel [36].

6. THE GREAT MATHEMATICAL SPUTNIK OF 1979

The Sputnik paper, which appeared in *The Sciences* and *The Mathematical Intelligencer* in 1980 [11], is Gene's most memorable foray into scientific journalism. It deals with the advent of the ellipsoid method, but also with the way in which professional journalists handle their material.

In January 1979, Rainer Burkard brought a *Doklady* paper to Oberwolfach, which he had obtained through a Polish colleague. The author was Leonid Khachiyan, the result was that the ellipsoid method solved linear programming problems in polynomial time, and the proofs were lacking. Gene brought the paper to Amsterdam, made a rough translation with the help of Milan Vlach, a visitor from Prague, and sent it around. Within weeks, the result had been verified, and the West knew about it before the *Doklady* issue had made it to the libraries. *Science* published a well-researched article about Khachiyan's achievement. A *New York Times* journalist managed to grossly misinterpret the *Science* article and reported on the front page that the TSP had been solved. All over the world, headlines appeared of the sort 'A Soviet Discovery Rocks World of Mathematics', 'Soviet Mathematician Is Obscure No More', 'Discovery Rocks World of Math, Computers', and 'The Russian Genius Who Has Rocked the Computer World'. It took some time before the dust settled. At the end, the *Times* printed a half-baked retraction.

The following is a quotation from the Sputnik paper.

> *The* **Times** *story appears to have been based on certain unshakable preconceptions of its writer, Malcolm W. Browne. Browne telephoned George Dantzig of Stanford University, a great pioneering authority on linear programming, and tried to force him into various admissions. Dantzig's version of the interview bears repeating. 'What about the Traveling Salesman Problem?' asked Browne. 'If there is a connection, I don't know what it is', said Dantzig. ('The Russian discovery proposed an approach for* [*solving*] *a class of problems related to the "Traveling Salesman Problem",' reported Browne.) 'What about cryptography?' asked Browne. 'If there is a connection, I don't know what it is', said Dantzig. ('The theory of codes could eventually be affected', reported Browne.) 'Is the Russian method practical?' asked Browne. 'No', said Dantzig. ('Mathematicians describe the discovery ... as a method by which computers can find solutions to a class of very hard problems that has hitherto been attacked on a hit-or-miss basis', reported Browne.)*

7. COMMUNICATION DELAYS

In the past decade, much attention has been paid to multiprocessor scheduling subject to precedence constraints with communication delays. With each arc (j, k) in the precedence digraph, a delay c_{jk} is associated; if jobs j and k are allocated to different machines, then k can only start c_{jk} time units after the completion of j.

The presence of positive communication delays implies that, among all the direct successors of a job, at most one, its *favoured child*, can start at its completion time; both jobs must then be allocated to the same machine. All the other successors will incur a delay. Similarly, among the direct predecessors of a job there is at most one *favoured parent*. The need to choose favoured children and parents causes an additional combinatorial difficulty, which implies that minimizing makespan is already NP-hard in case $m \geqslant n$, i.e., in the absence of machine capacity constraints.

Let us assume that we have m identical parallel machines, n unit-time jobs, and unit-time communication delays. Without delays, the case of tree-type constraints and the case of general precedence constraints and 2 machines are both solvable in linear time. With delays, the former problem is NP-hard [37] and the latter problem is open (constituting an annoying 2 rather than a mystical 2). Gene's contribution [17, 38] is a linear-time algorithm for tree-type constraints that computes a schedule whose length exceeds the optimum by no more than $m - 2$ time units; hence, its solves the 2-machine problem to optimality. A subsequent paper [39] proved a maximum deviation of $\lceil (m-2)/2 \rceil$, which is tight.

I will give an outline of Gene's approach. Let the precedence constraints be given in the form of an outforest F. A schedule is said to have the *favoured child property* if every parent has exactly one child scheduled earlier than the others. An interchange argument shows that there exists an optimal schedule with the favoured child property.

Given a choice of favoured children, it is easy to find a minimum-length schedule. First, replace each arc $(j, l) \in F$ by an arc (k, l) whenever j has a favoured child $k \neq l$; call the new outforest $\bar{F}$. Next, ignore the delays and schedule $\bar{F}$ in linear time, e.g., by Hu's algorithm [40].

It is apparently hard to make an optimal choice of favoured children. It is easy, however, to make a choice of favoured children that yields an outforest $\bar{F}^*$ of minimum depth, and thereby solves the problem in case $m \geqslant n$. For each job j, the depth d_j of the subtree of $\bar{F}^*$ rooted at j is given by

$$\begin{array}{ll} d_j = 1 & \text{when } j \text{ is a leaf} \\ d_j = \max\{1 + \max\limits_{(j,k)\in F} d_k, 2 + \max 2_{(j,k)\in F} d_k\} & \text{when } j \text{ is not a leaf} \end{array}$$

where the max 2-term denotes the highest d-value among the children of j after a child with highest d-value has been deleted. For any parent job in F, any child with maximum d-value is chosen as its favoured child.

This choice is not too bad when m is given as part of the problem instance. It requires a careful argument to show that the algorithm has the absolute performance guarantee mentioned above.

8. CLOSING REMARKS

Life is more than scientific achievement. I find it difficult to imagine anyone who enjoys life more than Gene did. He was a remarkable man, not only because of his inquiring mind, but also because of his expository gift and his personal commitment.

His expository talent was, to a large extent, related to his style of research. He usually preferred to do things his own way, by going back to the original question, identifying what he saw as the essential difficulty, and eventually achieving a deeper insight as a result. He would apply the same principle when he had difficulties in understanding cumbersome presentations of other people's work. By taking a fresh look at things, he often arrived at a simpler proof of a more general result.

His commitment was in no way restricted to topics of research. He was ready to approach any issue, scientific or not, in an intelligent and thought-provoking way. Throughout his years at Berkeley, he did everything imaginable to make the university a more humane and more stimulating place to study. He helped the individual student fight the bureaucracy, and reformed what the university taught and to whom it taught. He showed a true and pure interest in other people. It made him the most honest and inspiring friend one could wish to have.

ACKNOWLEDGEMENTS

I wish to thank Karen Aardal, Francesco Maffioli and David Shmoys for their help in taking care of Gene's scientific legacy, and for their comments on a draft of this paper. I am grateful to Tom Liebling, who invited me to give the talk on which this paper is based, to Edmund Burke, who asked me to contribute to the first issue of the new *Journal of Scheduling*, and to Petra Schuurman, who provided technical assistance. This research was partially supported by NATO Collaborative Research Grant 971550.

REFERENCES

1. E. L. Lawler, 'Old stories', in J. K. Lenstra, A. H. G. Rinnooy Kan and A. Schrijver (eds.), *History of Mathematical Programming: A Collection of Personal Reminiscences*, CWI, Amsterdam and North-Holland, Amsterdam, 1991, pp. 97–106.
2. In memoriam: Eugene L. Lawler, *Optima*, **44**, 10 (1994).
3. K. Aardal, J. K. Lenstra, F. Maffioli and D. B. Shmoys (eds.), *Selected papers of Eugene L. Lawler*. CWI Tract, CWI, Amsterdam, to appear.
4. J. K. Lenstra and D. B. Shmoys (eds.), *Mathematical Programming Series B*, **82**(1, 2), 1–290 (1998).
5. E. L. Lawler, *Combinatorial Optimization: Networks and Matroids*. Holt, Rinehart and Winston, New York, NY, 1976.

6. E. L. Lawler, J. K. Lenstra, A. H. G. Rinnooy Kan and D. B. Shmoys (eds.), *The Traveling Salesman Problem; A Guided Tour of Combinatorial Optimization.* Wiley, Chichester, 1985.
7. P. C. Gilmore, E. L. Lawler and D. B. Shmoys, 'Well-solved special cases', in E. L. Lawler, J. K. Lenstra, A. H. G. Rinnooy Kan and D. B. Shmoys (eds.), *The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization*, Wiley, Chichester, 1985, pp. 87–143.
8. R. L. Graham, E. L. Lawler, J. K. Lenstra and A. H. G. Rinnooy Kan, 'Optimization and approximation in deterministic sequencing and scheduling: a survey', *Ann. Discrete Math.*, **5**, 287–326 (1979).
9. E. L. Lawler, 'Recent results in the theory of machine scheduling', in A. Bachem, M. Grötschel and B. Korte (eds.), *Mathematical Programming; The State of the Art; Bonn 1982*, Springer, Berlin, 1983, pp. 202–234.
10. E. L. Lawler, J. K. Lenstra, A. H. G. Rinnooy Kan and D. B. Shmoys, 'Sequencing and scheduling: algorithms and complexity', in S. C. Graves, A. H. G. Rinnooy Kan and P. H. Zipkin (eds.), *Logistics of Production and Inventory; Handbooks in Operations Research and Management Science*, Vol. 4. North-Holland, Amsterdam, 1993, pp, 445–522.
11. E. L. Lawler, 'The great mathematical Sputnik of 1979', *The Sciences* (September 1980), 12–15, 34–35, *The Math. Intelligencer*, **2**, 190–198 (1980).
12. W. L. Eastman, 'Linear programming with pattern constraints', *Ph.D. thesis*, July 1958; Report BL-20, Computation Laboratory, Harvard University, Cambridge, MA, 1958.
13. E. L. Lawler, 'This week's citation classic', *Current Contents (Arts & Humanities; Social & Behavioral Sciences; Engineering, Technology & Applied Sciences; Physical, Chemical & Earth Sciences)*, vol. 2, January 12, 1987, p. 16.
14. E. L. Lawler, 'A "pseudopolynomial" algorithm for sequencing jobs to minimize total tardiness', *Ann. Discrete Math.*, **1**, 331–342 (1977).
15. B. J. Lageweg, E. L. Lawler, J. K. Lenstra and A. H. G. Rinnooy Kan, 'Computer aided complexity classification of deterministic scheduling problems', *Report BW 138/81, Afdeling Mathematische Besliskunde*, Mathematisch Centrum, Amsterdam, 1981.
16. B. J. Lageweg, J. K. Lenstra, E. L. Lawler and A. H. G. Rinnooy Kan, 'Computer-aided complexity classification of combinatorial problems', *Commun. ACM*, **25**, 817–822 (1982).
17. E. L. Lawler, 'Scheduling trees on multiprocessors with unit communication delays', *Manuscript, presented at the Workshop on Models and Algorithms for Planning and Scheduling Problems*, Villa Vigoni, Menaggio, 14–18 June, 1993, incorporated in [38].
18. K. R. Baker, E. L. Lawler, J. K. Lenstra and A. H. G. Rinnooy Kan, 'Preemptive scheduling of a single machine to minimize maximum cost subject to release dates and precedence constraints', *Oper. Res.*, **31**, 381–386 (1983).
19. W. E. Smith, 'Various optimizers for single-stage production', *Naval Res. Logist. Quart.*, **3**, 59–66 (1954).
20. W. A. Horn, 'Single-machine job sequencing with treelike precedence ordering and linear delay penalties', *SIAM J. Appl. Math.*, **23**, 189–202 (1972).
21. E. L. Lawler, 'Sequencing jobs to minimize total weighted completion time subject to precedence constraints', *Ann. Discrete Math.*, **2**, 75–90 (1978).
22. C. L. Monma and J. B. Sidney, 'Sequencing with series-parallel precedence constraints', *Math. Oper. Res.*, **4**, 215–224 (1979).
23. C. L. Monma and J. B. Sidney, 'Optimal sequencing via modular decomposition: characterizations of sequencing functions', *Math. Oper. Res.*, **12**, 22–31 (1987).
24. M. X. Goemans and D. P. Williamson, 'Two-dimensional Gantt charts and a scheduling algorithm of Lawler', *IBM Research Report RC 21032*, IBM Research Division, Yorktown Heights, NY, 1997.
25. M. Queyranne and Y. Wang, 'Single-machine scheduling polyhedra with precedence constraints', *Math. Oper. Res.*, **16**, 1–20 (1991); Erratum. *Math. Oper. Res.*, **20**, 768 (1995).
26. E. L. Lawler and G. A. Salton, 'The use of parenthesis-free notation for the automatic design of switching circuits', *IRE Trans. Electron. Comput.*, **EC-9**, 342–352 (1960).
27. J. Valdes, R. E. Tarjan and E. L. Lawler, 'The recognition of series parallel digraphs', *SIAM J. Comput.*, **11**, 298–313 (1982).
28. S. B. Akers, 'A graphical approach to production scheduling problems', *Oper Res.*, **4**, 244–245 (1956).
29. P. Brucker, 'A polynomial algorithm for the two machine job-shop scheduling problem with a fixed number of jobs', *OR Spektrum*, **16**, 5–7 (1994).
30. Y. N. Sotskov and N. V. Shakhlevich, 'NP-hardness of shop-scheduling problems with three jobs', *Discrete Appl. Math.*, **59**, 237–266 (1995).
31. P. Brucker, S. A. Kravchenko and Y. N. Sotskov, 'Preemptive job-shop scheduling problems with a fixed number of jobs', *Math. Meth. Oper. Res.*, to appear.
32. E. L. Lawler and J. Labetoulle, 'On preemptive scheduling of unrelated parallel processors by linear programming', *J. Assoc. Comput. Mach.*, **25**, 612–619 (1978).
33. W. A. Horn, 'Minimizing average flow time with parallel machines', *Oper. Res.*, **21**, 846–847 (1973).
34. J. L. Bruno, E. G. Coffman, Jr. and R. Sethi, 'Scheduling independent tasks to reduce mean finishing time', *Commun. ACM*, **17**, 382–387 (1974).
35. J. Labetoulle, E. L. Lawler, J. K. Lenstra and A. H. G. Rinnooy Kan, 'Preemptive scheduling of uniform machines subject to release dates', in W. R. Pulleyblank (ed.), *Progress in Combinatorial Optimization*, Academic Press, New York, NY, 1984, pp. 245–261.

36. E. L. Lawler and C. U. Martel, 'Computing maximal "polymatroidal" network flows', *Math. Oper. Res.*, **7**, 334–347 (1982).
37. J. K. Lenstra, M. Veldhorst and B. Veltman, 'The complexity of scheduling trees with communication delays', *J. Algorithms*, **20**, 157–173 (1996).
38. T. A. Varvarigou, V. P. Roychowdhury, T. Kailath and E. L. Lawler, 'Scheduling in and out forests in the presence of communication delays', *IEEE Trans. Parallel Distributed Systems*, **7**, 1065–1074 (1996).
39. F. Guinand, C. Rapine and D. Trystram, 'Worst case analysis of Lawler's algorithm for scheduling trees with communication delays', *IEEE Trans. Parallel Distributed Systems*, **8**, 1085–1086 (1997).
40. T. C. Hu, 'Parallel sequencing and assembly line problems', *Oper. Res.*, **9**, 841–848 (1961).