On Preemptive Scheduling of Unrelated Parallel Processors by Linear Programming

E. L. LAWLER

University of California at Berkeley, Berkeley, California

AND

J. LABETOULLE

IRIA, Rocquencourt, France

ABSTRACT It is shown that certain problems of optimal preemptive scheduling of unrelated parallel processors can be formulated and solved as linear programming problems As a by-product of the linear programming formulations of these problems, upper bounds are obtained on the number of preemptions required for optimal schedules In particular it is shown that no more than $O(m^2)$ preemptions are necessary, in order to schedule n jobs on m unrelated processors so as to minimize makespan

KEY WORDS AND PHRASES preemptive scheduling, parallel processors, linear programming

CR CATEGORIES. 4 32, 5 39

1. *Introduction*

The general problem we wish to deal with in this paper is that of finding optimal preemptive schedules for independent jobs on unrelated parallel processors. We show that certain specific scheduling problems of this type, e.g. minimization of makespan, can be formulated and solved as linear programming problems. We also show that the linear programming formulations provide a means for establishing upper bounds on the number of preemptions required for an optimal schedule.

As part of the general problem formulation, we assume that there are *m processors*, indexed $i = 1, 2, \dots, m$, and *n jobs*, indexed $j = 1, 2, \dots, n$. A processor can work on only one job at a time, and a job can be worked on by only one processor at a time. The processing of a job may be interrupted at any time and resumed at a later time, by the same processor or a different processor. There is no cost and no time loss associated with such an interruption or "preemption."

We assume that the input data for a problem instance include mn positive numbers p_{ij}, where p_{ij} represents the total processing time required to complete job j, if the job is worked on exclusively by processor i. More generally, if processor i works on job j for a total time t_{ij}, then it is necessary that

This is a revised version of the paper "Scheduling of Parallel Machines with Preemptions," IRIA Rapport de Recherche No 203, 1976, prepared while the first author was a visitor at IRIA The revision was supported in part by the National Science Foundation under Grant MCS 76-17605

Authors' addresses E L Lawler, Computer Science Division, University of California at Berkeley, Berkeley, CA 94720, J Labetoulle, IRIA, Rocquencourt 78150, France.

$$\sum_{i=1}^{m} \frac{t_{ij}}{p_{ij}} = 1,$$

in order for the job to be completed.

We assume no particular relation between the p_{ij} values. That is, the processors are *unrelated.* This is in contrast to two more specialized cases, as follows. If, for all i, j, k, $p_{ij} = p_{kj}$, then the processors are *identical.* If each p_{ij} can be expressed in the form $p_{ij} = q_i p_j$, where q_i (a "slowness factor") and p_j are parameters associated with machine i and job j, then the machines are said to be *uniform.*

For a given feasible schedule, the last point in time at which job j is processed is its *completion time* C_j The first and most important problem we wish to consider is that of finding a feasible schedule for which "makespan" or maximum completion time,

$$C_{\max} = \max_j \{C_j\},$$

is minimized. We shall demonstrate that there is a polynomial bounded reduction of this problem to a linear programming problem. More specifically, we shall formulate a linear programming problem in $mn + 1$ variables and $2n + m$ (equality and inequality) constraints. We shall then show that one can obtain an optimal schedule (via "open shop" theory) from an optimal solution to the linear programming problem. As a by-product of this analysis, we prove that there exists a $C_{\max}$-optimal schedule with no more than $O(m^2)$ preemptions.

These results are in contrast to the situation for identical and uniform processors. For identical processors, a very simple $O(n)$ algorithm, due to McNaughton [4], yields a $C_{\max}$-optimal schedule with no more than $m - 1$ preemptions. Gonzalez and Sahni [2] have obtained a more complex $O(n + m \log m)$ algorithm for the case of uniform processors, and show that no more than $2(m - 1)$ preemptions are required for an optimal solution.

There is no known polynomial-bounded algorithm for the general linear programming problem, nor has the problem been shown to be NP-complete. It follows that our solution to the $C_{\max}$ problem for unrelated processors does not resolve the question of whether the problem is either NP-complete or polynomial bounded. However, in a later paper we shall show that *for any fixed number of processors m* there is a polynomial-bounded algorithm. (Note: For the case $m = 2$, there is a particularly efficient algorithm [3].)

Following our discussion of the $C_{\max}$ problem, we consider extensions of the linear programming method of solution to objective functions more general than $C_{\max}$. In particular we consider the problem of minimizing

$$L_{\max} = \max_j \{L_j\}, \tag{1.1}$$

where

$$L_j = C_j - d_j \tag{1.2}$$

denotes the *lateness* of job j with respect to a given *due date* d_j. Upper bounds on the number of preemptions required for an optimal schedule are obtained for this, and for a much more general class of objective functions.

2. *Linear Programming Formulation of C_{max} Problem*

We suppose all jobs are available for processing at time $t = 0$. Consider any feasible schedule of n jobs on m unrelated processors, where with respect to this schedule,

$$t_{ij} = \text{the total amount of time that processor } i \text{ works on job } j.$$

It is evident that the values of $C_{\max}$ and t_{ij} for the schedule constitute a feasible solution to the following linear programming problem:

minimize $C_{\max}$

subject to

$$\sum_{i=1}^{m} \frac{t_{ij}}{p_{ij}} = 1, \qquad j = 1, 2, \ldots, n; \tag{2.1}$$

$$\sum_{i=1}^{m} t_{ij} \leq C_{\max}, \quad j = 1, 2, \ldots, n; \tag{2.2}$$

$$\sum_{j=1}^{n} t_{ij} \leq C_{\max}, \quad i = 1, 2, \ldots, m; \tag{2.3}$$

$$t_{ij} \geq 0.$$

We assert that the converse is also true. That is, for any feasible solution to the linear programming problem, there is a feasible schedule with the same values of t_{ij} and $C_{\max}$. In order to prove this assertion, we solve what Gonzalez and Sahni [1] call the preemptive "open shop" scheduling problem. In Section 3 we indicate a solution to this problem, rather than merely referring to [1], in order to have the tools more readily at hand for obtaining a bound on the number of preemptions required for an optimal solution.

3. *Construction of a Feasible Solution*

Suppose we are given an $m \times n$ nonnegative matrix $T = (t_{ij})$ and a value $C_{\max}$, where

$$C_{\max} = \max\left\{\max_i\left\{\sum_j t_{ij}\right\}, \max_j\left\{\sum_i t_{ij}\right\}\right\}. \tag{3.1}$$

We wish to show that it is possible to construct a feasible schedule with the given value of $C_{\max}$.

The pertinent assumptions are as follows. Processor i is to work on job j for a total amount of time t_{ij}. A processor can work on only one job at a time and a job can be worked on by only one processor at a time. There is no restriction on the order in which a given job can be worked on by the different processors, or on the order in which a given processor can work on jobs. (Hence the term "open shop.") There is no loss of time occasioned by the interruption or preemption of jobs. All jobs are available for processing at time $t = 0$.

Let us call row i (column j) of matrix T *tight* if $\sum_j t_{ij} = C_{\max}$ ($\sum_i t_{ij} = C_{\max}$), and *slack* otherwise. Suppose we are able to find a subset of strictly positive elements of T, with exactly one element of the subset in each tight row and in each tight column and no more than one element in any slack row or column. We shall call such a subset of elements a *decrementing set*, and use it to construct a *partial schedule* of length δ, for some suitably chosen $\delta > 0$. In this partial schedule processor i works on job j for $\min\{t_{ij}, \delta\}$ units of time, for each element t_{ij} in the decrementing set. We then replace t_{ij} by $\max\{0, t_{ij} - \delta\}$, for each element in the decrementing set, thereby obtaining a new matrix T', for which $C'_{\max} = C_{\max} - \delta$ satisfies condition (3.1).

For example, suppose $C_{\max} = 11$ and

$$T = \begin{array}{cc} \begin{pmatrix} 3 & \textcircled{4} & 0 & 4 \\ \textcircled{4} & 0 & 6 & 0 \\ 4 & 0 & 0 & \textcircled{6} \end{pmatrix} & \begin{array}{c} 11 \\ 10, \\ 10 \end{array} \\ \begin{array}{cccc} 11 & 4 & 6 & 10 \end{array} & \end{array}$$

with row and column sums as indicated on the margins of the matrix. One possible decrementing set is indicated by the encircled elements. Choosing $\delta = 4$, we obtain $C'_{\max} = 7$ and T' as shown below, with the partial schedule indicated to the right:

$$T' = \begin{array}{ccc} \begin{pmatrix} \textcircled{3} & 0 & 0 & 4 \\ 0 & 0 & \textcircled{6} & 0 \\ 4 & 0 & 0 & \textcircled{2} \end{pmatrix} & \begin{array}{c} 7 \\ 6 \\ 6 \end{array} & \begin{array}{c} \boxed{2} \\ \boxed{1} \\ \boxed{4} \end{array} \\ \begin{array}{cccc} 7 & 0 & 6 & 6 \end{array} & & 4 \end{array}$$

A decrementing set of T' is indicated by the encircled elements.

There are various constraints that must be satisfied by δ, in order for $C'_{\max} = C_{\max} - \delta$ to satisfy condition (3.1) with respect to T'. First, if t_{ij} is an element of the decrementing set in a tight row or column, then clearly it is necessary that $\delta \le t_{ij}$, else there will be a row or column sum of T' which is strictly greater than $C_{\max} - \delta$. Similarly, if t_{ij} is an element of the decrementing set in a slack row (slack column), then it is necessary that

$$\delta \le t_{ij} + C_{\max} - \sum_k t_{ik} \qquad \Big(\delta \le t_{ij} + C_{\max} - \sum_k t_{kj}\Big).$$

And if row i (column j) contains no element of the decrementing set (and is therefore necessarily slack), it is necessary that

$$\delta \le C_{\max} - \sum_j t_{ij} \qquad \Big(\delta \le C_{\max} - \sum_i t_{ij}\Big).$$

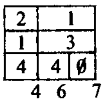
Thus for the example above we have

$$\delta \le t_{12} = 4, \qquad \delta \le t_{21} = 4,$$
$$\delta \le t_{34} + C_{\max} - \sum_k t_{3k} = 7, \qquad \delta \le t_{34} + C_{\max} - \sum_k t_{k4} = 7,$$
$$\delta \le C_{\max} - \sum_k t_{k3} = 5.$$

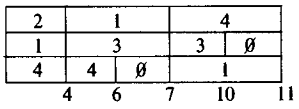
Suppose δ is chosen to be maximum, subject to conditions indicated above. Then either T' will contain at least one less strictly positive element than T or else T' will contain at least one more tight column or tight row (with respect to $C'_{\max}$) than T. It is thus apparent that no more than $r + m + n$ iterations, where r is the number of strictly positive elements in T, are necessary to construct a feasible schedule of length $C_{\max}$.

To illustrate this point, we continue with the example. Choosing $\delta = 3$, we obtain from T' the matrix T'', with the augmented partial schedule shown to the right:

$$T'' = \begin{array}{cc} \begin{pmatrix} 0 & 0 & 0 & \textcircled{4} \\ 0 & 0 & \textcircled{3} & 0 \\ \textcircled{4} & 0 & 0 & 0 \end{pmatrix} & \begin{matrix} 4 \\ 3 \\ 4 \end{matrix} \\ \begin{matrix} 4 & 0 & 3 & 4 \end{matrix} & \end{array}$$



(The symbol "Ø" indicates idle time.) The final decrementing set yields the following complete schedule:



To complete our proof, we need the following lemma.

LEMMA 1. *For any nonnegative matrix T and C_{max} satisfying condition* (3.1), *there exists a decrementing set.*

PROOF. From the $m \times n$ matrix T construct an $(m + n) \times (m + n)$ matrix U, as indicated below:

$$U = \left(\begin{array}{c|c} T & D_m \\ \hline D_n & T^t \end{array}\right)$$

Here T^t denotes the transpose of T. D_m and D_n are $m \times m$ and $n \times n$ diagonal matrices of nonnegative "slacks," determined in such a way that each row sum and column sum of U is equal to $C_{\max}$. It follows that $(1/C_{\max})U$ is a doubly stochastic matrix. The well-known Birkhoff–von Neumann theorem states that a doubly stochastic matrix is a convex combination of permutation matrices. It is easily verified that any one of the permutation matrices in such a convex combination is identified with a decrementing set of T. Q.E.D.

There are several possible ways to construct a decrementing set. For our purposes, it is sufficient to note that one can construct the matrix U from T and then solve an assignment problem over U, which can be done in polynomial time. This observation, together with the observation that no more than a polynomial number of such assignment problems need be solved, is sufficient to establish a polynomial bound for the schedule construction procedure. Gonzalez and Sahni [1] have obtained time bounds of $O(r^2)$ and $O(r(\min\{r, m^2\} + m \log n))$, where r is the number of strictly positive elements in T.

4. *Bounding the Number of Preemptions*

We now seek to establish an upper bound on the number of preemptions required for a $C_{\max}$-optimal schedule on unrelated parallel processors.

To be precise, we say that a job is *preempted* at time t if execution of the job is suspended on some processor at time t before its completion. If a processor begins or resumes execution of a job at time t' and its processing is continuous until time t, when the job is either completed or preempted, then $[t', t]$ is called an *active period* for the job. The total number of preemptions in a schedule is thus equal to the total number of active periods in excess of n.

Now consider the linear programming problem formulated in Section 2. This problem has n equality constraints (2.1), m inequality constraints (2.2), and m inequality constraints (2.3). It follows from elementary linear programming theory that there exists an optimal basic solution with no more than $n + r_2 + r_3$ strictly positive variables, where r_2 and r_3 denote the number of inequality constraints (2.2) and (2.3) which are satisfied with strict equality. Clearly $0 \leq r_3 \leq m$. If $n > m$, $0 \leq r_2 \leq m - 1$. And if $n \leq m$, at most $m - 1$ constraints (2.2) are nonredundant. It follows that there is an optimal solution with at most $n + 2m - 1$ strictly positive variables, one of which is $C_{\max}$.

We may thus assume there exists an optimal solution to the linear programming problem with no more than $n + 2(m - 1)$ strictly positive t_{ij} values. If we could construct a schedule (with the given value of $C_{\max}$) with exactly one active period for each positive t_{ij} value, then we should have an upper bound of $2(m - 1)$ on the number of preemptions required for a $C_{\max}$-optimal schedule. However, the schedule construction procedure generally introduces additional preemptions. We must now establish an upper bound on this number.

We shall propose a variation of the schedule construction procedure, with the objective of reducing the number of preemptions in the resulting schedule. (This variation also happens to admit a better polynomial bound on its running time, but this is not our principal concern here.) What we shall do is replace all of the jobs which are represented by a single positive t_{ij} value by m dummy jobs. We shall then apply the schedule construction procedure to find a feasible schedule with these dummy jobs. Finally, we shall create a schedule for the original set of jobs by reassigning the active periods for the dummy jobs to the jobs which they replaced.

Consider the example from the previous section where $C_{\max} = 11$ and

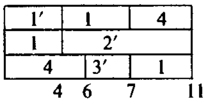
$$T = \begin{array}{c} \begin{pmatrix} 3 & 4 & 0 & 4 \\ 4 & 0 & 6 & 0 \\ 4 & 0 & 0 & 6 \end{pmatrix} \begin{array}{c} 11 \\ 10 \\ 10 \end{array} \\ \begin{array}{cccc} 11 & 4 & 6 & 10 \end{array} \end{array}$$

(This is actually not a *basic* feasible solution to the linear program, but this is of no consequence.) Let us remove the columns containing exactly one strictly positive t_{ij} value and add dummy columns, to obtain the matrix T':

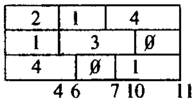
$$T' = \begin{array}{c} \begin{pmatrix} 3 & 4 & 4 & 0 & 0 \\ 4 & 0 & 0 & 7 & 0 \\ 4 & 6 & 0 & 0 & 1 \end{pmatrix} \begin{array}{c} 11 \\ 11 . \\ 11 \end{array} \\ \begin{array}{ccccc} 11 & 10 & 4 & 7 & 1 \end{array} \end{array}$$

The indices of the jobs identified with the first two columns of T' are 1 and 4. Let us assign indices 1', 2', 3' to the dummy jobs. Note that we have given the dummy jobs t_{ij} values so that all rows of T' are tight.

The schedule construction procedure applied to T' yields as a schedule:



We now fill in the active periods for the dummy jobs with active periods for the jobs which they replaced, plus idle time, obtaining the same schedule as we happened to obtain by the original procedure:



In the case of this example, a schedule was constructed for the matrix T' in which there were no preemptions of the dummy jobs. Hence it was possible to create a schedule for the original set of jobs in which there were no preemptions of any of the jobs which the dummy jobs replaced. In general the number of preemptions required for the original set of jobs is bounded by the number of preemptions in the schedule constructed for the matrix T'.

The matrix T' has at most $m + r_2 + r_3 - 1$ columns and at most $m + 2(r_2 + r_3 - 1)$ strictly positive elements. Each iteration of the schedule construction procedure either reduces an element of the T' matrix to zero, or causes an additional column to become tight. Exactly m elements become zero at the last iteration. Hence there are at most $2(r_2 + r_3) - 1$ iterations of the first kind. There are $m - r_2$ iterations of the second kind, and hence no more than $m + r_2 + 2r_3 - 1$ iterations in all. Each iteration introduces at most m active periods into the resulting schedule, so there are at most $m(m + r_2 + 2r_3 - 1)$ active periods in all. The number of active periods in excess of $m + r_2 + r_3 - 1$, and hence the number of preemptions, is thus bounded by $m(m + r_2 + 2r_3 - 1) - (m + r_2 + r_3 - 1)$. Taking $r_2 = m - 1$, $r_3 = m$, we have the following theorem.

THEOREM 1. *An upper bound on the number of preemptions required for a C_{max}-optimal schedule on unrelated processors is* $4m^2 - 5m + 2$.

The bound indicated by the theorem is certainly not tight, inasmuch as it is known that no more than 2 preemptions are required for the case $m = 2$ [3]. Moreover, we have not been able to establish that $O(m^2)$ preemptions may be required for an optimal schedule, or even that more than $2(m - 1)$ may be necessary.

5. *The L_{max} Problem*

We now formulate a linear programming problem to minimize $L_{\max}$, as defined by (1.1) and (1.2).

Assume the jobs are numbered in nondecreasing due date order, i.e. $d_1 \le d_2 \le \quad \le d_n$. Let

$t_{ij}^{(1)}$ = the total amount of time that processor i works on job j in the time period $[0, d_1 + L_{\max}]$,

and, for $k = 2, 3, \ldots, n$, let

$t_{ij}^{(k)}$ = the total amount of time that processor i works on job j in the time period $[d_{k-1} + L_{\max}, d_k + L_{\max}]$.

Then we have the linear programming problem

minimize $L_{\max}$

subject to

$$\sum_{i=1}^{m} \sum_{k=1}^{j} \frac{t_{ij}^{(k)}}{p_{ij}} = 1, \qquad j = 1, 2, \ldots, n;$$

$$\sum_{i=1}^{m} t_{ij}^{(1)} \le d_1 + L_{\max}, \quad j = 1, 2, \ldots, n;$$

$$\sum_{i=1}^{m} t_{ij}^{(k)} \le d_k - d_{k-1}, \quad j = k, k+1, \ldots, n, \quad k = 2, 3, \ldots, n;$$

$$\sum_{j=1}^{n} t_{ij}^{(1)} \le d_1 + L_{\max}, \quad i = 1, 2, \ldots, m;$$

$$\sum_{j=k}^{n} t_{ij}^{(k)} \le d_k - d_{k-1}, \quad i = 1, 2, \ldots, m, \qquad k = 2, 3, \ldots, n;$$

$$t_{ij}^{(k)} \ge 0.$$

Given an optimal solution to this linear programming problem, an $L_{\max}$-optimal schedule can be obtained by applying the schedule construction procedure of Section 3 to each matrix $T^{(k)} = (t_{ij}^{(k)})$, $k = 1, 2, \ldots, n$. Let $p(m, k) \le O(m^2)$ be an upper bound on the number of preemptions required for the subschedule constructed from $T^{(k)}$. Then an upper bound on the total number of preemptions required for an $L_{\max}$-optimal schedule can be seen to be

$$2(m-1)n + \sum_{k=1}^{n} p(m, k) \le O(m^2 n).$$

It is not difficult to construct examples for which $O(n)$ preemptions are necessary for an $L_{\max}$-optimal schedule.

6. *Costs of Processing*

The linear programming formulations we have obtained suggest that we might include a "cost of processing" in the objective functions for these problems. Let c_{ij} = the cost of processing job j on processor i for one unit of time. Then, for example, rather than only minimizing $L_{\max}$, we may choose to minimize

$$L_{\max} + \sum_i \sum_j \sum_k c_{ij} t_{ij}^{(k)}.$$

It is a well-known fact that the convex combination of any two feasible solutions to a linear programming problem is also a feasible solution. Thus, if $L_{\max}$, T and $L'_{\max}$, T' are feasible solutions, then so is $\lambda L_{\max} + (1-\lambda)L'_{\max}$, $\lambda T + (1-\lambda)T'$, for any λ, $0 \le \lambda \le 1$.

Let us say that (L, C) is a *feasible* pair of values if there exists a feasible schedule for which

$$\sum_i \sum_j \sum_k c_{ij} t_{ij}^{(k)} \le C, \qquad L_{\max} \le L.$$

The preceding remarks about convexity indicate that the feasible points (L, C) form a convex region in the plane, as indicated in Figure 1. Or, to put it another way, if $L(C)$ denotes the minimum attainable value of $L_{\max}$, over all schedules with cost of processing not exceeding C, then L is a convex function of C.

7. *A General Bound on the Number of Preemptions*

We shall now obtain an upper bound on the number of preemptions required for an optimal schedule, with respect to a very broad class of optimization criteria. Specifically, we suppose that we wish to find a schedule which minimizes

$$f(C_1, C_2, \ldots, C_n) + \sum_i \sum_j c_{ij} t_{ij},$$

where f is a monotone nondecreasing, but otherwise arbitrary, function of the completion times of the jobs, and c_{ij} is defined as in the previous section.

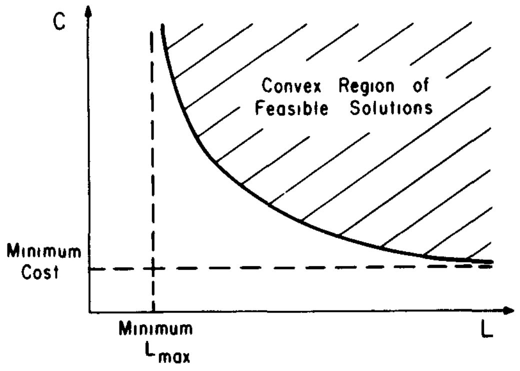


FIG 1

Suppose there is an optimal schedule with completion times $C_1^*, C_2^*, \dots, C_n^*$. We can let these completion times assume the roles of deadlines and solve a linear programming problem of the form described in Section 5. The schedule construction procedure can then be applied to obtain an optimal schedule for which we can bound the number of preemptions as follows. From our previous observations, we have the following theorem:

THEOREM 2. *For any monotone nondecreasing function f and coefficients c_{ij}, there exists an optimal schedule in which the number of preemptions is bounded by*

$$2(m-1)n + \sum_{k=1}^{n} p(m,k) \le O(m^2 n),$$

where $p(m, k)$ is an upper bound on the number of preemptions required for a C_{max}-optimal schedule for k jobs on m unrelated processors.

Theorem 2 can easily be generalized to the case in which we seek to minimize

$$f(S_1, S_2, \dots, S_n, C_1, C_2, \dots, C_n) + \sum_i \sum_j c_{ij} t_{ij},$$

where f is monotone nonincreasing in the starting times $S_1, S_2, \dots, S_n$ and monotone nondecreasing in the completion times $C_1, C_2, \dots, C_n$ We leave details to the reader.

REFERENCES

(Note Reference [5] is not cited in the text)

1 GONZALEZ, T, AND SAHNI, S Open shop scheduling to minimize finish time *J. ACM 23*, 4 (Oct 1976), 665-679

2 GONZALEZ, T, AND SAHNI, S Preemptive scheduling of uniform processor systems *J ACM 25*, 1 (Jan 1978), 92-101

3 SAHNI, S, AND GONZALEZ, T Preemptive scheduling of two unrelated machines Tech. Rep 76-16, Comptr Sci Dept, U of Minnesota, Minneapolis, Minn, Nov 1976

4 MCNAUGHTON, R Sequencing with deadlines and loss functions *Manage Sci 6* (1959), 1-12

5 STERN, H I Minimizing makespan for independent jobs on nonidentical parallel machines—An optimal procedure Tech Rep, Dept of Industrial Eng and Mgt, Ben-Gurion U of the Negev, Beer-Sheva, Israel, March 1976

RECEIVED AUGUST 1976, REVISED DECEMBER 1977