STAN-CS-300

COUNTEREXAMPLE TO A CONJECTURE OF FUJII, KASAMI AND NINOMIYA

by

Marc T. Kaufman

June 1972

Technical Note no. 17

DIGITAL SYSTEMS LABORATORY

Department of Electrical Engineering
Department of Computer Science

Stanford University

Stanford, California

This work was supported in part by the National Science Foundation under Grant GK 23315 and by the Joint -Services Electronic Programs U.S. Army, U.S. Navy and U.S. Air Force under contract N-00014-67-A-0112-0044.

STANFORD UNIVERSITY
Digital Systems Laboratory
Electrical Engineering Department Computer Science Department

COUNTEREXAMPLE TO A CONJECTURE OF FUJII, KASAMI AND NINOMIYA

by

Marc T. Kaufman

ABSTRACT

In a recent paper [1], Fujii, Kasami and Ninomiya presented a procedure for the optimal scheduling of a system of unit length tasks represented as a directed acyclic graph on two identical processors. The authors conjecture that the algorithm can be extended to the case where more than two processors are employed. This note presents a counterexample to that conjecture.

[1] Fujii, M., .T. Kasami and K. Ninomiya, "Optimal Sequencing of Two Equivalent Processors," SIAM J. Appl. Math., Vol. 17, No.4, July 1969, pp. 784-789.

Consider a system consisting of a set of tasks $T = \{T_i\}\ 1 \le i \le n$, and a directed graph G_p representing the precedence relations * among the n tasks. Each task is assumed to require exactly one unit of time. Fujii, Kasami and Ninomiya [1] have presented the following scheduling algorithm, which is optimal for the case of two processors. The algorithm is restated for the case of an arbitrary number of processors:

1. Partition T into a minimal number of subsets, subject to the following restrictions:

   a) The cardinality of each subset must not exceed p, the number of available processors.

   b) All of the members of any subset β in the partition must be compatible (i.e. if $T_i, T_j \in \beta$, $T_i \not> T_j$ and $T_i \not< T_j$).

Let P_1 be the partition be so formed.

2. Form a sequence $\beta_1, \ldots, \beta_k$ of subsets of T, which will correspond to the execution sequence of an optimal schedule, and a sequence of partitions $P_1, P_2 = P_1 - \beta_1, P_3 = P_2 - \beta_2, \ldots,$

   $$P_k = P_{k-1} - \beta_{k-1},\ P_{k+1} = \emptyset$$

   as follows:

   a) Select and remove from P_i a subset β_i of T in which every element of β_i is maximal (has no predecessors in any remaining subset of P_i). Terminate if $P_i = \emptyset$, the empty partition.

   b) If no such subset exists, form a new partition, P_i', in which such a subset does exist. This is always possible for p=2 by Lemma 1 of the paper [1]. By the Lemma, $|P_i'| = |P_i|$. Go to step 2a.

   c) Form $P_{i+1} = P_i - \beta_i$. Go to step 2a.

* We will use the notation $T_i < T_j$ (or $T_j > T_i$) to indicate the relation "T_i preceeds T_j".

In this algorithm, the cardinality of P decreases by 1 at each iteration, so that the sequence $\beta_1, \ldots, \beta_k$ has $k=|P_1|$, which is also a lower bound for the total execution time. Hence this is an optimal sequence.

The following counterexample shows that step 2.b is not always possible when there are 3 processors:

A minimal partition', P, is $\{\{T_1,T_5,T_6\}, \{T_4,T_2,T_3\}\}$, $|P| = 2$. However, the best time which can be achieved is 3, corresponding to a partition (e.g.1 P = $\{\{T_1,T_4\}, \{T_2,T_3,T_5\}, \{T_6\}\}$ with $|P| = 3$.

Hence, Lemma 1 does not generalize for $p > 2$ and the presented algorithm is not extendable to 3 processors.