A Proof that Multiprocessor Scheduling is NP-Complete

Shaun Meyer

Feb 2012

0.1 Partition Definition

INSTANCE: A finite set A and a "size" $s(a) \in Z^+$ for each $a \in A$.

QUESTION: Is there a subset $A' \subseteq A$ such that

$$\sum_{a \in A'} s(a) = \sum_{a \in A - A'} s(a)?$$

Informally, this problem may be viewed as a set A of bowling balls of varying weights. The question is then "Is there a subset of A' (a subset of all our bowling balls) whose weight equals the remaining $(A - A')$?

0.2 Multiprocessor Scheduling Definition

INSTANCE: A finite set A of "tasks," a "length" $l(a) \in Z^+$ for each $a \in A$, a number $m \in Z^+$ of "processors," and a "deadline" $D \in Z^+$.

QUESTION: Is there a partition $A = A_1 \cup A_2 \cup \cdots \cup A_m$ of A into m disjoin sets such that

$$\max \left\{ \sum_{a \in A_i} l(a) : 1 \leq i \leq m \right\} \leq D?$$

Proof(as stated in the text): Restrict to PARTITION by allowing only instances in which $m = 2$ and $D = \frac{1}{2} \sum_{a \in A_i} l(a)$.

Informally this problem may be viewed as a set of tasks whose *run-time is known*. Additionally, we have a deadline which is relevent to prevent process starvation and similar.

The question, then, is given a set of m processors, can we divide up the tasks m ways so that, running concurrently, they complete before the deadline?