On a two class on-line classification problem

Csanád Imreh

In an m-*class classification* problem, there are m classes and a number of different tasks arrive. The decision maker has to assign each task to exactly one of the m classes. This classification carries some cost, and we are to find a classification with minimum cost. In the on-line version we have to assign the tasks to a class immediately at their arrival and without any knowledge on the following tasks. This is a very general model which includes many load balancing and scheduling models from the literature.

In this work we present a mathematical model of the on -line classification problem, and we examine a special on-line two-class classification problem: we consider an on-line classification where the costs result from an off-line scheduling problem. We have two sets $\mathcal{P}$ and $\mathcal{S}$ of identical machines. The jobs arrive one by one. Each job j has two different processing times p_j and s_j, one for each set of machines. We have to decide in an on-line way on which set of machines to schedule each job. Finally, when the stream of jobs has come to an end, the constructed classification is evaluated in the following way: Schedule the jobs assigned to set $\mathcal{P}$ (respectively, the jobs assigned to set $\mathcal{S}$) on the machines of $\mathcal{P}$ (respectively, $\mathcal{S}$) so as to minimize the preemptive makespan, *i.e.* the maximal job completion time. Let C_P (respectively, C_S) denote this optimal makespan. Then the cost of the constructed classification is the maximum of the two makespans. This problem can be considered as a generalised version of the on-line scheduling problem.

We examine a greedy type algorithm for this problem, and we also present and examine an another algorithm which gives better result for the general case. As usually in the theory of on -line algorithms we measure the performance of an on-line algorithm by its competitive ratio. An on-line algorithm is called c-*competitive* if for each input the cost of the assignment produced by the algorithm is at most c times larger than the cost of the optimal assignment.