- **Algorithm**

  First, we guess the optimal value, OPT'. Then we sort the jobs in decreasing order of their sizes. When considering a job j, we assign j to the slowest machine i where the current load of the machine plus job j does not exceed $2 * s_i * OPT'$.

- **The algorithm is feasible.**

  Each job j cannot be assigned to machine i such that $OPT < p_j/s_i$ in the optimal solution, which is trivially true as the span of that one single job will exceed the optimal value. This will define a subset of machines that can possibly schedule the job. This subset cannot be empty because we know an optimal solution exists. Once we make a good estimation, we can schedule the jobs to a machine such that

  $$\frac{p_j}{s_i} \leq OPT$$

  until it is done.

- **The algorithm has an approximation ratio of 2.**

  Because we have proved that the algorithm is feasible, there must be at least one machine m with speed s_m whose current load is smaller than the optimal value, or

  $$\frac{1}{s_m} \sum_{j \in P} p_j \leq OPT$$

  where P denotes all the jobs that have been assigned to m. From the previous section we have

  $$\frac{p_{j+1}}{s_m} \leq OPT$$

  for machine m. We add the above two inequalities for the total workload after assignment,

  $$\frac{1}{s_m} \sum_{j \in P} p_j + \frac{p_{j+1}}{s_m} \leq 2OPT.$$

  The formula holds true throughout the process. Therefore, the algorithm results in a $\mathrm{O}(n)$-approximation whose ratio is 2.

(Discussed with Diqiu Zhou)