One way to do this works as follows: When each job arrives, we put it on the machine that currently ends the soonest. (Note that this determination involves taking into account the speeds of the machines.)

To give a bound on this algorithm, we first give some lower bounds on the optimum makespan T^*. The total time of all jobs is $\sum_j t_j$. Let

$$t = \frac{\sum_j t_j}{m + 2k}.$$

If jobs could be assigned to machines so that each slow machine had a set of jobs summing to less than t, and each fast machine had a set of jobs summing to less than $2t$, then we would have

$$\sum_j t_j < mt + 2kt = \sum_j t_j,$$

a contradiction. Thus, some machine runs for at least t time units, and hence

$$T^* \geq \frac{\sum_j t_j}{m + 2k}.$$

Also, we have

$$T^* \geq \frac{1}{2} t_j,$$

for every job j, since at best it runs on one of the fast machines.

Let $M(r)$ denote the set of jobs assigned to machine r. Consider a machine i that achieves the makespan, and let j be the last job to go on it. Let x denote the time it uses for all jobs before j. (This means that $\sum_{j \in M(i)} t_j$ is equal to x if it's a slow machine, and it is equal to $2x$ if it's a fast machine.) Then at the moment j is added, every slow machine s has $\sum_{j \in M(s)} t_j \geq x$, and and every fast machine f has $\sum_{j \in M(f)} t_j \geq 2x$. Thus we have $\sum_j t_j \geq mx + 2kx$, and hence $T^* \geq x$. Also, $2T^* \geq t_j$.

Since the makespan is achieved by i, it is at most $x + t_j \leq T^* + 2T^* = 3T^*$.

An alternate solution is to simply sort the jobs in decreasing order of size, and then run the Greedy-Balance algorithm as though all machines were slow. We know from the chapter that this would give a $\frac{3}{2}$-approximation if all machines really were slow. However, we are comparing to the optimum as though all its machines are slow; in reality, the optimum's makespan might be half as large as we think, since some of its machines are fast. Thus, this gives a 3-approximation.

[1] ex829.220.704