A NOTE ON ON-LINE ALGORITHMS WITH A PERFORMANCE RATIO LESS THAN $2 - \frac{1}{M}$

R. CHANDRASEKARAN AND P.R. NARAYANAN
THE UNIVERSITY OF TEXAS AT DALLAS

Abstract. In a recent paper [?], an on-line algorithm has been proposed for the makespan problem on identical parallel machines that guarantees a performance ratio better than $2 - \frac{1}{m}$ where m is the number of machines. This note shows that there is a minor error in the proof of lemma 4.3 of this paper which also has an impact on the final performance guarantee. We show how to modify the results in [?] and still have the result that the guarantee is better than $2 - \frac{1}{m}$; but the guarantee we get is different from that in [?].

In a recent paper [?], an on-line algorithm has been proposed for the makespan problem on identical parallel machines that guarantees a performance ratio better than $2 - \frac{1}{m}$ where m is the number of machines. This note shows that there is a minor error in the proof of lemma 4.3 of this paper which also has an impact on the final performance guarantee. We show how to modify the results in [?] and still have the result that the guarantee is better than $2 - \frac{1}{m}$; but the guarantee we get is different from that in [?]. While analyzing the proof of lemma 4.3 in [?], we found the following error.

In the proof of lemma 4.3, the inequality

$$L_1 \geq \alpha L_m \geq \alpha C^*$$

has been used. The second inequality i.e.,

$$\alpha L_m \geq \alpha C^*$$

is true iff

$$L_m \geq C^*$$

Whereas the quantities representing loads $L_1, L_2, ..., L_m$ on the machines have been defined prior to the assignment of the new job x, the quantity C^* represents the optimal makespan including the new job x. We give a counter-example to substantiate our claim that it is possible to have

$$L_m < C^*$$

in some cases. Assume $m = 4$, $\beta = \frac{5}{4}$ and $\alpha = 0.15$. Clearly these values satisfy the conditions on the parameters α, β and m.

Let us trace the assignment of the following sequence of jobs with processing time $4, 4, 4, 6$ and 3.5 units. It is clear that the first four tasks are each assigned to a different machine. After rearrangement of the machines we have $L_1 = L_2 = L_3 = 4$ units and $L_m = L_4 = 6$ units. Thus, when the new job of length 3.5 units is assigned to a machine by the heuristic RLS, the optimal makespan C^* is at least 7.5 units i.e.,

$$C^* \geq 7.5$$

which implies

$$C^* > L_m$$

Hence the proof of lemma 4.3 is incorrect as stated in [?]. Therefore, the inequality represented in (6) of [?] is incorrect. We provide below a correct version of lemma 4.3.

LEMMA 1 (4.3). *When the heuristic RLS leaves step (3), the inequality*

$$\frac{(L_2 + x)}{C^*} \leq 1 + \beta - \alpha \tag{1}$$

is true.

Proof: Case (i) $L_m > C^*$: In this case the inequality

$$\frac{(L_2 + x)}{C^*} < \frac{(m + (m-2)\beta - (m-1)\alpha)}{(m-1)}$$

is true by lemma 4.3 in the paper. It requires simple algebra to show that

$$\frac{(m + (m-2)\beta - (m-1)\alpha)}{(m-1)} \leq 1 + \beta - \alpha$$

Hence (**??**) is true for Case (i)

Case (ii) $L_m \leq C^*$: In this case we derive the inequality (**??**) as follows:

In step(3) of the heuristic,

$$\sim (L_1 + x, L_2, L_3, ..., L_m)$$

is true. This implies $L_1 + x \leq \beta L_m$ and together with $L_1 \geq \alpha L_m$ we get

$$x \leq (\beta - \alpha) L_m \tag{2}$$

Inequality (**??**) together with $C^* \geq L_m \geq L_2$ imply

$$\frac{(L_2 + x)}{C^*} \leq 1 + \beta - \alpha$$

Hence the lemma 4.3. With lemma 4.3 modified, theorem 4.7 of [?] also needs a corresponding modification. The inequality in (20) should be as follows:

$$\frac{(L_2 + x)}{C^*} \leq 1 + \beta - \alpha$$

The above modifications to lemma 4.3 and theorem 4.7, provides a different worst case bound to the performance ratio of the heuristic RLS. Nevertheless, the heuristic RLS still achieves a worst case performance ratio $R^{RLS}(m)$ such that

$$R^{RLS}(m) < 2 - \frac{1}{m}$$

for all $m \geq 4$.

REFERENCES

[1] Gabor Galambos and Gerhard Woeginger: *An On-line Scheduling Heuristic with Better Worst Case Ratio than Graham's List Scheduling*, SIAM J. on Computing, 22, #2, pp. 349-355.