Web supplementary material for the article: Malleable task-graph scheduling with a practical speed-up model

Loris Marchal, Bertrand Simon, Oliver Sinnen, and Frédéric Vivien

We provide here the proof of Lemma 2, which is used in the proof of Theorem 1. We first recall its statement.

Lemma 2. *In any valid schedule S for $\mathcal{J}$,*

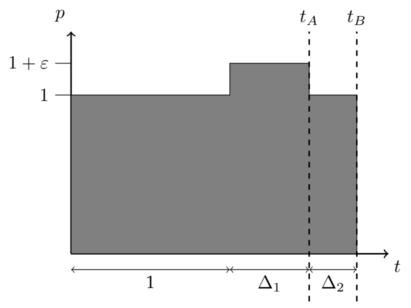
i) each pair of chains $L_{x_i}, L_{\overline{x_i}}$ is completely processed during time interval $[t_i, M - t_i]$,
ii) one of them is started at time t_i and the other one at time $t_i + 1$,
iii) all tasks of both chains are allocated their threshold,
iv) there is no idle time between any two consecutive tasks of each chain.

Proof. The proof is done by induction on i, by carefully checking when the first and last tasks of chains $L_{x_i}, L_{\overline{x_i}}$ may be scheduled, given the resources which are not used by the previous chains and by L_{pro}.

Base case: Consider $i = 1$. The critical path of chains L_{x_1} and $L_{\overline{x_1}}$ is $4n+2m-4i+3 = 4n+2m-1$. With $M = 4n+2m$, both have to start in the interval $[0, 1]$.

The following discussion of the base case is written in general terms (that is for any i) to reuse it in the inductive step, but applies here for $i = 1$, with $t_1 = 0$.

We consider the first task of chain L_{x_i} and the first task of chain $L_{\overline{x_i}}$. Both tasks have weight 1. Let A denote the first of these two tasks to complete (at a time t_A) and let B be the other one (which completes at time t_B). Given the $2(i-1)$ chains already scheduled (none for $i = 1$), the number of processors available during interval $[t_i, t_i + 1]$ is 1 and during interval $[t_i + 1, t_i + 2]$ is $1 + \varepsilon$. A and B both complete at or after time $t_i + 1$. We note $t_A = t_i + 1 + \Delta_1$ and $t_B = t_i + 1 + \Delta_1 + \Delta_2$ ($\Delta_1 \geq 0$ and $\Delta_2 \geq 0$). Note that because of the critical path length of the remaining tasks of both chains and the limited time span, $\Delta_1 \leq 1$ and $\Delta_1 + \Delta_2 \leq 1$. The following figure illustrates the previous notations and the amount of processors available for tasks A and B (note that after time t_A, B may use only $\delta_B = 1$ processor).



Since $w_A + w_B = 2$ work units have to be performed before time t_B, we have

$$1 + \Delta_1(1 + \varepsilon) + \Delta_2 \geq 2$$

and thus $\Delta_2 \geq 1 - \Delta_1(1 + \varepsilon)$ and $t_B \geq t_i + 2 - \Delta_1\varepsilon$.

We symmetrically apply the same reasoning to the last tasks C and D of these two chains, and their *starting* times t_C and t_D, assuming that C is started before D. By setting $t_D = M - t_i - 1 - \Delta'_1$, we get $t_C \leq M - t_i - 2 + \Delta'_1\varepsilon$. We distinguish between two cases, depending on the chains to which A, B, C, and D belong to:

- In the first case, we assume that A and D belong to the same chain. We consider the other chain, containing B and C. Because exactly $4(n-i) + 2m + 1$ tasks need to be processed between these two tasks, we have

  $$t_C \geq t_B + 4(n - i) + 2m + 1$$

  which gives

  $$\Delta'_1\varepsilon \geq 1 - \Delta_1\varepsilon$$

  We have $\Delta_1 \leq 1$ and similarly, $\Delta'_1 \leq 1$. Together with the previous inequality, this gives $\varepsilon \geq 1/2$ which is not possible since $\varepsilon = 1/4n$. Hence B and C cannot belong to the same chain.
- In the second case, we consider that A and C belong to the same chain. Because exactly $4(n-i)+2m+1$ tasks need to be processed between A and C (and between B and D), we have

  $$t_C \geq t_A + 4(n-i) + 2m + 1 \text{ and } t_D \geq t_B + 4(n-i) + 2m + 1$$

  which gives

  $$2m + 4n - t_i - 2 + \Delta'_1\varepsilon \geq t_i + 1 + \Delta_1 + 4(n-i) + 2m + 1$$

- L. Marchal, B. Simon and F. Vivien are with CNRS, INRIA and University of Lyon, LIP, ENS Lyon, 46 allée d'Italie, Lyon, France. E-mail: {loris.marchal,bertrand.simon,frederic.vivien}@ens-lyon.fr
- O. Sinnen is with Dpt. of Electrical and Computer Engineering, University of Auckland, New Zealand E-mail: o.sinnen@auckland.ac.nz

and

$$2m+4n-t_i-1-\Delta_1' \geq t_i+2-\Delta_1\varepsilon+4(n-i)+2m+1,$$

which are simplified (using $t_i = 2(i-1)$) into

$$\Delta_1'\varepsilon \geq \Delta_1 \text{ and } \Delta_1' \leq \Delta_1\varepsilon.$$

This leads to $\Delta_1 \leq \Delta_1\varepsilon^2$. As $0 < \varepsilon < 1$, we have $\Delta_1 = 0$, so $t_A = t_i + 1$. Then, no processor can be allocated to B during $[t_i, t_{i+1}]$.

In other words, one task among the first task of L_{x_i} and the first task of $L_{\overline{x_i}}$ is fully processed during interval $[t_i, t_i + 1]$ and the other one is not processed before $t_i + 1$. Because of its critical path length, the chain starting second must be processed at full speed (each task being allocated a number of processors equal to its threshold) and without idle time in the interval $[t_i+1, M-t_i]$. The last task of the chain starting at time t_i must then be completed at time $M-t_i-1$ and thus this chain must also be processed at full speed and without idle time. This also implies that all available processors are used in the intervals $[t_i, t_i + 2]$ and $[M - t_i - 2, M - t_i]$.

Inductive step: Now assume that the lemma holds for $i - 1$. With $t_1 = 0$ and the inductive property on the last observation we know that no processor is available for chains L_{x_i} and $L_{\overline{x_i}}$ before $2(i - 1)$ and after $M - 2(i - 1)$. The time span available for the remaining chains is thus $4n + 2m - 4i + 4$ while the critical path of chains L_{x_i} and $L_{\overline{x_i}}$ is $4n + 2m - 4i + 3$: these chains cannot be started after $2(i - 1) + 1$ to be completed within the time span. Setting $t_i = 2(i + 1)$ we reuse the above argument about the scheduling of the two chains L_{x_i} and $L_{\overline{x_i}}$, which proves (i)-(iv). $\square$