Scheduling on unspecified heterogeneous distributed resources

Daniel Millot, Christian Parrot

Scheduling on unspecified heterogeneous distributed resources

Daniel Millot and Christian Parrot

Computer Science Dept.
TELECOM sudParis
Évry, France
{Daniel.Millot, Christian.Parrot}@it-sudparis.eu

Abstract

In this paper, we present an adaptive method for scheduling parallel applications on unspecified distributed memory platforms. The presented method can be used to schedule parallel applications when the total workload and the execution parameters (communication speed, available computing power...) are unspecified. When used to schedule divisible load applications according to a master-worker model, this method delivers the workload through multiple rounds. In order to maximize the throughput of each worker, it can prevent both idleness in the use of workers and contentions in the use of the links between master and workers. Before focusing on the proposed scheduling method, the paper recalls the underlying methods on which its development relies. The paper then gives a theoretical analysis of the method before presenting results of simulations obtained with the SimGrid framework on a limited distributed memory platform.

***Keywords*-parallel application; multi-round divisible load scheduling; heterogeneous platform; adaptive scheduling; unspecified distributed memory platform;**

I. Introduction

We consider a master-worker model and a load that can be arbitrarily divided into any number of chunks; each of them can be processed by a worker independently from the others. The problem of scheduling the chunks over the set of workers belongs to the Divisible Load Scheduling (DLS) category. Clearly, such a load distribution can be of any use only when the chunk processing duration is higher than the corresponding communication durations (between master and worker). The resources are supposed to be heterogeneous and we want to maximize the throughput of each worker (independantly from the others). Let us suppose that the durations of communicating and processing a chunk are affine according to chunk size.

On the one hand, we suppose that the master has an unlimited buffering capability and receives a continuous input stream of data to be processed by the workers. The size of the total workload happens to be known only when the last item is acquired by the master; then the so called cleanup phase begins. But, as this size is unknown when scheduling starts, the master must proceed to an iterated distribution as the workload flows in. We call "round" a sequence of consecutive actions leading the master to feed all the workers once and collect the corresponding results. The resulting algorithm is therefore multi-round. After a worker has processed a chunk, the result of the processing is sent back to the master. On the other hand, we suppose that the platform is a distributed memory platform whose communication and computation ressources are inaccurately specified. The available communication speeds, available computation speeds, latencies... that characterize these resources, and that we call execution parameters in the sequel, are supposed constant according to time. The lack of information about both the size of the total workload and the execution parameters of the platform increases the difficulty of scheduling the processing of the chunks efficiently. In this paper, we assume that all the considered workers are used, in a predefined order. Based on the estimates of the execution parameters, the considered workers could be a priori selected in order to maximize the total throughput.

We assume that computation can overlap communication. As stated later on in the paper, we consider a 1-port bidirectional communication model (as used by [1]), which allows a communication from master to worker to overlap a communication from worker to master. With the 1-port communication model, a risk of contention may appear when workers compete to access the master.

Generally, when the input stream delivers a huge work-

load, scheduling methods focus on the steady state phase, and make all workers contribute to the computation as soon as possible, through a startup phase which provides the workers with small enough chunks. For our part, the method we present in this paper adjusts the chunk sizes for the first round in order to prevent idleness and contentions. In order to minimize the duration of the cleanup phase, the workers must complete their work simultaneously (cf DLS theory). Therefore, the scheduling in the method adapts the size of the chunks so that the remaining load on the workers is not too unbalanced; otherwise it would give rise to costly communications to reallocate the chunks during the cleanup phase.

The rest of the paper is organized as follows. The next section is dedicated to related works. Section III is a reminder of the underlying methods on which the AS4DR (Adaptive Scheduling for Distributed Resources) method relies. Section IV presents the proposed method, which overcomes the main drawbacks of the previous ones, and a theoretical analysis of its behaviour. Section V studies in more detail how AS4DR can avoid idleness and contentions. Finally, we conclude and give hints on future work.

II. Related works

Although most general researches on scheduling address applications with intertask dependencies [2], [3], quite a lot of applications [4] are eligible to the Divisible Load Scheduling (DLS) model. This model has been largely studied; it is the first model that enables optimality proofs for scheduling methods in the context we have chosen [4], [5], [6], [7], [8]. It usually deals with problems for which the size of the total workload and the execution parameters are known [9], [10], [8].

Several multi-round methods have been proposed in the literature [11], [10], [9]. On the one hand the iterated distributions of multi-round methods have the advantage (when using a 1-port model) of making the workers active earlier, and on the other hand they have the drawback of increasing the amount of time wasted by latencies. Several strategies for distributing the load to workers have already been studied. First of all, some strategies fix the number of rounds arbitrarily (multi-installment methods [11]). They are well suited to a model with linear costs for homogeneous platforms. Concerning heterogeneous platforms [12], other strategies have been proposed, which are able to take account of the non-linearity [13] of costs when determining the load of the nodes for each round [14]. For instance, the workload which is delivered at each round by the UMR method [10] follows a geometric progression whose common ratio is adjusted so that all the nodes work for the same duration in each round, and so that computation overlaps communication totally. It is proved [10] that this method minimizes the total execution time; provided that we know how to select the best set of nodes to be used. Another method [9] introduces periodicity for the rounds (without imposing any particular value for the period) and also requires that all nodes work during the whole period and that computation overlaps communication totally. It is proved [9] that this method maximizes the amount of load processed by time unit.
Unfortunately none of these methods can be used when the execution parameters are inaccurately specified. To deal with such a lack of information, some methods probe to get the missing information in order to adjust the scheduling [15], [16], [17], [18], [19]. For other methods, like RUMR method [20], the law of probability which governs the fluctuation of the value of the execution parameters is supposed to be known.
To the best of our knowledge, the only methods that can cope with the unspecification of the execution parameters and the unawareness of the total workload are the DA1 method [18] and the OLMR method [19]. Both methods make a very strong assumption: that the master can communicate with the workers without any contentions. These methods make the worker's access to the master become periodic in order to maximize the throughput of each worker. More precisely, from a contention-free situation, the DA1 and OLMR algorithms can keep up an organisation of the successive rounds, we call asymptotically periodic in the sequel, which gives each worker dedicated time frames to dialog with the master. It has been proved [18], [19] that, thanks to the multi-round approach and to the previous rounds history, DA1 and OLMR schedulings can cope with platform heterogeneity and with the lack of knowledge of execution parameters a priori. Drozdowski's results, established for a monodirectionnal 1-port communication model and linear costs [18], have been extended to a bidirectionnal 1-port communication model with affine costs [21]. Whereas DA1's scheduling experiences workers idleness systematically, the OLMR method aims to avoid it by allowing computation to overlap communication, thus improving the throughput of each worker.

However, starting with a contention-free situation is unlikely in practice. Therefore this paper reconsiders the OLMR method and presents the AS4DR scheduler which successfully avoids network contentions as well as workers idleness.

III. Reminder of the underlying methods

The underlying methods are the so called DA1 [18] and OLMR [22] methods. Let's first recall briefly the context in which these methods have been proposed and

studied. Their initial goal was to automatically adapt the scheduling of a divisible load application to the evolution of the execution parameters of the platform over time. Let $\alpha_{w,i}$ be the size of the chunk sent to a worker w for round i. Let τ and $\sigma_{w,i}$ be respectively the wanted and the estimated (measured for DA1 and computed for OLMR) time durations between the sending of a chunk of size $\alpha_{w,i}$ and the reception of the corresponding result by the master. In order to keep the communications contention-free, these methods adapt $\alpha_{w,i}$ so that $\sigma_{w,i}$ converges [18], [19] towards the same reference value τ for all the workers.

Sending subchunks of arbitrarily chosen sizes $\dot{\alpha}_{w,1}$ and $\ddot{\alpha}_{w,1}$ to each worker w for the first round, the OLMR scheduler then sends to worker w, for each round i, two subchunks $\dot{s}$ and $\ddot{s}$ of respective sizes $\dot{\alpha}_{w,i}$ and $\ddot{\alpha}_{w,i}$, such that

$$\dot{\alpha}_{w,i} + \ddot{\alpha}_{w,i} = \alpha_{w,i}; \tag{1}$$

where the value of $\alpha_{w,i}$ is determined by the following computation:

$$\alpha_{w,i} := \alpha_{w,i-1} \frac{\tau}{\sigma_{w,i-1}} \qquad \text{for } i > 1. \tag{2}$$

Dividing the chunks in two parts is enough to apply the principle of the multi-installment strategy [23], in order to allow the computation to overlap the communications between a worker and the master as can be seen in Figure 3. Let us suppose that the ratio between $\dot{\alpha}_{w,i}$ and $\alpha_{w,i}$ is constant, and let us denote θ_w this ratio :

$$\theta_w \equiv \frac{\dot{\alpha}_{w,i}}{\alpha_{w,i}}. \tag{3}$$

As the DA1 method can be considered like a specific case of the OLMR one (when θ_w equals one), only the OLMR method is presented in Figures 1 and 2.

As can be seen in Figure 3, worker round i for worker w is composed of three phases :

- transmission of the data from master to worker, lasting $\dot{D}_{w,i}$ and $\ddot{D}_{w,i}$ for subchunks $\dot{s}$ and $\ddot{s}$ respectively,
- worker computation on the received data, lasting $\dot{C}_{w,i}$ and $\ddot{C}_{w,i}$ for subchunks $\dot{s}$ and $\ddot{s}$ respectively,
- transmission of the computation result from worker to master, lasting $\dot{R}_{w,i}$ and $\ddot{R}_{w,i}$ for subchunks $\dot{s}$ and $\ddot{s}$ respectively,

As, we have supposed that the communication and computation costs are both roundwise affine in the size of the corresponding chunk,

$$\dot{D}_{w,i} = \theta_w \frac{\alpha_{w,i}}{B_w^D} + b_w^D, \quad \ddot{D}_{w,i} = (1-\theta_w) \frac{\alpha_{w,i}}{B_w^D} + b_w^D,$$
$$\dot{C}_{w,i} = \theta_w \frac{\alpha_{w,i}}{F_w} + f_w, \quad \ddot{C}_{w,i} = (1-\theta_w) \frac{\alpha_{w,i}}{F_w} + f_w,$$
$$\dot{R}_{w,i} = \theta_w \frac{\alpha_{w,i}}{B_w^R} + b_w^R, \quad \ddot{R}_{w,i} = (1-\theta_w) \frac{\alpha_{w,i}}{B_w^R} + b_w^R.$$

Where F_w is the available computation speed of worker w.

- Arbitrarily set τ and $(\alpha_{w,1}, \theta_w)_{0 \le w \le W-1}$
- Set $(\dot{\alpha}_{w,1}, \ddot{\alpha}_{w,1})_{0 \le w \le W-1}$ ..... (3), (1)
- As soon as possible, post $\dot{s}$ and $\ddot{s}$ data to each worker

while (the last data item has not been acquired) **do**
- Get a subchunk result from some worker w

if ($\dot{s}$) **then**
- Compute the size of the next $\dot{s}$ and $\ddot{s}$ for worker w ..... (2), (3), (1)
- Post $\dot{s}$ and $\ddot{s}$ data to worker w

end if
end while

Fig. 1. OLMR scheduling : master

while (the last subchunk has not been posted) **do**
- Get a subchunk data from master
- Process the subchunk data
- Post subchunk result to master

end while

Fig. 2. OLMR scheduling : worker

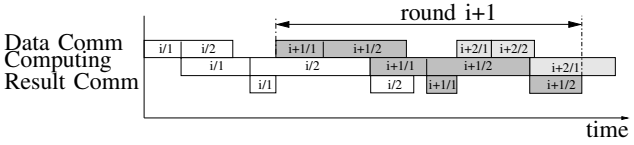

Fig. 3. Overlapping between communication and computation with the OLMR method

This computation speed is relative to the processing of one workload unit. It has to be noticed that the computation speed depends both on the platform and the application, by means of the clock frequency of worker w and the algorithm complexity respectively. Likewise, B_w^D (resp. B_w^R) is the available communication speed (relative to one workload unit) of the link from the master to worker w (resp. from worker w to the master). Finally b_w^D, b_w^R and f_w are the respective latencies for a transfer of data from the master to worker w, for a transfer of result from worker w to the master and for a computation on worker w.

Would a contention-free situation be workable, overlapping between computations and communications for OLMR would help keep workers busy, which DA1 scheduling cannot do. More, under this hypothesis, both underlying methods have ultimately reached another target, that to adapt the scheduling to the evolution of the execution parameters: the automatic taking into account of the platform heterogeneity.

IV. Presentation and analysis of the AS4DR method

A large number of simulations we made (with SimGrid [24]) of the OLMR scheduling convinced us that working out a contention-free situation is unlikely in practice. The main contribution of AS4DR is to address this contentions avoidance problem.

Figures 4 and 5 give the scheduling algorithm of the AS4DR method and Figure 6 shows how computation overlaps communication whith this method. In order to make contentions avoidance easier, AS4DR delays the return of $\ddot{\mathrm{s}}$. As it can be noticed in Figure 6 that, contrary to OLMR scheduling, the result corresponding to the $\ddot{\mathrm{s}}$ subchunk of some round is returned to the master just after the result corresponding to the $\dot{\mathrm{s}}$ subchunks of the next round has been itself returned. As will be stated later on, in Proposition 1, data sending and computation is asymptotically periodic over the successive rounds. Thus delaying the return of $\ddot{\mathrm{s}}$ result makes result return asymptotically periodic too. We will see that this asymptotic periodicity helps contentions avoidance.

- With CIP set τ and $(\alpha_{\mathrm{w},1}, \theta_{\mathrm{w}})_{0\leq \mathrm{w}\leq \mathrm{W}-1}$ ..... (Figure 12)
- Set $(\dot{\alpha}_{\mathrm{w},1}, \ddot{\alpha}_{\mathrm{w},1})_{0\leq \mathrm{w}\leq \mathrm{W}-1}$ ..... (3), (1)
- After an appropriate delay, post $\dot{\mathrm{s}}$ and $\ddot{\mathrm{s}}$ data to each worker

while (the last data item has not been acquired) **do**
- Get a $\dot{\mathrm{s}}$ result from some worker w
- Compute the size of the next $\dot{\mathrm{s}}$ and $\ddot{\mathrm{s}}$ for worker w ..... (4), (6), (3), (1)
- Post $\dot{\mathrm{s}}$ and $\ddot{\mathrm{s}}$ data to worker w
- Get previous $\ddot{\mathrm{s}}$ result from worker w

end while

Fig. 4. AS4DR scheduling : master

while (the last subchunk has not been posted) **do**
- Get a subchunk data from master
- Process the subchunk data

if ($\dot{\mathrm{s}}$) **then**
- Post $\dot{\mathrm{s}}$ and previous $\ddot{\mathrm{s}}$ results to master

end if

end while

Fig. 5. AS4DR scheduling : worker

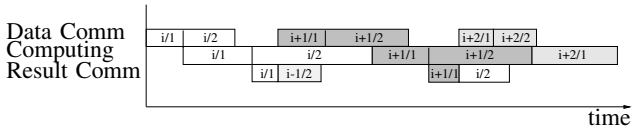

Fig. 6. Overlapping between communication and computation with the AS4DR method

We define

$$\widetilde{\sigma}_{\mathrm{w,i}} \equiv \frac{\dot{\mathrm{C}}_{\mathrm{w,i}} - \mathrm{f_w}}{\theta_{\mathrm{w}}} + 2\mathrm{f_w}. \tag{4}$$

The value of $\widetilde{\sigma}_{\mathrm{w,i}}$ is an extrapolation of the measured value of $\dot{\mathrm{C}}_{\mathrm{w,i}}$, in order to estimate $\mathrm{C_{w,i}}$. So,

$$\widetilde{\sigma}_{\mathrm{w,i}} = \alpha_{\mathrm{w,i}} \frac{1}{\mathrm{F_w}} + 2\mathrm{f_w}. \tag{5}$$

The AS4DR method computes the next chunk size using equation (6) in the same way as DA1 method does using equation (2).

$$\alpha_{\mathrm{w,i}} := \alpha_{\mathrm{w,i-1}} \frac{\tau}{\widetilde{\sigma}_{\mathrm{w,i-1}}} \qquad \text{for } \mathrm{i} > 1. \tag{6}$$

Proposition 1: For each worker w the sequence $\left(\widetilde{\sigma}_{\mathrm{w,i}}\right)_{\mathrm{i}}$ built according to the AS4DR method converges, and

$$\lim_{\mathrm{i}\to+\infty} \widetilde{\sigma}_{\mathrm{w,i}} = \tau.$$

So, if the communications between master and workers are contention-free, both dates of posting data from the master and dates of getting results by the master are asymptotically periodic, with the same period for all the workers. In the absence of idleness of the workers, this period equals, τ.

Proof: From (5) we can deduce that, $\forall \mathrm{i} \in \mathbb{N}$,

$$\widetilde{\sigma}_{\mathrm{w,i+2}} - \widetilde{\sigma}_{\mathrm{w,i+1}} = \left(\alpha_{\mathrm{w,i+2}} - \alpha_{\mathrm{w,i+1}}\right) \frac{1}{\mathrm{F_w}}. \tag{7}$$

Besides, according to (6),

$$\alpha_{\mathrm{w,i+2}} = \alpha_{\mathrm{w,i+1}} \frac{\tau}{\widetilde{\sigma}_{\mathrm{w,i+1}}} \quad \text{and} \quad \alpha_{\mathrm{w,i+1}} = \alpha_{\mathrm{w,i}} \frac{\tau}{\widetilde{\sigma}_{\mathrm{w,i}}}.$$

Hence

$$\widetilde{\sigma}_{\mathrm{w,i+2}} - \widetilde{\sigma}_{\mathrm{w,i+1}} = \tau \left(\frac{\alpha_{\mathrm{w,i+1}}}{\widetilde{\sigma}_{\mathrm{w,i+1}}} - \frac{\alpha_{\mathrm{w,i}}}{\widetilde{\sigma}_{\mathrm{w,i}}} \right) \frac{1}{\mathrm{F_w}}. \tag{8}$$

Due to (7), equation (8) can be rewritten

$$\widetilde{\sigma}_{\mathrm{w,i+2}} - \widetilde{\sigma}_{\mathrm{w,i+1}} = \frac{2\tau \mathrm{f_w}}{\mathrm{F_w} \widetilde{\sigma}_{\mathrm{w,i+1}} \widetilde{\sigma}_{\mathrm{w,i}}} \left(\alpha_{\mathrm{w,i+1}} - \alpha_{\mathrm{w,i}}\right). \tag{9}$$

Now let us suppose that

$$\frac{\tau}{\widetilde{\sigma}_{\mathrm{w,i}}} \geq 1,$$

then it follows from (6) that $\alpha_{\mathrm{w,i+1}} \geq \alpha_{\mathrm{w,i}}$ and thus from (9) that

$$\widetilde{\sigma}_{\mathrm{w,i+2}} \geq \widetilde{\sigma}_{\mathrm{w,i+1}}. \tag{10}$$

Due to (7), inequality (10) can be rewritten

$$\alpha_{\mathrm{w,i+2}} \geq \alpha_{\mathrm{w,i+1}}.$$

This proves that the sequence $(\alpha_{\mathrm{w,i}})_{\mathrm{i}}$ is monotonously growing when $\frac{\tau}{\widetilde{\sigma}_{\mathrm{w,i}}} \geq 1$. A similar demonstration can prove that the sequence $(\alpha_{\mathrm{w,i}})_{\mathrm{i}}$ is monotonously decreasing when $\frac{\tau}{\widetilde{\sigma}_{\mathrm{w,i}}} \leq 1$.

From (7), $\widetilde{\sigma}_{\mathrm{w,i}}$ is a growing function of $\alpha_{\mathrm{w,i}}$. So the sequence $(\widetilde{\sigma}_{\mathrm{w,i}})_{\mathrm{i}}$ is monotonously increasing when $\frac{\tau}{\widetilde{\sigma}_{\mathrm{w,i}}} \geq 1$, and monotonously decreasing when $\frac{\tau}{\widetilde{\sigma}_{\mathrm{w,i}}} \leq 1$.

In order to show that both sequences $(\alpha_{\mathrm{w,i}})_{\mathrm{i}}$ and $(\widetilde{\sigma}_{\mathrm{w,i}})_{\mathrm{i}}$ have limits, it suffices to show that they are either upper bounded when increasing or lower bounded when decreasing. But, when sequence $(\widetilde{\sigma}_{\mathrm{w,i}})_{\mathrm{i}}$ is increasing, sequence $(\alpha_{\mathrm{w,i}})_{\mathrm{i}}$ is increasing too, and from (6), we have $\frac{\tau}{\widetilde{\sigma}_{\mathrm{w,i}}} \geq 1,\ \forall \mathrm{i} \in \mathbb{N}$. In this case, the elements of sequence $(\widetilde{\sigma}_{\mathrm{w,i}})_{\mathrm{i}}$ are upper bounded by τ. Thus, from (5), we have

$$\alpha_{\mathrm{w,i}} \leq \frac{\tau - 2\mathrm{f_w}}{\frac{1}{\mathrm{F_w}}} \qquad \forall \mathrm{i} \in \mathbb{N}.$$

So, if $\frac{\tau}{\widetilde{\sigma}_{\mathrm{w,i}}} \geq 1$, the increasing sequences $(\alpha_{\mathrm{w,i}})_{\mathrm{i}}$ and $(\widetilde{\sigma}_{\mathrm{w,i}})_{\mathrm{i}}$ are upper bounded thus have a limit. A similar argumentation would prove that, if $\frac{\tau}{\widetilde{\sigma}_{\mathrm{w,i}}} \leq 1$, then the decreasing sequences $(\alpha_{\mathrm{w,i}})_{\mathrm{i}}$ et $(\widetilde{\sigma}_{\mathrm{w,i}})_{\mathrm{i}}$ are lower bounded and thus have a limit. From now on, we denote α_{w} and $\widetilde{\sigma}_{\mathrm{w}}$ the respective limits of $(\alpha_{\mathrm{w,i}})_{\mathrm{i}}$ and $(\widetilde{\sigma}_{\mathrm{w,i}})_{\mathrm{i}}$:

$$\alpha_{\mathrm{w}} \equiv \lim_{\mathrm{i}\to\infty} \alpha_{\mathrm{w,i}}, \tag{11}$$

$$\widetilde{\sigma}_{\mathrm{w}} \equiv \lim_{\mathrm{i}\to\infty} \widetilde{\sigma}_{\mathrm{w,i}}.$$

There now remains to show that the limit of $\widetilde{\sigma}_{\mathrm{w,i}}$ does not depend on the chosen worker w, and more precisely that this limit is τ.

So let us show (ab absurdo) that $\widetilde{\sigma}_{\mathrm{w}}$ equals τ. Assume, first, that

$$\widetilde{\sigma}_{\mathrm{w}} < \tau. \tag{12}$$

Let

$$\varepsilon_0 \equiv \alpha_{\mathrm{w}} \left(1 - \frac{\widetilde{\sigma}_{\mathrm{w}}}{\tau}\right).$$

By hypothesis (12),

$$\varepsilon_0 > 0.$$

Thanks to the convergence of $\alpha_{\mathrm{w,i}}$ up towards α_{w}, we can say that $\forall \varepsilon \in \mathbb{R}^{+*},\ \exists \mathrm{N} \in \mathbb{N}^*$ such that i > N entails $\alpha_{\mathrm{w}} - \alpha_{\mathrm{w,i}} < \varepsilon$.

In particular, for

$$\varepsilon = \varepsilon_0$$

a natural i_0 exists such that

$$\alpha_{\mathrm{w}} - \alpha_{\mathrm{w,i_0}} < \alpha_{\mathrm{w}} \left(1 - \frac{\widetilde{\sigma}_{\mathrm{w}}}{\tau}\right),$$

$$\alpha_{\mathrm{w,i_0}} > \alpha_{\mathrm{w}} \frac{\widetilde{\sigma}_{\mathrm{w}}}{\tau}. \tag{13}$$

Besides, the sequence $(\widetilde{\sigma}_{\mathrm{w,i}})_{\mathrm{i}}$ is increasing and upper bounded by $\widetilde{\sigma}_{\mathrm{w}}$. In particular,

$$\widetilde{\sigma}_{\mathrm{w}} \geq \widetilde{\sigma}_{\mathrm{w,i_0}}. \tag{14}$$

Thanks to (6), we have

$$\alpha_{\mathrm{w,i_0+1}} = \alpha_{\mathrm{w,i_0}} \frac{\tau}{\widetilde{\sigma}_{\mathrm{w,i_0}}}.$$

By (13) and (14), this equation allows us to write

$$\alpha_{\mathrm{w,i_0+1}} > \alpha_{\mathrm{w}} \frac{\widetilde{\sigma}_{\mathrm{w}}}{\tau} \frac{\tau}{\widetilde{\sigma}_{\mathrm{w}}},$$

$$> \alpha_{\mathrm{w}}.$$

This contradicts the fact that α_{w} is an upper bound of $\alpha_{\mathrm{w,i}}$ for all $\mathrm{i} \in \mathbb{N}^*$. Therefore, hypothesis (12) is false. Similarly, the hypothesis :

$$\widetilde{\sigma}_{\mathrm{w}} > \tau,$$

leads to a contradiction too. Therefore, $\widetilde{\sigma}_{\mathrm{w}}$ equals τ. ■

Remark 1: It has to be noticed that this result is true either if the master serves the workers in a pre-determined order (round-robin) or if it serves the first worker to be idle systematically (FIFO).

Figure 8 shows a simulation result obtained with SimGrid [24], which illustrates the convergence of $\widetilde{\sigma}_{\mathrm{w,i}}$ towards τ stated in Proposition 1. To get this illustration, we considered a small star-shaped platform with a master connected to 4 workers $(\mathrm{w_i})_{\mathrm{i}=1,4}$ by bi-directionnal links (Figure 7). Table I gives the value of the execution parameters considered for this platform. Times are in seconds, computation and communication speeds are in MB/s.

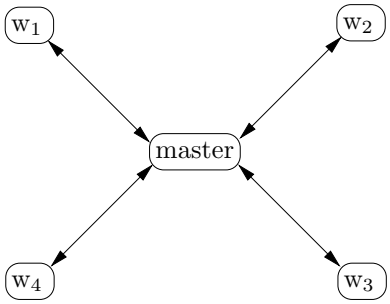

Fig. 7. Star-shaped platform

		w_1	w_2	w_3	w_4
computation	latency	3.0	0.1	0.1	0.1
	speed	0.0935	0.0893	0.0587	0.0617
communication	latency	0.8	0.1	0.9	0.8
master $\longrightarrow$ w_i	speed	0.9025	0.9318	0.9633	0.9143
communication	latency	2.0	0.8	1.0	0.9
master $\longleftarrow$ w_i	speed	0.9025	0.9318	0.9633	0.9143

TABLE I. Execution parameter values

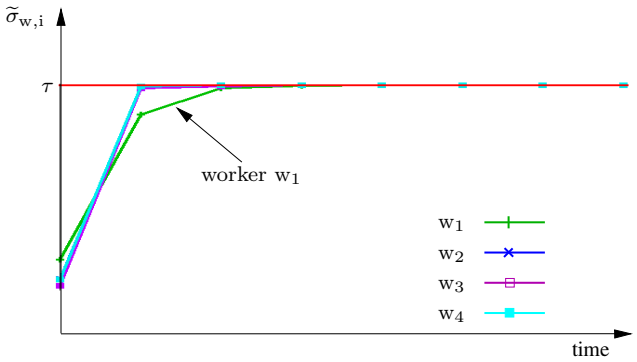

Fig. 8. AS4DR method: convergence of $\widetilde{\sigma}_{w,i}$

From the contents of Table I, it can be claimed that the considered platform is heterogeneous. For instance, it takes worker w_1 3.0935 seconds to process a chunk of unit size whereas it takes worker w_3 only 0.1587 seconds. As a result, the duration to process a chunk can turn out to be about 20 times longer, from one worker to another one.

For this elementary example, $\widetilde{\sigma}_{w,i}$ converges slower for worker 1 than for the other workers because its computation latency is 30 times larger than that of the others (see Table I).

As AS4DR aims to improve the throughput of the DA1 method, let us compare the respective average throughputs δ_w^{DA1} and δ_w^{AS4DR} of both methods on a time interval of duration τ :

$$\delta_w^{DA1} \equiv \frac{\alpha_w^{DA1}}{\tau} \quad \text{and} \quad \delta_w^{AS4DR} \equiv \frac{\alpha_w^{AS4DR}}{\tau}.$$

Proposition 2: Let suppose firstly that the communications between master and workers are contention-free and that there is no intra-round idleness, and secondly that

$$\frac{F_w}{B_w^D} + \frac{F_w}{B_w^R} \geq \frac{3f_w - b_w^D - b_w^R}{\tau - 2f_w}; \tag{15}$$

where $\tau \neq 2f_w$.

If

$$\frac{f_w}{\tau} < \frac{1}{2} \tag{16}$$

then

$$\delta_w^{AS4DR} \geq \delta_w^{DA1}$$

else

$$\delta_w^{AS4DR} \leq \delta_w^{DA1}.$$

Proof: Inequation (15) leads to

$$1 + \frac{F_w}{B_w^D} + \frac{F_w}{B_w^R} \geq 1 + \frac{3f_w - b_w^D - b_w^R}{\tau - 2f_w},$$

$$\frac{\frac{1}{B_w^D} + \frac{1}{F_w} + \frac{1}{B_w^R}}{\frac{1}{F_w}} \geq \frac{\tau - \left(b_w^D - f_w + b_w^R\right)}{\tau - 2f_w}.$$

Thus, if (16) holds, we have :

$$\frac{\tau - 2f_w}{\frac{1}{F_w}} \geq \frac{\tau - \left(b_w^D - f_w + b_w^R\right)}{\frac{1}{B_w^D} + \frac{1}{F_w} + \frac{1}{B_w^R}},$$

$$\alpha_w^{AS4DR} \geq \alpha_w^{DA1},$$

$$\delta_w^{AS4DR} \geq \delta_w^{DA1}.$$

Conversely, if (16) does not hold, then

$$\delta_w^{AS4DR} \leq \delta_w^{DA1}.$$

■

Remark 2: Practically, the duration τ, typically targeted for a round, is much larger than that of the computation latency f_w. Therefore, inequation (16) should generally hold and AS4DR throughput be larger than DA1's one.

Remark 3: Inequation (15) states that the smaller the communication speeds and the larger the communication latencies, the more efficient the overlapping between communication and computation. Although latencies have to be considered in order to validate the convergence result for AS4DR, it should be clear that multi-round scheduling is only interesting when latencies are negligible compared to the duration τ of rounds. In this case, hypothesis (15) can be rewritten:

$$\frac{F_w}{B_w^D} + \frac{F_w}{B_w^R} \geq 0$$

which is always true.

In order to measure the possible throughput improvement provided by AS4DR over DA1, simulations have been conducted with SimGrid [24]. To get a deeper insight into the respective performance of both methods, we define γ:

$$\gamma \equiv 100.\frac{x_{AS4DR} - x_{DA1}}{x_{DA1}};$$

for each scheduling characteristic under consideration (idleness, throughput...), where x_{DA1} and x_{AS4DR} are the measured values for DA1 and AS4DR respectively. In order to be able to start DA1 without contention, the CIP algorithm, which is part of AS4DR (see section V), had to be used (with θ = 1). Table II gives the results obtained

when each method was used to schedule a 20000 MB workload on the previous simple platform (cf Table I). Idleness is in seconds, throughput is in MB/s.

	DA1	AS4DR	γ
Idleness of the workers	52858.70	0.00	+100.00%
Throughput	0.2506	0.3020	+20.50%

TABLE II. Comparison between DA1 and AS4DR

These results illustrate Proposition 2: with the AS4DR method, the measured throughput is 20.5% higher than the one obtained with the DA1 method, coupled with the CIP algorithm (in order to avoid contentions). As AS4DR totally prevents idleness, workers are fully busy computing, and thus no other improvement of the throughput can be hoped: for this example, the scheduling is optimal.

To the best of our knowledge, it is not possible to avoid contentions when using the OLMR method, even if this method is coupled with the CIP algorithm (because of the unpredictability of the date for the return of the subchunks s̈ with OLMR). As the contention avoidance is a necessary condition to use the OLMR method [22], the comparison of AS4DR with the OLMR method cannot be made.

V. Towards prevention of idleness and contentions

In spite of their theoretical interest, intensive SimGrid [24] simulations have revealed a huge drawback of the DA1 and OLMR methods: the difficulty to work out values for τ and $\alpha_{w,1}$ that lead to a contention-free starting situation. More, contention prevention is not enough to give a practical interest to the AS4DR method, as it could experience workers idleness. So the preliminary step CIP (for Contentions and Idleness Prevention) described in this section has been introduced, in order to try to avoid both pitfalls. Firstly, we are going to exhibit necessary and sufficient conditions for the prevention of workers idleness. Then, we are going to expose necessary and sufficient conditions to avoid contentions on the links between master and workers. Finally, we will see how the CIP algorithm takes these two results into account to avoid contentions and idleness during the AS4DR scheduling first round.

Figures 9 and 10 illustrate the two kinds of idleness we can meet: inter-round idleness and intra-round idleness.

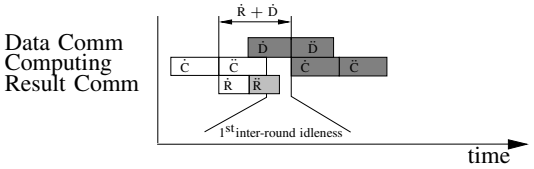

Fig. 9. Example of inter-round idleness with AS4DR

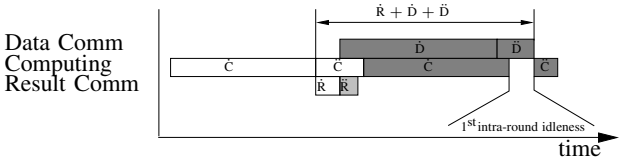

Fig. 10. Example of intra-round idleness with AS4DR

Proposition 3: Let us respectively define, $\forall$ i, $\theta^{\min}_{w,i}$ and $\theta^{\max}_{w,i}$ as:

$$\theta^{\min}_{w,i+1} \equiv \frac{D_{w,i+1} - \left(b^D_w + f_w\right)}{D_{w,i+1} + C_{w,i+1} - 2\left(b^D_w + f_w\right)}, \quad (17)$$

$$\theta^{\max}_{w,i+1} \equiv \frac{C_{w,i} - \left(b^D_w + f_w + b^R_w\right)}{D_{w,i+1} + C_{w,i} + R_{w,i} - 2\left(b^D_w + f_w + b^R_w\right)}. \quad (18)$$

The AS4DR method prevents idleness, if and only if, for each worker w, and $\forall$ i

$$\theta^{\max}_{w,i+1} \geq \theta_w \geq \theta^{\min}_{w,i+1}. \quad (19)$$

Proof: If $\theta^{\max}_{w,i+1} \geq \theta_w$, then

$$\frac{\frac{\alpha_{w,i}}{F_w} + f_w - b^R_w - b^D_w}{\frac{\alpha_{w,i}}{F_w} + \frac{\alpha_{w,i+1}}{B^D_w} + \frac{\alpha_{w,i}}{B^R_w}} \geq \theta_w,$$

$$(1-\theta_w)\frac{\alpha_{w,i}}{F_w} + f_w \geq \left(\theta_w \frac{\alpha_{w,i+1}}{B^D_w} + b^D_w\right) + \left(\theta_w \frac{\alpha_{w,i}}{B^R_w} + b^R_w\right).$$

Therefore,

$$\ddot{C}_{w,i} \geq \dot{R}_{w,i} + \dot{D}_{w,i+1},$$

and no inter-round idleness can occur; and reciprocally.

Now, if we suppose furthermore that $\theta_w \geq \theta^{\min}_{w,i+1}$, then

$$\theta_w \geq \frac{\frac{\alpha_{w,i+1}}{B^D_w} + b^D_w - f_w}{\alpha_{w,i+1}\left(\frac{1}{F_w} + \frac{1}{B^D_w}\right)},$$

$$\theta_{\mathrm{w}} \frac{\alpha_{\mathrm{w,i+1}}}{\mathrm{F_w}} + \mathrm{f_w} \geq (1-\theta_{\mathrm{w}}) \frac{\alpha_{\mathrm{w,i+1}}}{\mathrm{B_w^D}} + \mathrm{b_w^D}.$$

Thus

$$\dot{\mathrm{C}}_{\mathrm{w,i+1}} \geq \ddot{\mathrm{D}}_{\mathrm{w,i+1}}.$$

As $\theta^{\max}_{\mathrm{w,i+1}} \geq \theta_{\mathrm{w}}$, then $\ddot{\mathrm{C}}_{\mathrm{w,i}} \geq \dot{\mathrm{R}}_{\mathrm{w,i}} + \dot{\mathrm{D}}_{\mathrm{w,i+1}}$ holds too. Summing these two inequalities leads to :

$$\ddot{\mathrm{C}}_{\mathrm{w,i}} + \dot{\mathrm{C}}_{\mathrm{w,i+1}} \geq \dot{\mathrm{R}}_{\mathrm{w,i}} + \dot{\mathrm{D}}_{\mathrm{w,i+1}} + \ddot{\mathrm{D}}_{\mathrm{w,i+1}},$$

and so no intra-round idleness can occur; and reciprocally. ■

Figure 11 shows the model of asymptotic τ-periodic

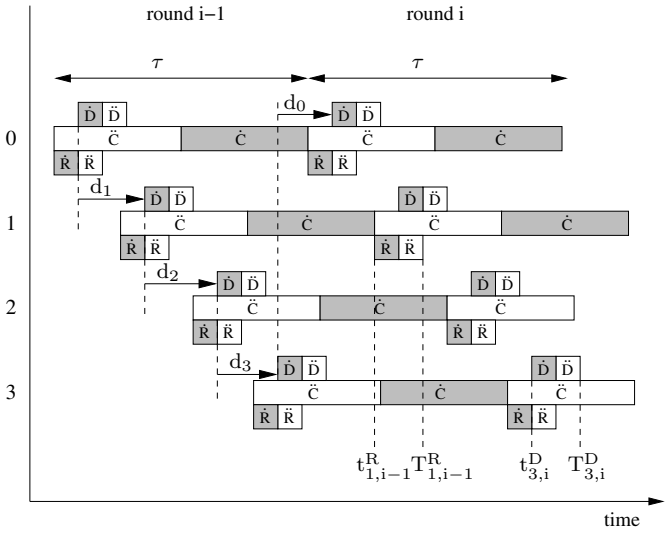

Fig. 11. contention-free asymptotic τ-periodic schedule

(thus round-robin) scheduling we are looking for, in the case of a four workers platform. It illustrates the definition of instants $\mathrm{t^R_{w,i}}$, $\mathrm{T^R_{w,i}}$, $\mathrm{t^D_{w,i}}$ and $\mathrm{T^D_{w,i}}$.

To make the instant each worker accesses to the master far enough from the instants the others access too, time delays $\mathrm{d_w}$ are introduced before posting the very first subchunk to each worker w. We are going to see that introducing such laxity in the scheduling makes the AS4DR method more compliant with the errors on the estimate of the execution parameters.

Proposition 4: The AS4DR method prevents contentions, if and only if, $\forall$ i,

$$\mathrm{d_w} \geq \max\Big(\dot{\mathrm{D}}_{\mathrm{w-1,i+1}} + \ddot{\mathrm{D}}_{\mathrm{w-1,i+1}}, \ddot{\mathrm{R}}_{\mathrm{w-1,i-1}} + \dot{\mathrm{R}}_{\mathrm{w,i}}\Big) \quad (20)$$

for each worker w (modulo W).

Proof: There are no contentions if and only if, for any worker and any round, the instant for data posting from the master to a worker occurs after the end of all past emissions of data to other workers and before the beginning of all future receptions of results from other workers. In fact, due to the spontaneous round-robin order, this condition amounts, for worker w and all round i, to:

$$\mathrm{T^R_{w-1,i}} \leq \mathrm{t^R_{w,i}} \text{ and } \mathrm{T^D_{w-1,i}} \leq \mathrm{t^D_{w,i}}. \quad (21)$$

Hence, for all w (modulo W) system (21) can be rewritten as:

$$\mathrm{d_w} \geq \dot{\mathrm{D}}_{\mathrm{w-1,i+1}} + \ddot{\mathrm{D}}_{\mathrm{w-1,i+1}},$$
$$\mathrm{d_w} \geq \ddot{\mathrm{R}}_{\mathrm{w-1,i-1}} + \dot{\mathrm{R}}_{\mathrm{w,i}}.$$

Thus, for 0 ≤ w ≤ W-1 (modulo W),

$$\mathrm{d_w} \geq \max\left(\dot{\mathrm{D}}_{\mathrm{w-1,i+1}} + \ddot{\mathrm{D}}_{\mathrm{w-1,i+1}}, \ddot{\mathrm{R}}_{\mathrm{w-1,i-1}} + \dot{\mathrm{R}}_{\mathrm{w,i}}\right).$$

■

The larger the $\mathrm{d_w}$ time intervals, the lower the risk of contentions in case of inaccurate execution parameters estimation.

When launching the AS4DR scheduler (see Figure 4), the value for τ and three sets of values $(\alpha_{\mathrm{w},1})_{0\leq \mathrm{w} \leq \mathrm{W}-1}$, $(\theta_{\mathrm{w}})_{0\leq \mathrm{w} \leq \mathrm{W}-1}$ and $(\mathrm{d_w})_{0\leq \mathrm{w} \leq \mathrm{W}-1}$ must be determined in order to start the scheduling without contention. This is done by CIP algorithm from an estimate of the execution parameters, as shown by Figure 12.

Let $\overline{\mathrm{B_w^D}}$ (resp. $\overline{\mathrm{F_w}}$, $\overline{\mathrm{B_w^R}}$, $\overline{\mathrm{b_w^D}}$, $\overline{\mathrm{f_w}}$ and $\overline{\mathrm{b_w^R}}$) denote this estimate of $\mathrm{B_w^D}$ (resp. $\mathrm{F_w}$, $\mathrm{B_w^R}$, $\mathrm{b_w^D}$, $\mathrm{f_w}$ and $\mathrm{b_w^R}$).

Depending on the inaccuracy in the estimation of the execution parameters, the delay $\mathrm{d_w}$, computed by the CIP algorithm to be introduced in the scheduling, is defined as follows.

$$\mathrm{d_w} := (1+\lambda_{\mathrm{w}}) \max\Big(\overline{\dot{\mathrm{D}}_{\mathrm{w-1}}} + \overline{\ddot{\mathrm{D}}_{\mathrm{w-1}}}, \overline{\ddot{\mathrm{R}}_{\mathrm{w-1}}} + \overline{\dot{\mathrm{R}}_{\mathrm{w}}}\Big) \text{ (modulo W)}; \quad (22)$$

where λ_{w} is a positive constant factor for worker w, and $\overline{\dot{\mathrm{D}}_{\mathrm{w}}}$ (resp. $\overline{\ddot{\mathrm{D}}_{\mathrm{w}}}$, $\overline{\dot{\mathrm{R}}_{\mathrm{w}}}$ and $\overline{\ddot{\mathrm{R}}_{\mathrm{w}}}$) are estimates of $\dot{\mathrm{D}}_{\mathrm{w},1}$ (resp. $\ddot{\mathrm{D}}_{\mathrm{w},1}$, $\dot{\mathrm{R}}_{\mathrm{w},1}$ and $\ddot{\mathrm{R}}_{\mathrm{w},1}$) thanks to

$$\overline{\dot{\mathrm{D}}_{\mathrm{w}}} := \theta_{\mathrm{w}} \alpha_{\mathrm{w}} \frac{1}{\overline{\mathrm{B_w^D}}} + \overline{\mathrm{b_w^D}}, \quad (23)$$

$$\overline{\ddot{\mathrm{D}}_{\mathrm{w}}} := (1-\theta_{\mathrm{w}}) \alpha_{\mathrm{w}} \frac{1}{\overline{\mathrm{B_w^D}}} + \overline{\mathrm{b_w^D}}, \quad (24)$$

$$\overline{\dot{\mathrm{R}}_{\mathrm{w}}} := \theta_{\mathrm{w}} \alpha_{\mathrm{w}} \frac{1}{\overline{\mathrm{B_w^R}}} + \overline{\mathrm{b_w^R}}, \quad (25)$$

$$\overline{\ddot{\mathrm{R}}_{\mathrm{w}}} := (1-\theta_{\mathrm{w}}) \alpha_{\mathrm{w}} \frac{1}{\overline{\mathrm{B_w^R}}} + \overline{\mathrm{b_w^R}}. \quad (26)$$

For relation (20) to hold, the greater this inaccuracy on the estimate of the execution parameters, the larger λ_{w} should be chosen.

The computation of $\overline{\dot{\mathrm{D}}_{\mathrm{w}}}$, $\overline{\ddot{\mathrm{D}}_{\mathrm{w}}}$, $\overline{\dot{\mathrm{R}}_{\mathrm{w}}}$ and $\overline{\ddot{\mathrm{R}}_{\mathrm{w}}}$ requires the

values of α_w and θ_w:

$$\alpha_{\mathrm{w}} := \frac{\tau - 2\overline{\mathrm{f_w}}}{\frac{1}{\overline{\mathrm{F_w}}}} \tag{27}$$

$$\theta_{\mathrm{w}} := \phi_{\mathrm{w}}\theta_{\mathrm{w}}^{\max} + (1-\phi_{\mathrm{w}})\,\theta_{\mathrm{w}}^{\min}; \tag{28}$$

where $\theta_{\mathrm{w}}^{\min}$ and $\theta_{\mathrm{w}}^{\max}$ are respectively deduced from (17) and (18) as follows

$$\theta_{\mathrm{w}}^{\min} := \frac{\frac{\alpha_{\mathrm{w}}}{\overline{\mathrm{B_w^D}}} + \overline{\mathrm{b_w^D}} - \overline{\mathrm{f_w}}}{\alpha_{\mathrm{w}}\left(\frac{1}{\overline{\mathrm{F_w}}} + \frac{1}{\overline{\mathrm{B_w^D}}}\right)},$$

$$\theta_{\mathrm{w}}^{\max} := \frac{\frac{\alpha_{\mathrm{w}}}{\overline{\mathrm{F_w}}} + \overline{\mathrm{f_w}} - \overline{\mathrm{b_w^R}} - \overline{\mathrm{b_w^D}}}{\alpha_{\mathrm{w}}\left(\frac{1}{\overline{\mathrm{F_w}}} + \frac{1}{\overline{\mathrm{B_w^D}}} + \frac{1}{\overline{\mathrm{B_w^R}}}\right)}.$$

When

$$\phi_{\mathrm{w}} = 0.5, \tag{29}$$

the risks of intra-round idleness and inter-round idleness are well balanced. But the value of θ_{w} could be obtained by averaging $\theta_{\mathrm{w}}^{\min}$ and $\theta_{\mathrm{w}}^{\max}$ with other weights than 0.5 (see Remark 5).

- Estimation of $\left(\mathrm{B_w^D},\ \mathrm{F_w},\ \mathrm{B_w^R},\ \mathrm{b_w^D},\ \mathrm{f_w},\ \mathrm{b_w^R}\right)_{0\le \mathrm{w}\le \mathrm{W}-1}$
- Set $(\phi_{\mathrm{w}})_{0\le \mathrm{w}\le \mathrm{W}-1}$ ..... (29)
- Set $(\lambda_{\mathrm{w}})_{0\le \mathrm{w}\le \mathrm{W}-1}$ ..... (36)
- $\tau := 2 \max\limits_{0\le \mathrm{s}\le \mathrm{W}-1} \overline{\mathrm{f_w}}$

repeat

- $\tau := \tau + \widehat{\tau}$
- Set $(\alpha_{\mathrm{w}})_{0\le \mathrm{w}\le \mathrm{W}-1}$ ..... (27)
- Set $(\theta_{\mathrm{w}})_{0\le \mathrm{w}\le \mathrm{W}-1}$ ..... (28)
- Set $\left(\dot{\mathrm{D}}_{\mathrm{w}},\ \ddot{\mathrm{D}}_{\mathrm{w}},\ \dot{\mathrm{R}}_{\mathrm{w}},\ \ddot{\mathrm{R}}_{\mathrm{w}}\right)_{0\le \mathrm{w}\le \mathrm{W}-1}$ ..... (23),(24),(25),(26)
- Set $(\mathrm{d_w})_{0\le \mathrm{w}\le \mathrm{W}-1}$ ..... (22)

until $\left(\tau \ge \sum_{\mathrm{w}=0}^{\mathrm{W}-1} \mathrm{d_w}\right)$

Fig. 12. CIP algorithm

Besides, the time intervals $\mathrm{d_w}$ should allow all the workers to be served during the first round, i.e. within a τ period. Thus τ must verify:

$$\tau \ge \sum_{\mathrm{w}=0}^{\mathrm{W}-1} \mathrm{d_w}. \tag{30}$$

So, starting from a computed initial value, an iterative process increments τ with an arbitrarily fixed value $\widehat{\tau}$, then computes $(\alpha_{\mathrm{w}})_{0\le \mathrm{w}\le \mathrm{W}-1}$ and $(\mathrm{d_w})_{0\le \mathrm{w}\le \mathrm{W}-1}$ successively, thanks to (27) and (22) respectively, until (30) holds.

Once CIP is processed, proper AS4DR scheduling starts with a first round which sets the initial time-lags between successive round beginnings, according to the values $(\mathrm{d_w})_{0\le \mathrm{w}\le \mathrm{W}-1}$ previously computed by CIP.

For the next rounds, the AS4DR method, thanks to assignment (6), helps maintain the duration of each round close to the reference value τ. Then the algorithm loops until the end of the flow. In order to state the next Proposition, we define the value Λ_{w} for each worker w as

$$\Lambda_{\mathrm{w}} \equiv \frac{1}{\overline{\mathrm{F_w}} \max\left(\mathrm{K_w} + \frac{1}{\overline{\mathrm{B_w^D}}},\ \frac{1}{\overline{\mathrm{B_w^R}}}\right)} - 1;$$

$$\text{where } \mathrm{K_w} \equiv \frac{1}{\overline{\mathrm{B_w^R}}}\left(\phi_{\mathrm{w}} \frac{1}{1+\overline{\mathrm{F_w}}\left(\frac{1}{\overline{\mathrm{B_w^D}}} + \frac{1}{\overline{\mathrm{B_w^R}}}\right)} + (1-\phi_{\mathrm{w}})\,\frac{1}{1+\frac{\overline{\mathrm{B_w^D}}}{\overline{\mathrm{F_w}}}}\right).$$

Proposition 5: Let us assume that for each worker w

$$\lambda_{\mathrm{w}} \le \min\left(\Lambda_{\mathrm{w}}, \Lambda_{\mathrm{w}-1}\right) \quad (\text{modulo W}). \tag{31}$$

Then the CIP preliminary step provides a τ value and sets of values $(\theta_{\mathrm{w}})_{0\le \mathrm{w}\le \mathrm{W}-1}$, $(\alpha_{\mathrm{w}})_{0\le \mathrm{w}\le \mathrm{W}-1}$ and $(\mathrm{d_w})_{0\le \mathrm{w}\le \mathrm{W}-1}$ that allow the AS4DR scheduling to start with neither contention nor idleness.

Proof: In order to make this proof easier to read, we drop the overlined notation for the estimate values in the body of the proof.

As the value of ϕ_{w} is chosen within the interval [0, 1], θ_{w} satisfies the hypotheses of Proposition 3; therefore the AS4DR method avoids idleness during the first round.

As both τ and $\sum_{\mathrm{w}=0}^{\mathrm{W}-1} \mathrm{d_w}$ are affine functions of α_{w}, showing that τ increases quicker than $\sum_{\mathrm{w}=0}^{\mathrm{W}-1} \mathrm{d_w}$ does, should ensure that τ will finally dominate $\sum_{\mathrm{w}=0}^{\mathrm{W}-1} \mathrm{d_w}$ and the loop will end; and thus prove the result. Due to hypothesis (31), for each worker w:

$$\lambda_{\mathrm{w}} \le \Lambda_{\mathrm{w}} \quad \text{and} \quad \lambda_{\mathrm{w}+1} \le \Lambda_{\mathrm{w}} \quad (\text{modulo W}).$$

Thus,

$$\max\left(\lambda_{\mathrm{w}}, \lambda_{\mathrm{w}+1}\right) \le \Lambda_{\mathrm{w}},$$

$$\le \frac{1}{\mathrm{F_w} \max\left(\mathrm{K_w} + \frac{1}{\mathrm{B_w^D}},\ \frac{1}{\mathrm{B_w^R}}\right)} - 1.$$

Therefore

$$\left(1 + \max\left(\lambda_{\mathrm{w}}, \lambda_{\mathrm{w}+1}\right)\right) \max\left(\mathrm{K_w} + \frac{1}{\mathrm{B_w^D}}, \frac{1}{\mathrm{B_w^R}}\right) \le \frac{1}{\mathrm{F_w}}. \tag{32}$$

Since

$$\frac{\partial\left(\dot{\mathrm{D}}_{\mathrm{w}}+\ddot{\mathrm{D}}_{\mathrm{w}}\right)}{\partial\alpha_{\mathrm{w}}}=\frac{1}{\mathrm{B}_{\mathrm{w}}^{\mathrm{D}}},$$
$$\frac{\partial\dot{\mathrm{R}}_{\mathrm{w}}}{\partial\alpha_{\mathrm{w}}}=\mathrm{K}_{\mathrm{w}},$$
$$\frac{\partial\ddot{\mathrm{R}}_{\mathrm{w}}}{\partial\alpha_{\mathrm{w}}}=\frac{1}{\mathrm{B}_{\mathrm{w}}^{\mathrm{R}}}-\mathrm{K}_{\mathrm{w}},$$
$$\text{and }\frac{\partial\tau}{\partial\alpha_{\mathrm{w}}}=\frac{1}{\mathrm{F}_{\mathrm{w}}}.$$

then inequality (32) can be rewritten

$$(1+\max(\lambda_{\mathrm{w}},\lambda_{\mathrm{w}+1}))\left(\frac{\partial\dot{\mathrm{R}}_{\mathrm{w}}}{\partial\alpha_{\mathrm{w}}}+\max\left(\frac{\partial\left(\dot{\mathrm{D}}_{\mathrm{w}}+\ddot{\mathrm{D}}_{\mathrm{w}}\right)}{\partial\alpha_{\mathrm{w}}},\ \frac{\partial\ddot{\mathrm{R}}_{\mathrm{w}}}{\partial\alpha_{\mathrm{w}}}\right)\right)\leq\frac{\partial\tau}{\partial\alpha_{\mathrm{w}}}. \quad (33)$$

Taking into account the definition of $\mathrm{d_w}$, given by (22), the derivation of $\sum_{\mathrm{w}=0}^{\mathrm{W}-1}\mathrm{d_w}$ with respect to α_{w} can potentially lead to six values according to the α_{w} dependency of the different terms of the sum :

$$0\ ,\ (1+\lambda_{\mathrm{w}+1})\frac{\partial\left(\dot{\mathrm{D}}_{\mathrm{w}}+\ddot{\mathrm{D}}_{\mathrm{w}}\right)}{\partial\alpha_{\mathrm{w}}},$$
$$(1+\lambda_{\mathrm{w}})\frac{\partial\dot{\mathrm{R}}_{\mathrm{w}}}{\partial\alpha_{\mathrm{w}}}\ ,\ (1+\lambda_{\mathrm{w}+1})\frac{\partial\ddot{\mathrm{R}}_{\mathrm{w}}}{\partial\alpha_{\mathrm{w}}},$$
$$(1+\lambda_{\mathrm{w}})\frac{\partial\dot{\mathrm{R}}_{\mathrm{w}}}{\partial\alpha_{\mathrm{w}}}+(1+\lambda_{\mathrm{w}+1})\frac{\partial\left(\dot{\mathrm{D}}_{\mathrm{w}}+\ddot{\mathrm{D}}_{\mathrm{w}}\right)}{\partial\alpha_{\mathrm{w}}},$$
$$(1+\lambda_{\mathrm{w}})\frac{\partial\dot{\mathrm{R}}_{\mathrm{w}}}{\partial\alpha_{\mathrm{w}}}+(1+\lambda_{\mathrm{w}+1})\frac{\partial\ddot{\mathrm{R}}_{\mathrm{w}}}{\partial\alpha_{\mathrm{w}}}.$$

As all the derivatives have positive values, due to the cost model, the greater of the last two of these six values can be an upper bound for the derivative of $\sum_{\mathrm{w}=0}^{\mathrm{W}-1}\mathrm{d_w}$ with respect to α_{w}.
So,

$$\frac{\partial\sum_{\mathrm{w}=0}^{\mathrm{W}-1}\mathrm{d_w}}{\partial\alpha_{\mathrm{w}}}\leq(1+\max(\lambda_{\mathrm{w}},\lambda_{\mathrm{w}+1}))\left(\frac{\partial\dot{\mathrm{R}}_{\mathrm{w}}}{\partial\alpha_{\mathrm{w}}}+\max\left(\frac{\partial\left(\dot{\mathrm{D}}_{\mathrm{w}}+\ddot{\mathrm{D}}_{\mathrm{w}}\right)}{\partial\alpha_{\mathrm{w}}},\ \frac{\partial\ddot{\mathrm{R}}_{\mathrm{w}}}{\partial\alpha_{\mathrm{w}}}\right)\right). \quad (34)$$

From inequalities (33) and (34), we deduce

$$\frac{\partial\sum_{\mathrm{w}=0}^{\mathrm{W}-1}\mathrm{d_w}}{\partial\alpha_{\mathrm{w}}}\leq\frac{\partial\tau}{\partial\alpha_{\mathrm{w}}}. \quad (35)$$

So, under the hypotheses of Proposition 5, τ as a function of α_{w} increases quicker than $\sum_{\mathrm{w}=0}^{\mathrm{W}-1}\mathrm{d_w}$. Hence, incrementing α_{w} for each worker w iteratively, as CIP does, leads the loop to an end. ■

Thanks to Proposition 5, the CIP algorithm sets the λ_{w} values as follows:

$$\lambda_{\mathrm{w}}:=\min(\Lambda_{\mathrm{w}},\Lambda_{\mathrm{w}-1})\quad(\text{modulo W}). \quad (36)$$

If the chosen values for λ_{w} are too small for inequation (20) to hold, then contentions avoidance during AS4DR is no more guaranteed (Proposition 4). So a condition should be added to the main loop of AS4DR in order to exit in case of contention, and re-run the scheduling with a reduced number of workers which makes equation (30) easier to satisfy. The optimal selection among the workers through this reduction process remains to be tackled.

Remark 4: To get an acceptable θ_{w} value, the interval $\left[\theta_{\mathrm{w}}^{\min},\theta_{\mathrm{w}}^{\max}\right]$ must not be empty. This is the case, for example, when latencies are negligible compared to effective computation and communication durations and when the available communication speed is greater than the available computation one. Indeed:

$$\overline{\mathrm{F_w}}\leq\overline{\mathrm{B_w^D}},$$
$$\frac{1}{\overline{\mathrm{F_w}}}\left(\frac{1}{\overline{\mathrm{F_w}}}+\frac{1}{\overline{\mathrm{B_w^R}}}\right)\geq\frac{1}{\left(\overline{\mathrm{B_w^D}}\right)^2},$$
$$\frac{1}{\overline{\mathrm{F_w}}}\left(\frac{1}{\overline{\mathrm{F_w}}}+\frac{1}{\overline{\mathrm{B_w^D}}}+\frac{1}{\overline{\mathrm{B_w^R}}}\right)\geq\frac{1}{\overline{\mathrm{B_w^D}}}\left(\frac{1}{\overline{\mathrm{F_w}}}+\frac{1}{\overline{\mathrm{B_w^D}}}\right),$$
$$\theta_{\mathrm{w}}^{\max}\geq\theta_{\mathrm{w}}^{\min}.$$

Remark 5: By analysing the evolution of $\mathrm{K_w}$ according to $\phi_{\mathrm{w}}\in[0,1]$, it can be proved that Λ_{w} is an increasing function of ϕ_{w} if and only if

$$\frac{1}{\overline{\mathrm{F_w}}}\geq\frac{1}{\overline{\mathrm{B_w^D}}}\sqrt{1+\frac{\overline{\mathrm{B_w^D}}}{\overline{\mathrm{B_w^R}}}},$$

and decreasing otherwise. So, in order to reduce the risk of contentions, the value of ϕ_{w} can be adjusted to increase the value of Λ_{w} (Proposition 5). But, such an adjustment brings θ_{w} closer to the boudary of the interval $\left[\theta_{\mathrm{w}}^{\min},\theta_{\mathrm{w}}^{\max}\right]$ and thus increases the risk of idleness.

Remark 6: Multi-round scheduling is known to favor early involvement of the workers by sending small chunks for the first round. Here, paradoxically, the size of chunks for the first round is not minimized. Still worse, sending the first chunk to successive workers is delayed. These are only apparent paradoxes, since we consider applications for which the duration of the first round is negligible compared to the computation time for the whole flow.

VI. Conclusion

The AS4DR method presented in this paper succeeds in maximizing the throughput of each worker, when scheduling a divisible load of unknown total size on heterogeneous distributed resources with inaccurately specified steady characteristics. Up to now we have considered that the whole set of resources is used. Despite the fact that a bidirectionnal 1-port communication model is prone to contention, AS4DR can avoid this pitfall, thanks both to the asymptotic periodicity it installs (for both data and results) and to its preliminary step CIP. It has been proved and illustrated by simulations, that CIP can give a valid collection of initial values which reduce the risk of both link contention and worker idleness. Simulations with SimGrid have shown that there definitely exists an application domain for the proposed scheduling method.

Firstly, simulations of more realistic and larger platforms will be conducted in order to assess the scalability of AS4DR. Of course, for a given set of execution parameters values, using all the available ressources will ultimately become impossible with an evermore increasing amount of resources. So, secondly, we will study the relationship between the optimality of the throughput of the global scheduling and that of each worker individually, when all the ressources are not usable. From the conclusion of this work, the better understanding of this relationship could lead to a method to select a relevant subset of resources.

Besides, being able to provide an efficient scheduling without knowing at the initial moment the constant value of execution parameters, AS4DR should also be able to provide an efficient scheduling when this value is unknown at each moment when it is needed. For this reason AS4DR should be able to provide an efficient scheduling when the value of execution parameters evolves over time. Thus, future work will also try to check this hypothesis by means of simulations, studying the impact on the throughput of diverse ways to introduce dynamicity into the behaviour of the platform: frequency, amplitude, smoothness,... of the variations of the execution parameters value.

References

[1] C. Banino, O. Beaumont, L. Carter, J. Ferrante, A. Legrand, and Y. Robert, "Scheduling strategies for master-slave tasking on heterogeneous processor platforms," IEEE Transactions on Parallel and Distributed Systems, vol. 15, no. 4, pp. 319–330, April 2004.

[2] G. Cordasco, G. Malewiczb, and A. L. Rosenberg, "Applying ic-scheduling theory to familiar classes of computations," in Proceeding of the 21th International Parallel and Distributed Processing Symposium (IPDPS'07), vol. 1, no. 1, IEEE Computing Society Press, Mars 2007, pp. 1–18.

[3] G. Cordasco, G. Malewiczb, and A. Rosenberg, "Extending ic-scheduling via the sweep algorithm," Journal of Parallel and Distributed Computing, vol. 70, no. 3, pp. 201–211, 2010.

[4] A. Shokripour and M. Othman, "Categorizing dlt researches and its applications," European Journal of Scientific Research, vol. 37, no. 3, pp. 496–515, 2009.

[5] J. Sohn, T. Robertazzi, and S. Luryi, "Optimizing computing costs using divisible load analysis," IEEE Transactions on Parallel and Distributed Systems, vol. 9, no. 3, March 1998.

[6] V. Bharadwaj, D. Ghose, V. Mani, and T. Robertazzi, "Scheduling divisible loads in parallel and distributed systems," IEEE Computing Society Press, 1996.

[7] T. Robertazzi, "Ten reasons to use divisible load theory," IEEE Computer, vol. 36(5), no. 63-68, 2003.

[8] T.Robertazzi, "Divisible load scheduling," http://www.ece.sunysb.edu/∼tom/dlt.html.

[9] O. Beaumont, A. Legrand, and Y. Robert, "Optimal algorithms for scheduling divisible workloads on heterogeneous systems," INRIA, Le Chesnay(France), Tech. Rep. RR-4595, October 2002.

[10] Y. Yang and H. Casanova, UMR: A Multi-Round Algorithm for Scheduling Divisible Workloads, IEEE Computing Society Press, April 2003.

[11] V. Bharadwaj, D. Ghose, and V. Mani, "Multi-installment load distribution in tree networks with delays," IEEE Transactions on Aerospace and Electronic Systems, vol. 31, no. 2, pp. 555–567, 1995.

[12] A. Rosenberg and R. Chiang, "Toward understanding heterogeneity in computing," in Proceeding of the 24th International Parallel and Distributed Processing Symposium (IPDPS'10), vol. 1, no. 1, IEEE Computing Society Press, April 2010, pp. 1–10.

[13] R. W. Hockney, "The communication challenge for mpp: Intel paragon and meiko cs-2," Parallel Comput., vol. 20, no. 3, pp. 389–398, 1994.

[14] J. T. Hung and T. Robertazzi, "Scheduling nonlinear computational loads," Stony Brook University, NYC, Tech. Rep. CEAS Technical Report 823, September 2006.

[15] J. Jia, B. Veeravalli, and D. Ghose, "Adaptive load distribution strategies for divisible load processing on resource unaware multilevel tree networks," IEEE Transactions On Computers, vol. 56, no. 7, pp. 999–1005, 2007.

[16] H. Li, G. Sun, and Y. Xu, "Distributed scheduling strategies for processing multiple divisible loads with unknown network resources," in Proceedings of the International Conference on Network and Parallel Computing Workshops, IEEE Computing Society Press, 2007, pp. 824–829.

[17] D. Luong, J. Deogun, and S. Goddard, "Feedback scheduling of real-time divisible loads in clusters," SIGBED Rev., vol. 5, no. 2, pp. 1–4, 2008.

[18] M. Drozdowski, "Selected problems of scheduling tasks in multiprocessor computing systems," Ph.D. dissertation, Instytut Informatyki Politechnika Poznanska, Poznan, 1997.

[19] S. Boutammine, D. Millot, and C. Parrot, "Dynamically scheduling divisible load for grid computing," in Proceedings of the 2nd International Conference High Performance Computing and Communications (HPCC06 2006), Springer-Verlag, 2006.

[20] Y. Yang and H. Casanova, "Rumr: Robust scheduling for divisible workloads," in HPDC, 2003, pp. 114–125.

[21] D. Millot and C. Parrot, "Contention-free scheduling in a dynamic context," in Proceedings of the 2008 International Conference on Parallel and Distributed Systems (ICPADS'08), IEEE Computing Society Press, 2008.

[22] S. Boutammine, D. Millot, and C. Parrot, "A runtime scheduling method for dynamic and heterogeneous platforms," in Proceedings of the 2006 International Conference on Parallel Processing Workshops (ICPPW'06), IEEE Computing Society Press, 2006.

[23] V. Bharadwaj and G. Barlas, "Efficient scheduling strategies for processing multiple divisible loads on bus networks," Journal of Parallel and Distributed Computing, vol. 62, no. 1, pp. 132–151, January 2002.

[24] H. Casanova, A. Legrand, and M. Quinson, "SimGrid: a Generic Framework for Large-Scale Distributed Experiments," in 10th IEEE International Conference on Computer Modeling and Simulation, Mar. 2008.

Daniel Millot received his PhD degree in Computer Science in 1991, from the Paris XI University. Associate Professor at Telecom SudParis, his research interests are in High Performance Computing : parallel algorithms, load balancing, heterogeneous programming.

Christian Parrot received the Ph.D. degrees in mathematics from the Université Paris 6, France, in 1993. He is currently an associate professor in the Computer Science Department of Telecom sudParis. His main research interest is scheduling methods for distributed resources.