Journal of Scheduling manuscript No.
(will be inserted by the editor)

On Multiprocessor Temperature-Aware Scheduling Problems

Evripidis Bampis · Dimitrios Letsios · Giorgio Lucarelli · Evangelos Markakis · Ioannis Milis

the date of receipt and acceptance should be inserted later

Abstract We study temperature-aware scheduling problems under the model introduced in [Chrobak et al., AAIM 2008], where unit-length jobs of given heat contributions and common release dates are to be scheduled on a set of parallel identical processors. We consider three optimization criteria: makespan, maximum temperature and (weighted) average temperature. On the positive side, we present polynomial time approximation algorithms for the minimization of the makespan and the maximum temperature, as well as, optimal polynomial time algorithms for minimizing the average temperature and the weighted average temperature. On the negative side, we prove that there is no an approximation algorithm of absolute ratio $\frac{4}{3} - \epsilon$ for the problem of minimizing the makespan for any $\epsilon > 0$, unless $\mathcal{P} = \mathcal{NP}$.

1 Introduction

The exponential increase in the processing power of recent (micro)processors has led to an analogous increase in the energy consumption of computing systems of any kind, from compact mobile devices to large scale data centers. This has also led to vast heat emissions and high temperatures affecting the processors' performance and reliability. Moreover, high temperatures reduce the lifetime of chips and may permanently damage the processors. For this reason, manufacturers have set appropriate

A preliminary version of this work has been published in the Proceedings of the FAW-AAIM 2012.

This work was partially supported by the French Agency for Research under the DEFIS program TODO, ANR-09-EMER-010, and by the GDR-RO group of CNRS (E. Bampis, D. Letsios, G. Lucarelli), by the European Social Fund and Greek national resources under the THALES-ALGONOW project (E. Bampis, G. Lucarelli, E. Markakis, I. Milis) and by the Basic Research Funding Program of Athens University of Economics and Business (I. Milis).

Evripidis Bampis
LIP6, Université Pierre et Marie Curie
E-mail: Evripidis.Bampis@lip6.fr

Dimitrios Letsios, Giorgio Lucarelli
IBISC, Université d' Évry & LIP6, Université Pierre et Marie Curie
E-mail: {Dimitris.Letsios, Giorgio.Lucarelli}@lip6.fr

Evangelos Markakis, Ioannis Milis
Department of Informatics, Athens University of Economics and Business
E-mail: {markakis, milis}@aueb.gr

temperature thresholds for their processors and use cooling systems to control the temperature below these thresholds. However, the energy consumption and heat emission of these cooling systems have to be added to that of the whole system.

The issues of energy and thermal management in the (micro)processor and system design levels date back to the first computer systems. During the last few years these issues have been also addressed at the operating system's level, generating new interesting questions. In this context the operating system has to decide the order in which the jobs should be scheduled so that the system's temperature (and/or energy consumption) remains as low as possible, while at the same time some standard user or system oriented criterion (e.g., makespan, response time, throughput, etc) is optimized. Clearly, the minimization of the temperature and the optimization of the scheduling criteria are typically in conflict, and several models have been proposed in the literature in order to analyze such conflicts and trade-offs. A first model is based on the speed-scaling technique for energy saving and the Newton's law of cooling; see for example [5,4] as well as recent reviews on speed scaling in [15,1,2]. In another model proposed in [18], a thermal RC circuit is utilized to capture the temperature profile of a processor.

In this work, we adopt the simplified model for cooling and thermal management introduced by Chrobak et al. [9], who were motivated by [17]. In particular, they consider a set of unit-length jobs (corresponding to slices of the processes to be scheduled), each one with a given heat contribution, and model the thermal behavior of the system as follows: if a job of heat contribution h is executed on a processor within a time interval $[t-1,t)$, $t \in \mathbb{N}$, and the temperature of the processor at time $t-1$ is Θ, then the processor's temperature at time t is $\frac{\Theta+h}{2}$. Although in practice the heat contribution of the executed jobs and the cooling effect are spread over time [19], the authors in [9] consider the above simplified discrete model in which the heat contribution of the job to be executed is first added to the current temperature and then this sum is halved, in order to take into account the cooling effect.

In [9], the authors study the problem of scheduling a set of unit-length jobs with release dates and deadlines on a single processor so as to maximize the throughput, i.e., the number of jobs that meet their deadlines, without exceeding a given temperature threshold θ at any time $t \in \mathbb{N}$. Extending the well-known three-field notation for scheduling problems [13], this problem is denoted as $1|r_i, p_i = 1, h_i, \theta| \sum U_i$. They prove that this problem is NP-hard even for the special case when all jobs are released at time 0 and their deadlines are equal, i.e., $1|p_i = 1, h_i, \theta| \sum U_i$. Furthermore, in the presence of release dates and deadlines it is shown that a family of reasonable list scheduling algorithms, including the *coolest first* and *earliest deadline first* algorithms, have a competitive ratio of at most two. This result implies also an approximation factor of two for the off-line problem. In the negative side, they also give an instance that shows that there is no deterministic on-line algorithm with competitive ratio less than two.

The same model has been also adopted by Birks et al. in [7,6,8] where online algorithms for several generalizations of the throughput maximization problem have been studied. In fact, in [7] the cooling effect is generalized by multiplying the temperature by $1/c$, where $c > 1$, instead of one half. In [6] the weighted throughput objective is considered, while in [8] the jobs have equal (non-unit) processing times.

Our problems and results. Under the thermal model of Chrobak et al. [9], we initiate the study of scheduling a set $J = \{J_1, J_2, \dots, J_n\}$ of n jobs on a system of m identical processors, unlike the previous works that study only single processor systems. All jobs have common release dates and unit processing times, and for each one of them we are given a heat contribution h_i, $1 \le i \le n$. Let $h_{\max} = \max\{h_i, J_i \in J\}$ be the maximum heat contribution among all jobs. We consider each job J_i executed in a time interval $[t-1,t)$, $t \in \mathbb{N}$, which we call slot t, on some processor. By Θ_t^j we denote the temperature of processor j at time t. As in [9], if we start executing job J_i at time $t-1$, then $\Theta_t^j = \frac{\Theta_{t-1}^j + h_i}{2}$. The initial temperature of each processor (the ambient temperature) is considered to be zero, i.e., $\Theta_0^j = 0$. In what follows, we simplify the notation by using Θ_t instead of Θ_t^j, when the processor is specified by the context. We consider two natural variants of the above model:

The threshold thermal model. In this model, a given threshold θ on the temperature of the processors cannot be violated at any time $t \in \mathbb{N}$. This is the case with the throughput maximization problems studied in [9,7,6,8]. It is clear that, for a given instance in this model, a feasible schedule may exist only if $h_i \leq 2 \cdot \theta$ for each job J_i. By normalizing the values of h_i's and θ we can assume w.l.o.g. that $0 < h_i \leq 2$ and $\theta = 1$, as in [9]. Moreover, if a processor at time $t-1$ has temperature Θ_{t-1} and it holds that $\frac{\Theta_{t-1}+h_i}{2} > 1$, for every job J_i that has not yet been scheduled, then this processor will remain idle for the slot $[t-1, t)$ and its temperature at time t will be reduced by half, i.e., $\frac{\Theta_{t-1}}{2}$. Note also that once a processor has executed some job(s), its temperature will never become exactly zero. Therefore, in this model, a feasible instance cannot contain more than m jobs of heat contributions equal to 2, as there are m slots with $\Theta_0 = 0$ (the first slots in each one of the m available processors). Under this model we study the makespan minimization problem, that is $P|p_i = 1, h_i, \theta|C_{max}$.

The optimization thermal model. In this model, no explicit threshold on the processors' temperature is given. The lack of such a threshold is counterbalanced by studying the problems of minimizing the maximum and average temperature of a schedule. For any instance in this model, any schedule of length at least $\lceil \frac{n}{m} \rceil$ is feasible, independently of the range of the jobs' heat contributions. However, the optimum value of our objectives depends on the time available to execute the given set of jobs: the maximum or average temperature of a schedule of length equal to $\lceil \frac{n}{m} \rceil$ is, clearly, greater than that of a schedule of longer length, where we are allowed to introduce idle slots. In what follows, we are interested in minimizing these two objective functions with respect to a given schedule length (makespan or deadline) of $d \geq \lceil \frac{n}{m} \rceil$. Such a schedule will contain $md - n$ idle slots and we can consider them as executing $md-n$ fictitious jobs of heat contribution equal to zero. This length d is part of our problems' instances, denotes the time available to complete the execution of all the jobs and represents the need to complete them within a given time at the price of higher temperatures. Thus, in both problems we consider under this model (minimizing the maximum and the average temperature) we are accounting the temperatures at the end of any of the md slots available on the m processors. The problems of minimizing the maximum and average temperature we consider under this model are denoted by $P|p_i = 1, h_i, d|\Theta_{max}$ (where $\Theta_{max} = \max\{\Theta_t^j,\ 1 \leq t \leq d,\ 1 \leq j \leq m\}$) and $P|p_i = 1, h_i, d|\sum \Theta_t^j$ (where $1 \leq t \leq d,\ 1 \leq j \leq m$), respectively.

The complexity of our problems is strongly related to the complexity of the throughput maximization problem studied in [9]. It is already mentioned in [9], that the NP-hardness of the maximum throughput problem of scheduling jobs with common release dates and deadlines on a single processor $1|p_i = 1, h_i, \theta|\sum U_i$ implies the NP-hardness of our makespan minimization problem $1|p_i = 1, h_i, \theta|C_{max}$. In fact, the decision version of the latter problem asks for the existence of a feasible schedule where all jobs complete their execution by some given deadline d. Moreover, the decision version of the maximum temperature problem on a single processor $1|p_i = 1, h_i, d|\Theta_{max}$ asks for the existence of a schedule where all jobs complete their execution by some given deadline d without exceeding a given temperature threshold θ. Therefore, the same reduction gives NP-hardness for both makespan and maximum temperature minimization problems. The NP-hardness for our problems on an arbitrary number of parallel processors follows trivially.

Given these NP-hardness results, in this paper we focus on approximation algorithms and inapproximability results for the above mentioned problems, under the threshold and optimization thermal models for the case of multiple processors. We start in Section 2 with the problem $P|p_i = 1, h_i, \theta|C_{max}$ of minimizing the schedule length (makespan) in the threshold thermal model. We first prove that this problem cannot be approximated within an absolute ratio less than 4/3. Then we present a generic algorithm of approximation ratio 2ρ, where ρ is the approximation ratio of an algorithm $\mathcal{A}$ for the classical makespan problem on parallel machines, used as a subroutine in our algorithm. This leads to a $(2+\epsilon)$-approximation ratio within a running time that is polynomial in n but exponential in $1/\epsilon$ for m processors (by using the known PTAS's for minimizing makespan), and a 2-approximation ratio for a single processor, within $O(n \log n)$ time. If in the place of algorithm $\mathcal{A}$ we use the standard LPT $(\frac{4}{3} - \frac{1}{3m})$-approximation algorithm, we are able to give a tighter analysis, improving the 2ρ-approximation ratio to $\frac{7}{3} - \frac{1}{3m}$, while the overall running time is $O(n \log n)$. Then in Sections 3 and 4, we move to the optimization thermal model. In Section 3, we study the problem $P|p_i = 1, h_i, d|\Theta_{max}$ of minimizing the maximum temperature of a schedule, and we give a 4/3 approximation algorithm. In Section 4, we prove that the problem $P|p_i = 1, h_i, d| \sum \Theta_t^j$ of minimizing the average temperature of a schedule, as well as a time-dependent weighted version of this problem are both solvable in polynomial time. We conclude in Section 5.

2 Makespan Minimization

In this section we study the approximability of makespan minimization under the threshold thermal model, that is, $P|p_i = 1, h_i, \theta|C_{\max}$.

We start with a negative result on the approximability of our problem. The proof of the next theorem is along the same lines with the NP-hardness reduction for the throughput maximization problem under the same model [9].

Theorem 1 *There is no polynomial time algorithm achieving an absolute approximation ratio better than 4/3 for the minimum makespan problem $P|p_i = 1, h_i, \theta|C_{\max}$, unless $\mathcal{P} = \mathcal{NP}$.*

Proof We give a reduction from Numerical 3-Dimensional Matching (N3DM) where we are given three sets A, B, C of n integers each and an integer β, and the question is whether $A \cup B \cup C$ can be partitioned into n disjoint triples $(a, b, c) \in A \times B \times C$ such that each triple contains exactly one integer from each of A, B, C, and $a + b + c = \beta$ for each triple. W.l.o.g., we assume that $\sum_{x \in A \cup B \cup C} x = \beta n$ and $x \leq \beta$ for each $x \in A \cup B \cup C$. The N3DM problem is known to be NP-complete (see [11]).

Given an instance I of N3DM, we construct an instance I' of $P|p_i = 1, h_i,\ \theta|C_{\max}$ consisting of n processors and $3n$ jobs, one for each integer in $A \cup B \cup C$. Considering the function $f(x) = \frac{1}{25}\left(1 + \frac{x}{8\beta}\right)$, we set $h(a) = 8f(a) + 1$ for each $a \in A$, $h(b) = 4f(b) + 1$ for each $b \in B$ and $h(c) = 2f(c) + 1$ for each $c \in C$.

The reduction works by showing that it is hard to decide whether the optimal schedule is of length three or not.

Claim There is a N3DM for instance I if and only if there is a feasible schedule for the instance I' of $P|p_i = 1, h_i, \theta|C_{\max}$ of length three.

Proof ($\Rightarrow$) Assume that there is a solution for N3DM. For the i-th triple (a_i, b_i, c_i), $1 \leq i \leq n$, in this solution, we schedule in the i-th processor the jobs corresponding to a_i, b_i and c_i in the first, second and third slots, respectively. For the temperatures, $\Theta_{a_i}, \Theta_{b_i}, \Theta_{c_i}$, of the i-th processor after each one of those executions we have

$$\Theta_{a_i} = \tfrac{8f(a_i)+1}{2} \leq \tfrac{8f(\beta)+1}{2} = \tfrac{\frac{8}{25}\left(1+\frac{\beta}{8\beta}\right)+1}{2} = \tfrac{34}{50} \leq 1$$

$$\Theta_{b_i} = \tfrac{8f(a_i)+1}{4} + \tfrac{4f(b_i)+1}{2} = \tfrac{3}{4} + 2\left(\tfrac{1}{25}\left(1 + \tfrac{a_i}{8\beta}\right) + \tfrac{1}{25}\left(1 + \tfrac{b_i}{8\beta}\right)\right) \leq \tfrac{3}{4} + \tfrac{4}{25} + \tfrac{\beta}{100\beta} = \tfrac{92}{100} \leq 1$$

$$\Theta_{c_i} = \tfrac{8f(a_i)+1}{8} + \tfrac{4f(b_i)+1}{4} + \tfrac{2f(c_i)+1}{2} = \tfrac{7}{8} + \tfrac{1}{25}\left(1 + \tfrac{a_i}{8\beta}\right) + \tfrac{1}{25}\left(1 + \tfrac{b_i}{8\beta}\right) + \tfrac{1}{25}\left(1 + \tfrac{c_i}{8\beta}\right) = \tfrac{7}{8} + \tfrac{3}{25} + \tfrac{\beta}{200\beta} = 1$$

and hence there is a feasible schedule of length three.

($\Leftarrow$) Assume, now, that there is a feasible schedule of length three. In this schedule there are exactly three jobs in each processor, since there are $3n$ jobs in total.

If a job corresponding to an integer $a \in A$ is scheduled to the second slot of a processor, then the temperature threshold $\theta = 1$ is violated after the third slot of this processor. Indeed the temperature at this slot will be at least

$$\tfrac{2f(0)+1}{8} + \tfrac{8f(0)+1}{4} + \tfrac{2f(0)+1}{2} = \tfrac{7}{8} + \tfrac{1}{25}\left(1 + \tfrac{0}{8\beta}\right)\left(\tfrac{2}{8} + \tfrac{8}{4} + \tfrac{2}{2}\right) = \tfrac{201}{200} > 1.$$

In a similar way, we can show that a job corresponding to an integer $a \in A$ cannot be scheduled to the third slot of a processor:

$$\tfrac{2f(0)+1}{8} + \tfrac{2f(0)+1}{4} + \tfrac{8f(0)+1}{2} = \tfrac{7}{8} + \tfrac{1}{25}\left(1 + \tfrac{0}{8\beta}\right)\left(\tfrac{2}{8} + \tfrac{2}{4} + \tfrac{8}{2}\right) = \tfrac{213}{200} > 1.$$

Hence, each of the n jobs corresponding to one of the n integers $a \in A$ is scheduled to the first slot of a processor. Moreover, we can show that a job corresponding to an integer $b \in B$ cannot be scheduled to the third slot of a processor:

$$\tfrac{8f(0)+1}{8} + \tfrac{2f(0)+1}{4} + \tfrac{4f(0)+1}{2} = \tfrac{7}{8} + \tfrac{1}{25}\left(1 + \tfrac{0}{8\beta}\right)\left(\tfrac{8}{8} + \tfrac{2}{4} + \tfrac{4}{2}\right) = \tfrac{203}{200} > 1.$$

In all, in each processor exactly three jobs are scheduled: a job $a \in A$ in the first slot, a job $b \in B$ in the second slot, and a job $c \in C$ in the third slot. Therefore, the jobs of a processor correspond to a feasible triple for N3DM.

To finish our proof, we have to show that each triple sums up to β. If this does not hold then there is a triple (a, b, c) for which $a + b + c > \beta$, since $\sum_{x \in A \cup B \cup C} x = \beta n$. The temperature of the third slot of the processor in which the corresponding jobs to this triple are scheduled is

$$\tfrac{8f(a)+1}{8} + \tfrac{4f(b)+1}{4} + \tfrac{2f(c)+1}{2} = \tfrac{7}{8} + \tfrac{1}{25}\left(3 + \tfrac{a+b+c}{8\beta}\right) > \tfrac{7}{8} + \tfrac{1}{25}\left(3 + \tfrac{\beta}{8\beta}\right) = 1,$$

which is a contradiction that there is a feasible schedule. □

This completes the proof of Theorem 1 since an approximation ratio better than 4/3 would be able to decide the N3DM problem. □

Note that the result of Theorem 1 allows the possibility of an asymptotic PTAS or even an additive constant approximation ratio.

In what follows in this section, we present an approximation algorithm for the minimum makespan problem. Note that, in order to respect the temperature threshold, a schedule may have to contain idle slots. To argue about the number of idle slots that are needed before the execution of each job, we will introduce first an appropriate partition of the set of jobs according to their heat contribution. In particular, for each integer $k \geq 0$, we can argue separately for jobs whose heat contribution belongs to the interval $(2 - \frac{1}{2^{k-1}}, 2 - \frac{1}{2^k}]$; recall that $h_i \leq 2$, for $1 \leq i \leq n$. Moreover, the interval to which a job of heat contribution h_i belongs to is indexed by k_i, that is

$$k_i = \max\{k \in \mathbb{N} \mid h_i > 2 - \tfrac{1}{2^{k-1}}\}$$

Our algorithm and its analysis are based on the following proposition for the structure of any feasible schedule.

Proposition 1
(i) Let J' be the set of jobs of heat contribution $h_i > 1$; $|J'| = n'$. Any feasible schedule can be transformed into another feasible one of at most the same length where exactly $\min\{n', m\}$ *jobs in J' are executed in the*

first slot of the processors.
(ii) Any schedule where every J_i is executed right after k_i consecutive idle slots is feasible.
(iii) In an optimal schedule, if a job J_j is executed before a job J_i on the same processor, where $h_j, h_i > 1$, then there are at least $k_i - 1$ slots between J_j and J_i, which are either idle or execute jobs of heat contribution at most one.

Proof
(i) Consider a feasible schedule that has less than $\min\{n', m\}$ jobs in J' executed in the first slot of the processors.

Assume, first, that in this schedule there is a processor, p, in which a job $J_i \in J \setminus J'$ is executed in its first slot and there is at least one job of J' executed in p. Let $J_j \in J'$ be the earliest of these jobs which is executed in slot $s > 1$. By swapping the jobs J_i and J_j, the temperature Θ'_s of processor p after slot s is decreased. Indeed, let Θ_s be the temperature of processor p after slot s and Θ' be the contribution of jobs executed in slots $2, 3, \ldots, s-1$ to Θ_s, that is $\Theta_s = \frac{h_i}{2^s} + \Theta' + \frac{h_j}{2}$. After the swap it holds that $\Theta'_s = \frac{h_j}{2^s} + \Theta' + \frac{h_i}{2} < \Theta_s$, since $h_i < h_j$. Thus, the temperature of any slot $s' \geq s$ in p is decreased. Moreover, by assumption, each slot s', $2 \leq s' \leq s-1$, of p executes a job in $J \setminus J'$. Hence, no new idle slots are required for these jobs, although the temperature before their execution is increased. Therefore, the new schedule is feasible and it has the same length.

If there is not such a processor, then let $J_i \in J \setminus J'$ be a job executed in the first slot of some processor p and $J_j \in J'$ be a job executed in s-th, $s > 1$, slot of processor q. By swapping the jobs J_i and J_j the temperature of any slot $s' \geq s$ of processor q is decreased as $h_i < h_j$. Moreover, by assumption, the processor p contains only jobs in $J \setminus J'$, and, as in the previous case, no new idle slots are required for these jobs. Therefore, after the swap we get a feasible schedule of the same length.

(ii) Consider a schedule that is feasible up until the execution of the job preceding J_i. Let x be the number of idle slots before the execution of job J_i and let Θ be the temperature of the processor before the first of these x slots. Since the schedule is feasible before J_i, we have that $\Theta \leq 1$. The temperature will become $\frac{\Theta}{2^x}$, after the last idle slot, and $\frac{\frac{\Theta}{2^x}+h_i}{2}$ after the execution of job J_i. For such a schedule to be feasible we need that $\frac{\frac{\Theta}{2^x}+h_i}{2} \leq 1$, that is, $2^x \geq \frac{\Theta}{2-h_i}$. Since $h_i \leq \frac{2^{k_i+1}-1}{2^{k_i}}$, it follows that $\frac{\Theta}{2-h_i} \leq \frac{1}{2-\frac{2^{k_i+1}-1}{2^{k_i}}} = 2^{k_i}$. This means that with at least k_i idle slots, feasibility is ensured.

(iii) Let Θ_t be the temperature of the processor before executing J_j. Next, after the execution of J_j we have $\Theta_{t+1} = \frac{\Theta_t + h_j}{2}$. Then, after x slots (idles or executing jobs of heat contribution $h \leq 1$) we get a temperature $\Theta_{t+x+1} \geq \frac{\Theta_t + h_j}{2} \cdot \frac{1}{2^x}$. In order for J_i to be executed in the next slot, it should hold that $\Theta_{t+x+1} + h_i \leq 2$, that is $2^x \geq \frac{\Theta_t + h_j}{2(2-h_i)}$. Since, $\Theta_t \geq 0$, $h_j > 1$ and $h_i > \frac{2^{k_i}-1}{2^{k_i-1}}$ we get $2^x \geq \frac{\Theta_t + h_j}{2(2-h_i)} > \frac{1}{2(2-\frac{2^{k_i}-1}{2^{k_i-1}})} = \frac{1}{\frac{2}{2^{k_i-1}}} = 2^{k_i-2}$, that is $x \geq k_i - 1$. □

In what follows we consider instances with $n > m$, for otherwise the problem becomes trivial. By Proposition 1(i), we also assume that the number of jobs of heat contribution $h_i > 1$ is greater than m. If this is not the case, all jobs can be executed without any idle slot before them and the length of an optimal schedule is exactly $\lceil \frac{n}{m} \rceil$. We consider the jobs in non-increasing order of their heat contributions, i.e., $h_1 \geq h_2 \geq \ldots \geq h_n$, and we define $A = \{J_1, J_2, \ldots, J_m\}$ and $B = \{J_{m+1}, J_{m+2}, \ldots, J_n\}$. Our algorithm schedules first the jobs in A to the first slot of each processor. Each one of the jobs in B is scheduled by leaving before its execution *exactly* k_i idle slots, according to the Proposition 1(ii). In this way, our problem, for the jobs in B, is transformed to an instance of the classical makespan problem on parallel machines, $P||C_{\max}$, where the processing time of each job is $p_i = k_i + 1$, that is, k_i idle slots plus its original unit processing time. Then, these jobs are scheduled using any known approximation algorithm $\mathcal{A}$ for $P||C_{\max}$.

¿From now on we fix an instance of our problem and we denote by SOL the length of the schedule $\mathcal{S}$ provided by Algorithm MAX_C and by OPT the length of an optimal schedule $\mathcal{S}^*$ for our original scheduling problem.

Algorithm MAX_C

```
1: Sort the jobs in non-increasing order of their heat contributions: $h_1 \geq h_2 \geq ... \geq h_n$;
2: Let $A = \{J_1, J_2, \dots, J_m\}$, and $B = \{J_{m+1}, J_{m+2}, \dots, J_n\}$;
3: Schedule each job $J_i \in A$ to the first slot of processor $i$;
4: Run an algorithm $\mathcal{A}$ for $P||C_{\max}$ on instance $\mathcal{I}_B^+$;
```

For the presentation and the analysis of our algorithm, we denote by $\mathcal{I}_B$ and $\mathcal{I}_B^+$ the instances of $P||C_{\max}$ consisting only of jobs in B with processing times $p_i = k_i$ and $p_i = k_i + 1$, respectively, for each $J_i \in B$. For an instance $\mathcal{I}$ of $P||C_{\max}$, we denote by $\mathcal{S}(\mathcal{I})$ the schedule found by an algorithm $\mathcal{A}$ and by $\mathcal{C}(\mathcal{I})$ the length of this schedule. In a similar way, we denote by $\mathcal{S}^*(\mathcal{I})$ and $\mathcal{C}^*(\mathcal{I})$ an optimal schedule for $P||C_{\max}$ and the length of this optimal schedule, respectively.

Clearly, $SOL = 1 + \mathcal{C}(\mathcal{I}_B^+)$. To analyze our Algorithm MAX_C, we need a lower bound on the optimal makespan. To derive this bound we will utilize an optimal schedule $\mathcal{S}^*(\mathcal{I}_B)$. Note that for jobs with $h_i \in (0, 1]$, $k_i = 0$, hence the schedule $\mathcal{S}^*(\mathcal{I}_B)$ involves only jobs for which $h_i > 1$.

Lemma 1 *For the optimal makespan it holds that*

$$OPT \geq \max\{\frac{n}{m}, 1 + \mathcal{C}^*(\mathcal{I}_B)\}$$

Proof The first bound on the optimal makespan follows trivially by considering all jobs requiring a single slot for their execution.

For the second bound, let A^*, $|A^*| = m$, be the set of jobs executed in the first slot of the m processors in an optimal solution and $B^* = J \setminus A^*$.

Consider, first, an auxiliary schedule of length OPT^-, identical to the optimal apart from the fact that each job in $B^* \cap A$ has been replaced by a different job in $A^* \cap B$. Observe that in this schedule, the jobs executed in the first slot of the processors remain A^* while the jobs executed in the remaining slots are the jobs in B. Since each job in B has smaller or equal heat contribution than any job in A, it follows that $OPT \geq OPT^-$.

Consider, next, the schedule $\mathcal{S}^*(\mathcal{I}_B)$. For this schedule it holds that, $OPT^- \geq 1 + \mathcal{C}^*(\mathcal{I}_B)$, since by Proposition 1(i),(iii) each job in B requires at least k_i slots to be executed; recall that we consider instances where the number of jobs of heat contribution $h_i > 1$ is greater than m and that jobs in B with $h_i \leq 1$, and hence $k_i = 0$, do not appear in the schedule $\mathcal{S}^*(\mathcal{I}_B)$. □

It is well-known that the $P||C_{\max}$ problem is strongly NP-hard and a series of constant approximation algorithms and PTASs have been proposed. Our main result in this section is that in step 4 of Algorithm MAX_C we can use any algorithm $\mathcal{A}$ for $P||C_{\max}$ to obtain twice the approximation ratio of $\mathcal{A}$ for our problem.

Theorem 2 *Algorithm* MAX_C *achieves a 2ρ approximation ratio for $P|p_i = 1, h_i, \theta|C_{\max}$, where ρ is the approximation ratio of the algorithm $\mathcal{A}$ for $P||C_{\max}$.*

Proof A ρ-approximation algorithm $\mathcal{A}$ implies that $\frac{\mathcal{C}(\mathcal{I}_B^+)}{\mathcal{C}^*(\mathcal{I}_B^+)} \leq \rho$. Hence, $SOL = 1 + \mathcal{C}(\mathcal{I}_B^+) \leq 1 + \rho \cdot \mathcal{C}^*(\mathcal{I}_B^+)$.

To obtain an upper bound to $\mathcal{C}^*(\mathcal{I}_B^+)$ we start from the schedule $\mathcal{S}^*(\mathcal{I}_B)$. The processing times of jobs in the latter schedule are reduced by one with respect to the former one, and the jobs in B with $h \leq 1$ do not appear in schedule $\mathcal{S}^*(\mathcal{I}_B)$. Let $B' \subseteq B$ be this set of jobs.

We transform the schedule $\mathcal{S}^*(\mathcal{I}_B)$ to a new schedule $\mathcal{S}'(\mathcal{I}_B^+)$ in two successive steps: (i) we increase the processing time of jobs in $B \setminus B'$ from k_i to $k_i + 1$, and (ii) we introduce the jobs in B' with unit processing time, at the end of the resulting schedule in a first-fit manner. Clearly, for the length, $\mathcal{C}'(\mathcal{I}_B^+)$, of this new schedule it holds that $\mathcal{C}^*(\mathcal{I}_B^+) \leq \mathcal{C}'(\mathcal{I}_B^+)$ as both of them refer to the same instance $\mathcal{I}_B^+$. Let us now bound $\mathcal{C}'(\mathcal{I}_B^+)$ in terms of $\mathcal{C}^*(\mathcal{I}_B)$.

If $\mathcal{C}'(\mathcal{I}_B^+) \leq 2\mathcal{C}^*(\mathcal{I}_B)$, then $\frac{SOL}{OPT} \leq \frac{1+2\rho\mathcal{C}^*(\mathcal{I}_B)}{1+\mathcal{C}^*(\mathcal{I}_B)} \leq 2\rho$, since $\rho \geq 1$.

If $\mathcal{C}'(\mathcal{I}_B^+) > 2\mathcal{C}^*(\mathcal{I}_B)$, then we consider the construction of $\mathcal{S}'(\mathcal{I}_B^+)$ and we argue about the completion time of a critical processor in $\mathcal{S}^*(\mathcal{I}_B)$, i.e., the processor that finishes last. By step (i), the length of schedule $\mathcal{S}^*(\mathcal{I}_B)$ increases at most twice, since each job in $B \setminus B'$ has processing time at least one and this is increased by 1. As $\mathcal{C}'(\mathcal{I}_B^+) > 2\mathcal{C}^*(\mathcal{I}_B)$, in the last slot of $\mathcal{S}'(\mathcal{I}_B^+)$ all non-idle processors execute jobs of B'. By step (ii), all but the last time slots of $\mathcal{S}'(\mathcal{I}_B^+)$ are busy. Hence, the critical processor in $\mathcal{S}^*(\mathcal{I}_B)$ finishes in $\mathcal{S}'(\mathcal{I}_B^+)$ the earliest at time $\mathcal{C}'(\mathcal{I}_B^+) - 1$. Moreover, this processor is assigned the minimum total increase at the end of the transformation, since it finishes last in $\mathcal{S}^*(\mathcal{I}_B)$. As the total increase of the processing times from $\mathcal{S}^*(\mathcal{I}_B)$ to $\mathcal{S}'(\mathcal{I}_B^+)$ is $n-m$, it follows that the length of the critical processor increases at most by $\frac{n-m}{m}$. Hence, $\mathcal{C}'(\mathcal{I}_B^+) - 1 \leq \mathcal{C}^*(\mathcal{I}_B) + \frac{n-m}{m}$, that is $\mathcal{C}'(\mathcal{I}_B^+) \leq \mathcal{C}^*(\mathcal{I}_B) + \frac{n}{m}$. Thus, by Lemma 1 we get $\frac{SOL}{OPT} \leq \frac{1+\rho(\mathcal{C}^*(\mathcal{I}_B)+\frac{n}{m})}{\max\{\frac{n}{m},1+\mathcal{C}^*(\mathcal{I}_B)\}} \leq \frac{1+\rho\mathcal{C}^*(\mathcal{I}_B)}{1+\mathcal{C}^*(\mathcal{I}_B)} + \frac{\rho\frac{n}{m}}{\frac{n}{m}} \leq 2\rho$. □

For the case of a single processor the $1||C_{max}$ problem is trivially polynomial, whereas for multiple processors there are well known PTAS's, e.g., [14,3]. Hence the main implication of Theorem 2 is:

Corollary 1 *For any $\epsilon > 0$, there is a $(2+\epsilon)$-approximation algorithm for $P|p_i = 1, h_i, \theta|C_{\max}$. For the case of a single processor, there is an algorithm that achieves an approximation ratio of* 2.

To obtain the ratio of $2+\epsilon$, as stated above, one needs to use a PTAS for the classical makespan problem in step 4 of Algorithm MAX_C, resulting in a running time that is exponential in $1/\epsilon$. To achieve more practical running times, we can investigate the use of other algorithms for step 4. In particular, if the standard Longest Processing Time (LPT) algorithm is used, then Theorem 2 leads to a $2(\frac{4}{3} - \frac{1}{3m})$ approximation ratio within $O(n \log n)$ time. Recall that the LPT algorithm greedily assigns the next job (in non-increasing order of their processing times) to the first available processor [12]. In the next theorem we are able to improve this ratio to 7/3, based on an LPT oriented analysis of Algorithm MAX_C.

Theorem 3 *Algorithm* MAX_C *using the LPT rule in step 4 achieves an approximation ratio of $\frac{7}{3} - \frac{1}{3m}$ for $P|p_i = 1, h_i, \theta|C_{\max}$ within $O(n \log n)$ time.*

Proof Our proof follows the standard analysis given in [12], for the classical multiprocessor scheduling problem. For the lower bound on the length of an optimal schedule we use Lemma 1 and the fact that $\mathcal{C}^*(\mathcal{I}_B) \geq \frac{\sum_{i=m+1}^{n} k_i}{m}$. Hence, $OPT \geq \max\{\frac{n}{m}, 1 + \frac{\sum_{i=m+1}^{n} k_i}{m}\}$, and by the standard average argument we get

$$OPT \geq \frac{m+\sum_{i=m+1}^{n} k_i + n}{2m} = 1 + \frac{\sum_{i=m+1}^{n}(k_i+1)}{2m}.$$

To upper bound the length SOL of the schedule S returned by Algorithm MAX_C we consider the job J_ℓ which finishes last in S. Clearly $\ell > m$, for otherwise there are at most m jobs to be scheduled and the problem becomes trivial.

The job J_ℓ will start being executed not later than $1 + \frac{\sum_{i=m+1, j\neq\ell}^{n}(k_i+1)}{m}$, and hence, it holds that

$$SOL \le 1 + \frac{\sum_{i=m+1, j\neq \ell}^{n}(k_i+1)}{m} + (k_\ell+1) = 1 + \frac{\sum_{i=m+1}^{n}(k_i+1)}{m} + \left(1-\frac{1}{m}\right)(k_\ell+1).$$

Thus, we get $SOL \le 2OPT - 1 + \left(1-\frac{1}{m}\right)(k_\ell+1)$.

If $k_\ell \le OPT/3$, then the theorem follows directly.

If $k_\ell > OPT/3$, then we consider the subinstance, I', of the original problem that contains only the jobs of heat contribution at least h_ℓ, i.e., $J' = \{J_1, J_2, \dots, J_\ell\}$. Obviously, $k_1 \ge k_2 \ge \dots \ge k_\ell > \frac{OPT}{3}$ and $k_\ell \ge 1$, as k_ℓ is an integer. Moreover, for the length of an optimal schedule, $C^*(I')$, of the subinstance I' it holds that $C^*(I') \le OPT$. As $\ell > m$, the lengths of the schedules returned by Algorithm MAX_C for instances I and I' are equal, i.e., $C(I') = SOL$. Hence, $\frac{SOL}{OPT} \le \frac{C(I')}{C^*(I')}$.

In an optimal schedule of I' there are at most three jobs in each processor, for otherwise, if there is a processor with four assigned jobs, the length of that schedule will be, by Proposition 1(iii), at least $1 + 3k_\ell > OPT$, a contradiction. Hence, $\ell \le 3m$.

Algorithm MAX_C schedules the jobs of I' as follows: the job J_i, $1 \le i \le m$, is scheduled to the first slot of processor i, the job J_{m+i}, $1 \le i \le m$, to the $(1 + (k_{m+i} + 1))$-th slot of processor i and job J_{2m+i}, $1 \le i \le m$, accordingly to the LPT rule.

If $m < \ell \le 2m$, then the length of the above schedule is $C(I') = 1 + (k_{m+1} + 1) = 2 + k_{m+1}$. By Lemma 1 it follows that $C^*(I') \ge 1 + k_{m+1}$, since there is a processor executing at least two jobs in $\{J_1, J_2, \dots, J_{m+1}\}$. Hence, $\frac{SOL}{OPT} \le \frac{C(I')}{C^*(I')} \le \frac{2+k_{m+1}}{1+k_{m+1}} \le \frac{3}{2}$, as $k_{m+1} \ge k_\ell \ge 1$.

If $2m < \ell \le 3m$, then the Algorithm MAX_C schedules in the first processor either the jobs J_1 and J_{m+1} or the jobs J_1, J_{m+1} and J_ℓ. In the first case, the job J_ℓ starts its execution not later than the slot $1 + (k_{m+1} + 1)$, for otherwise J_ℓ would have been scheduled by Algorithm MAX_C in processor 1, that is $C(I') \le 1 + (k_{m+1} + 1) + (k_\ell + 1)$. In the second case, J_ℓ is the job that finishes last, that is $C(I') = 1 + (k_{m+1} + 1) + (k_\ell + 1)$. Thus, in both cases it holds that $C(I') \le 3 + k_{m+1} + k_\ell$.

For an optimal schedule for I', Lemma 1 implies as before that $C^*(I') \ge 1 + k_{m+1}$. Moreover, in such a schedule there is a processor with at least three jobs, and hence $C^*(I') \ge 1 + 2k_\ell$. Combining these two bounds we get $C^*(I') \ge 1 + \frac{k_{m+1}}{2} + k_\ell$.

Therefore, we get $\frac{SOL}{OPT} \le \frac{C(I')}{C^*(I')} \le \frac{6+2k_{m+1}+2k_\ell}{2+k_{m+1}+2k_\ell}$. This ratio is decreasing with k_ℓ and as $k_\ell \ge 1$ we get $\frac{SOL}{OPT} \le \frac{8+2k_{m+1}}{4+k_{m+1}} = 2$, and the proof is completed. □

Note that the $\left(\frac{4}{3} - \frac{1}{3m}\right)$-approximation ratio of the LPT algorithm for the classical makespan problem on parallel machines is tight. Concerning the tightness of our algorithm, we are able to give an instance where it achieves a 2-approximation ratio. This instance consists of $m(k+2)$ jobs: a set $\mathcal{J}_1$ of m jobs of heat contribution $h_i = 2$, a set $\mathcal{J}_2$ of m jobs of heat contribution $h_i = 2 - \frac{3}{2^{k+1}}$, and a set $\mathcal{J}_3$ of mk jobs of heat contribution $h_i = \frac{1}{2(2^k-1)}$.

An optimal solution for this instance is to schedule the jobs in the following way: every processor executes a job of $\mathcal{J}_1$ in the first slot, k jobs of $\mathcal{J}_3$ in slots $2, 3, \dots, k+1$, and a job of $\mathcal{J}_2$ in slot $k+2$. The temperature of every processor after slot $k+1$ is $\frac{1}{2^k} + \frac{1}{2(2^k-1)} \cdot \frac{2^k-1}{2^k} = \frac{3}{2^{k+1}}$, and hence a job of $\mathcal{J}_2$ can be executed in slot $k+2$. Moreover, as the jobs of $\mathcal{J}_3$ have heat contribution $h_i \le 1$, this schedule is feasible. On the other hand, our algorithm schedules in every processor a job of $\mathcal{J}_1$ in the first slot, a job of $\mathcal{J}_2$ in the slot $k+2$, and k jobs of $\mathcal{J}_3$ in slots $k+3, k+4, \cdots, 2k+2$. Therefore, the ratio achieved by our algorithm is $\frac{2k+2}{k+2} \simeq 2$.

3 Maximum Temperature Minimization

Now, we turn our attention to the optimization thermal model and to the problem of minimizing the maximum temperature, i.e., $P|p_i = 1, h_i, d|\Theta_{max}$. Recall that as we discussed in the Introduction, we consider a schedule length $d \geq \lceil \frac{n}{m} \rceil$ and that $n = m \cdot d$, by adding the appropriate number of fictitious jobs. Recall also that the maximum is taken over the temperatures at the end of any of the md slots available on the m processors. In the sequel, we will denote by Θ^*_{max} the maximum temperature of an optimal schedule.

We start with the observation that any algorithm for this problem achieves a 2 approximation ratio. Indeed, it holds that $\Theta^*_{\max} \geq h_{max}/2$, no matter how we schedule the job of maximum heat contribution. It also holds that for any algorithm, $\Theta_{\max} \leq h_{max}$, with $\Theta_{\max}$ being the maximum temperature of the algorithm's schedule. Therefore, $\Theta_{\max} \leq 2 \cdot \Theta^*_{\max}$.

To improve this trivial ratio we propose the Algorithm MAX_T below, which is based on the intuitive idea of alternating the execution of hot and cool jobs.

Algorithm MAX_T

```
1: Sort the jobs in non-increasing order of their heat contributions: $h_1 \geq h_2 \geq ... \geq h_n$;
2: Using the order of Step 1, schedule the $\lceil \frac{d}{2} \rceil m$ hottest jobs to the odd slots of the processors using Round-Robin;
3: Using the reverse order of Step 1, schedule the $\lfloor \frac{d}{2} \rfloor m$ coolest jobs to the even slots of the processors using
   Round-Robin;
```

To elaborate a little more on how the algorithm works, note that processor 1 will be assigned the job J_1, followed by J_n, then followed by J_{m+1}, and then by J_{n-m} and this alternation of hot and cool jobs will continue till the end of the schedule. Similarly processor 2 will be assigned the jobs J_2, J_{n-1}, J_{m+2}, J_{n-m-1}, and so on. The schedule is illustrated further in Table 1.

1	J_1	J_n	J_{m+1}	J_{n-m}	J_{2m+1}	...
2	J_2	J_{n-1}	J_{m+2}	J_{n-m-1}	J_{2m+2}	...
...	...	...	...	...	...	...
m	J_m	J_{n-m+1}	J_{2m}	J_{n-2m+1}	J_{3m}	...

Table 1 The schedule produced by Algorithm MAX_T.

To analyze the Algorithm MAX_T, we start with the proposition below, which is implied by the Round-Robin scheduling of jobs in Steps 2 and 3 of the algorithm.

Proposition 2 *In the schedule returned by Algorithm* MAX_T*:*
(i) A job J_i, $i \geq (\lfloor \frac{d}{2} \rfloor + 1)m + 1$, is succeeded by the job $J_{n-i+m+1}$.
(ii) A job J_i, $m + 1 \leq i \leq \lceil \frac{d}{2} \rceil m$, is preceded by the job $J_{n-i+m+1}$.

The maximum temperature may occur at various points of the schedule of Algorithm MAX_T. The next lemma states that one of these points satisfies a certain property regarding the heat contribution of the job executed right before.

Lemma 2 *In the schedule returned by Algorithm* MAX_T*, the maximum temperature is achieved after the execution of a job J_i, with $i \leq (\lfloor \frac{d}{2} \rfloor + 1)m$.*

Proof Assume that all the points where the maximum temperature $\Theta_{\max}$ occurs are after the execution of a job J_i, with $i \geq (\lfloor \frac{d}{2} \rfloor + 1)m + 1$. By Proposition 2, such a job is succeeded by a job $J_{i'}$, $i' = n - i + m + 1$, in the schedule returned by Algorithm MAX_T. It is easy to check that $i > i'$, hence $h_{i'} \geq h_i$. Let $\Theta, \Theta' \leq \Theta_{\max}$ be the temperatures before the execution of J_i and after the execution of $J_{i'}$, respectively.

Then, $\Theta_{\max} = \frac{\Theta + h_i}{2}$ and $h_i \geq \Theta_{\max}$, since $\Theta_{\max} \geq \Theta$. Moreover, $\Theta' = \frac{\Theta_{\max} + h_{i'}}{2} \geq \Theta_{\max}$, since $h_{i'} \geq h_i$. This implies that $\Theta' = \Theta_{\max}$, since $\Theta' \leq \Theta_{\max}$. But this means that the maximum temperature is also achieved after the execution of job $J_{i'}$, which is a contradiction because

$$i' = n - i + m + 1 \leq m(d - \lfloor \tfrac{d}{2} \rfloor) \leq m(\lfloor \tfrac{d}{2} \rfloor + 1)$$

contrary to what we assumed in the beginning of the proof. □

Lemma 3 *For the maximum temperature of an optimal schedule it holds that* $\Theta^*_{max} \geq \frac{h_{n-i+m+1}}{4} + \frac{h_i}{2}$, *for any* $i \geq m + 1$.

Proof Consider a job J_i and let $J_{i'}$ be its previous job in the same processor in an optimal schedule S^*. The jobs executed in the first slot of each processor in S^* do not have a previous one. To simplify the presentation of our proof, we assume that they are preceded by hypothetical jobs J_{n+j}, $1 \leq j \leq m$.

If $i' \leq n - i + m + 1$, then $\Theta^*_{max} \geq \frac{h_{i'}}{4} + \frac{h_i}{2} \geq \frac{h_{n-i+m+1}}{4} + \frac{h_i}{2}$, since $h_{i'} \geq h_{n-i+m+1}$.

If $i' > n - i + m + 1$, then let $B = \{J_{n-i+m+2}, J_{n-i+m+3}, \ldots, J_n, J_{n+1}, \ldots, J_{n+m}\}$ and let A be the set of jobs that precede the jobs $J_1, J_2, \ldots, J_{i-1}$ in the optimal schedule. Clearly, $|B| = |A| = i - 1$, $J_{i'} \in B$ and $J_{i'} \notin A$ since $J_{i'}$ precedes J_i in S^*.

Therefore, there is a job $J_{k'} \in A$ such that $J_{k'} \notin B$, that is $k' < n - i + m + 2$. The job $J_{k'}$ precedes a job J_k in S^* and since $J_{k'} \in A$ it follows, by the definition of the set A, that $k < i$. Hence, $\Theta^*_{max} \geq \frac{h_{k'}}{4} + \frac{h_k}{2} \geq \frac{h_{n-i+m+1}}{4} + \frac{h_i}{2}$, since $h_k \geq h_i$ and $h_{k'} \geq h_{n-i+m+1}$. □

Theorem 4 *Algorithm* MAX_T *achieves a* $\frac{4}{3}$ *approximation ratio for* $P|p_i = 1, h_i, d|\Theta_{max}$.

Proof By Lemma 2 the maximum temperature in the schedule, S, obtained by Algorithm MAX_T occurs after the execution of a job J_i, $i \leq (\lfloor \frac{d}{2} \rfloor + 1)m$ (the maximum may be achieved in other timeslots as well).

If $1 \leq i \leq m$, then the maximum occurs at the first processor and $\Theta_{\max} = \frac{h_1}{2} \leq \Theta^*_{\max}$ and, hence, the algorithm returns an optimal schedule.

If $m + 1 \leq i \leq \lceil \frac{d}{2} \rceil m$ then by Proposition 2, the job J_i is preceded in the schedule S by the job $J_{n-i+m+1}$. Let Θ be the temperature before the execution of the job $J_{n-i+m+1}$. By Lemma 3, and since $\Theta \leq \Theta_{\max}$, $\Theta_{\max} = \frac{\Theta}{4} + \frac{h_{n-i+m+1}}{4} + \frac{h_i}{2} \leq \frac{\Theta_{\max}}{4} + \Theta^*_{\max}$. Hence, $\Theta_{\max} \leq \frac{4}{3} \cdot \Theta^*_{\max}$.

Note that if d is odd, then $\lceil \frac{d}{2} \rceil m = (\lfloor \frac{d}{2} \rfloor + 1)m$ and the analysis of the previous case holds. Hence the only remaining case is that d is even and $\lceil \frac{d}{2} \rceil m + 1 \leq i \leq (\lfloor \frac{d}{2} \rfloor + 1)m$. For this case, let $\Theta' \leq \Theta_{\max}$ be the temperature before the execution of J_i. Then, $h_i \geq \Theta_{\max}$, since $\Theta_{\max} = \frac{\Theta' + h_i}{2}$ and $\Theta_{\max} \geq \Theta'$. Thus, there are at least $\lceil \frac{d}{2} \rceil m + 1$ jobs of heat contribution at least $\Theta_{\max}$. Note that, in any schedule, each processor can execute at most $\lceil \frac{d}{2} \rceil$ jobs without any pair of them scheduled in two consecutive slots. Hence, in an optimal schedule, there are at least two jobs J_p and J_q, $p, q \leq i$, of heat contribution at least $\Theta_{\max}$ executed in consecutive slots in the same processor. Therefore, $\Theta^*_{\max} \geq \frac{h_p}{4} + \frac{h_q}{2} \geq \frac{\Theta_{\max}}{4} + \frac{\Theta_{\max}}{2} = \frac{3}{4} \cdot \Theta_{\max}$, that is $\Theta_{\max} \leq \frac{4}{3} \cdot \Theta^*_{\max}$. □

For the tightness of the analysis of Algorithm MAX_T consider an instance of m processors, mn^2 jobs and $d = n^2$; suppose that there are mn hot jobs of heat contribution $h = 2$ and $mn(n-1)$ cool jobs of heat contribution $h = \epsilon$. We consider n to be sufficiently large and that ϵ tends to 0. The algorithm in each processor alternates n hot jobs with $n - 1$ cool jobs and schedules $n(n-2) + 1$ cool jobs at the end. The maximum temperature of the algorithm's schedule is attained exactly after the execution of the last hot job on each processor. This job is executed at slot $2n - 1$, and thus $\Theta_{max} = \frac{2}{2^{2n-1}} + \frac{\epsilon}{2^{2n-2}} + \frac{2}{2^{2n-3}} + \frac{\epsilon}{2^{2n-4}} + \ldots + \frac{\epsilon}{2^2} + \frac{2}{2^1} \simeq 2\frac{\frac{1}{2}}{1-\frac{1}{4}} = \frac{4}{3}$. On the other hand, the optimal solution alternates in each processor a hot job with $n-1$ cool jobs. The temperature before the execution of any hot job tends to zero and the maximum temperature is one.

4 Average Temperature Minimization

In this section, we look at the problem of minimizing the average temperature, $P|p_i = 1, h_i, d| \sum \Theta_t^j$, instead of the maximum temperature. We will again consider a schedule length d and assume that the number of jobs is $n = md$. Contrary to the maximum temperature, we show that minimizing the average temperature of a schedule is solvable in polynomial time. Our algorithm is based on the following lemma.

Lemma 4 *In any optimal solution for the average temperature, jobs are scheduled in a coolest first order, i.e., for any pair of jobs J_i, J_j such that $h_i > h_j$ scheduled at slots t and t', respectively, it holds that $t' \leq t$, regardless of the processor they are assigned to.*

Proof Consider the job J_i to be scheduled at slot t of some processor p in a schedule S. The contribution of job J_i to the temperature of the s-th slot of processor p (with $t \leq s \leq d$), is $\frac{h_i}{2^{s-t+1}}$, while this job does not affect the temperature of any other slot in any processor. Hence, the contribution of job J_i to the objective function, $\sum \Theta_i$, of schedule S is

$$\sum_{s=t}^{d} \frac{h_i}{2^{s-t+1}} = h_i \cdot \sum_{s=1}^{d-t+1} \frac{1}{2^s} = h_i \cdot (1 - \frac{1}{2^{d-t+1}}) = h_i \cdot \frac{2^{d+1} - 2^t}{2^{d+1}}.$$

Therefore, the later job J_i is scheduled, the smaller its contribution to the objective function becomes.

Assume, now, that in an optimal schedule S^* the job J_i is scheduled at slot t of some processor, while the job J_j at slot $t' > t$ in any processor. By swapping the execution of this pair of jobs the contribution of the job J_i to the objective function decreases by $h_i \cdot \frac{2^{t'} - 2^t}{2^{d+1}}$ and the contribution of job J_j increases by $h_j \cdot \frac{2^{t'} - 2^t}{2^{d+1}}$. As $h_i > h_j$, it follows that the resulting schedule contradicts the optimality of the schedule S^* and this completes the proof of the lemma. □

The previous lemma leads directly to the next simple algorithm.

Algorithm AVR_T

```
1: Sort the jobs in non-decreasing order of their heat contributions: $h_1 \leq h_2 \leq ... \leq h_n$;
2: According to this order schedule the jobs to processors using Round-Robin;
```

Algorithm AVR_T finds a schedule in $O(n \log n)$ time. The optimality of this schedule follows directly by the Round-Robin scheduling of the jobs in non-decreasing order of their heat contributions and Lemma 4.

Theorem 5 *An optimal schedule for the problem $P|p_i = 1, h_i, d| \sum \Theta_t^j$ of minimizing the average temperature can be found in polynomial time.*

4.1 Weighted Average Temperature Minimization

In what follows, we consider a time-dependent weighted version of average temperature minimization. In particular, we consider each slot of every processor to be associated with a given positive weight w_t, $1 \leq t \leq d$, and our problem is denoted as $P|p_i = 1, h_i, d| \sum w_t^j \Theta_t^j$. The weights w_t could represent the interest of the system manager to keep its processors/computers cool during specific time periods of peak loads. This leads to some special, but more practical cases, of our formulation where the weights of some slots (e.g., the slot corresponding to some given time t in all processors, or an interval of consecutive slots for some processor) could be considered equal. Moreover, our analysis allows the weight of the t-th slot of processor j to depend on the processor too and we denote this by w_t^j, $1 \leq t \leq d$, $1 \leq j \leq m$.

Similarly with the un-weighted case, we consider a job J_i of heat contribution h_i scheduled in the t-th slot of processor j in a schedule S. The contribution of this job to the weighted temperature of the s-th slot of processor j, with $t \leq s \leq d$, is $w_s^j \cdot \frac{h_i}{2^{s-t+1}}$, and this job does not affect the temperature of any other slot in any processor. Hence, the contribution of job J_i to the total weighted temperature of the schedule S is $\sum_{s=t}^{d} w_s^j \cdot \frac{h_i}{2^{s-t+1}} = h_i \cdot \sum_s^d \frac{w_s^j}{2^{s-t+1}}$. Clearly, the quantity $c_t^j = \sum_s^d \frac{w_s^j}{2^{s-t+1}}$ is a constant that depends only on the slot t of processor j and not on the job executed in this slot.

Based on this, we transform our problem to a weighted bipartite matching problem and we prove the next theorem.

Theorem 6 *The problem $P|p_i = 1, h_i, d| \sum w_t^j \Theta_t^j$ of minimizing the weighted average temperature is polynomially solvable.*

Proof We transform the problem to a weighted bipartite matching problem. Consider a complete bipartite graph $G = (V, U; E)$ where the vertices in V correspond to the n jobs and the vertices in U to the $m \cdot d$ slots available in all processors. We set the weight of the edge between a job J_i and the slot t of processor j to be equal to $h_i \cdot c_t^j$. Hence, the weight of this edge represents the contribution of job J_i to the objective function, if it is scheduled in slot t of processor j. A perfect matching in the graph G corresponds to a feasible schedule and the weight of such a matching to the value of the objective function for this schedule. Therefore, a minimum weight perfect matching corresponds to an optimal solution for our problem. Such a matching can be found in polynomial time (see for example [10]). □

5 Conclusions

We have provided algorithms as well as negative results for various optimization criteria in scheduling under thermal management models. There are many interesting open questions remaining. The most important is to improve the approximation ratio both for the problem of minimizing the makespan and for minimizing the maximum temperature. Also it would be interesting to generalize our results in the case where the cooling effect is different than one half, as in [7,6,8]. Towards a different direction, one can also consider other objectives under the threshold thermal model, in line with the objectives that have been studied in the more traditional models of job scheduling. Resolving these questions seems technically more challenging than the classic scheduling problems due to the different nature of the constraints that are introduced by temperature management models. Note that scheduling problems under the threshold thermal model can be seen as scheduling problems with sequence-dependent setup times; such a setup time for a job corresponds to the idle slots required to respect the temperature threshold. In scheduling problems with setup times (see for example [16]), the setup time of a job usually depends only on the job itself and the previous job in the schedule. However, in our case, the number of idle slots, required before executing a job, depends on all the jobs scheduled before as well as on their order. Hence existing results from the literature cannot be applied.

Acknowledgment. We would like to thank the two anonymous referees for their thorough and valuable comments.

References

1. S. Albers. Energy-efficient algorithms. *Communications of the ACM*, 53:86–96, 2010.

2. S. Albers. Algorithms for dynamic speed scaling. In T. Schwentick and C. Dürr, editors, *28th International Symposium on Theoretical Aspects of Computer Science (STACS'11)*, volume 9 of *LIPIcs*. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2011.
3. N. Alon, Y. Azar, G. Woeginger, and T. Yadid. Approximation schemes for scheduling on parallel machines. *Journal of Scheduling*, 1:55–66, 1998.
4. L. Atkins, G. Aupy, D. Cole, and K. Pruhs. Speed scaling to manage temperature. In A. Marchetti-Spaccamela and M. Segal, editors, *1st International ICST Conference on Theory and Practice of Algorithms in Computer Systems (TAPAS'11)*, volume 6595 of *LNCS*, pages 9–20. Springer, 2011.
5. N. Bansal, T. Kimbrel, and K. Pruhs. Speed scaling to manage energy and temperature. *Journal of the ACM*, 54(1):Article 3, 2007.
6. M. Birks, D. Cole, S. P. Y. Fung, and H. Xue. Online algorithms for maximizing weighted throughput of unit jobs with temperature constraints. In M. J. Atallah, X-Y. Li, and B. Zhu, editors, *Joint International Conference on Frontiers in Algorithmics and Algorithmic Aspects in Information and Management (FAW-AAIM'11)*, volume 6681 of *LNCS*, pages 319–329. Springer, 2011.
7. M. Birks and S. P. Y. Fung. Temperature aware online scheduling with a low cooling factor. In J. Kratochvíl, A. Li, J. Fiala, and P. Kolman, editors, *7th Annual Conference on Theory and Applications of Models of Computation (TAMC'10)*, volume 6108 of *LNCS*, pages 105–116. Springer, 2010.
8. M. Birks and S. P. Y. Fung. Temperature aware online algorithms for scheduling equal length jobs. In M. J. Atallah, X-Y. Li, and B. Zhu, editors, *Joint International Conference on Frontiers in Algorithmics and Algorithmic Aspects in Information and Management (FAW-AAIM'11)*, volume 6681 of *LNCS*, pages 330–342. Springer, 2011.
9. M. Chrobak, Ch. Dürr, M. Hurand, and J. Robert. Algorithms for temperature-aware task scheduling in microprocessor systems. In R. Fleischer and J. Xu, editors, *4th International Conference on Algorithmic Aspects in Information and Management (AAIM'08)*, volume 5034 of *LNCS*, pages 120–130. Springer, 2008.
10. H. N. Gabow. A scaling algorithm for weighted matching on general graphs. In *26th Annual Symposium of the Foundations of Computer Science (FOCS'85)*, pages 90–100. IEEE Computer Society, 1985.
11. M. R. Garey and D. S. Johnson. *Computers and intractability: A guide to the theory of NP-completeness.* Freeman, San Francisco, 1979.
12. R. L. Graham. Bounds on multiprocessing timing anomalies. *SIAM Journal on Applied Mathematics*, 17:416–426, 1969.
13. R. L. Graham, E. L. Lawler, J. K. Lenstra, and A. H. G. Rinnooy Kan. Optimization and approximation in deterministic sequencing and scheduling. *Annals of Discrete Mathematics*, 5:287–326, 1979.
14. D. S. Hochbaum and D. B. Shmoys. Using dual approximation algorithms for scheduling problems: theoretical and practical results. *Journal of the ACM*, 34:144–162, 1987.
15. S. Irani and K. R. Pruhs. Algorithmic problems in power management. *ACM SIGACT News*, 36:63–76, 2005.
16. M. Pinedo. *Scheduling: Theory, Algorithms and Systems.* Prentice-Hall, 1995.
17. J. Yang, X. Zhou, M. Chrobak, Y. Zhang, and L. Jin. Dynamic thermal management through task scheduling. In *IEEE International Symmposium on Performance Analysis of Systems and Software (ISPASS'08)*, pages 191–201. IEEE Computer Society, 2008.
18. S. Zhang and K. S. Chatha. Approximation algorithm for the temperature-aware scheduling problem. In G. G. E. Gielen, editor, *IEEE/ACM International Conference on Computer-Aided Design (ICCAD'07)*, pages 281–288. IEEE Press, 2007.
19. X. Zhou, J. Yang, M. Chrobak, and Y. Zhang. Performance-aware thermal management via task scheduling. *ACM Transactions on Architecture and Code Optimization*, 7:1–31, 2010.