Algorithms for the Thermal Scheduling Problem

Koyel Mukherjee, Samir Khuller and Amol Deshpande
Department of Computer Science
University of Maryland
College Park, USA
{koyelm,samir,amol}@cs.umd.edu

***Abstract*—The energy costs for cooling a data center constitute a significant portion of the overall running costs. Thermal imbalance and hot spots that arise due to imbalanced workloads lead to significant wasted cooling effort – in order to ensure that no equipment is operating above a certain temperature, the data center may be cooled more than necessary. Therefore it is desirable to schedule the workload in a data center in a *thermally aware* manner, assigning jobs to machines not just based on local load of the machines, but based on the overall thermal profile of the data center. This is challenging because of the spatial cross-interference between machines, where a job assigned to a machine may impact not only that machine's temperature, but also nearby machines.**

Here, we continue formal analysis of the *thermal scheduling* problem that we initiated recently [25]. In that work, the notion of *effective load of a machine* which is a function of the local load on the machine as well as the load on nearby machines, was introduced, and optimal scheduling policies for a simple model (where cross-effects are restricted within a rack) were presented, under the assumption that jobs can be split among different machines. Here we consider the more realistic problem of *integral* assignment of jobs, and allow for cross-interference among different machines in adjacent racks in the data center. The integral assignment problem with cross-interference is NP-hard, even for a simple two machine model. We consider three different heat flow models, and give constant factor approximation algorithms for maximizing the number (or total profit) of jobs assigned in each model, without violating thermal constraints. We also consider the problem of minimizing the maximum temperature on any machine when all jobs need to be assigned, and give constant factor algorithms for this problem.

***Keywords*-data centers, scheduling, algorithms**

I. INTRODUCTION

Modern data centers consist of thousands of computers closely packed in a dense space, typically arranged as hundreds of *racks* of processors. This results in increased power density leading to higher temperatures of machines. The energy costs of a data center have been compared to that of a small town, with a significant portion contributed by the cost incurred in cooling the machines [4]. The energy cost of cooling is directly driven by the *supply temperature* (denoted T_{sup}) of the cold air being blown in to cool the data center – the incoming air is often kept at a lower than necessary temperature to prevent hotspots from forming since those can damage the hardware. For instance, it has been observed that servers near the top of a rack often run hotter and are subject to higher failure rates [15]. Thermal balancing through judicious task scheduling can lead to fewer hotspots and thus lower overall cooling costs and lower failure rates. Similarly in multi-core chip architectures, the increasing density of cores and a movement toward 3D architectures [37] has made *thermal management* a key challenge. Increasing temperatures affect circuit reliability and longevity over the long term, and result in increased power consumption (because of increased leakage power) and overall high cooling costs.

In this framework, we address a basic scheduling problem, referred to as the *thermal scheduling* problem, where we want to maximize the number of assigned jobs while keeping the maximum machine temperature below a pre-specified value. Alternatively, we may want to minimize the maximum machine temperature when all jobs need to be assigned. The key differentiating factor here from the existing scheduling literature is the notion of *spatial cross-interference*: the heat generated by jobs running on a machine raises that machine's temperature as well as the temperatures of nearby machines due to recirculation effects. These cross effects are often asymmetric depending on the data center *geometry*.

In traditional scheduling models, one tries to optimize various measures like makespan, completion time, tardiness etc. while scheduling jobs on multiple machines. In all these models the scheduling cost (or load) of a machine simply depends on its *local* load. For thermal scheduling problems, there is a fundamental shift in this assumption due to the cross effects. Hence this new paradigm of scheduling and packing problems in the presence of cross-interference among machines requires new techniques and analysis.

A formal study of the problem of thermal scheduling in presence of spatial cross-interference modeling a single rack of machines was initiated by us recently [25]. In that work, we analyze the case where fractional or continuous assignments are allowed, i.e., a job can be arbitrarily split across a set of machines without any penalty. We also provide justification for the thermal model chosen [25]. In the current work, we continue the formal analysis of the thermal scheduling problem by considering the problem where only integral assignments of jobs are permitted, i.e., a job must be fully assigned to a single machine. We examine this problem for the case of a single rack of machines, which we call the one-dimensional case. We then extend the model to the more general case of multiple racks of machines, which we call the two-dimensional case. In fact, we analyze three different models to capture the spatial cross-interference for the case of multiple racks. The fractional version of all these problems can be solved optimally using linear programming. However, we only need to know the *structure* of an optimum solution in our algorithms for integral assignment. We derive this structure *combinatorially*. We outline our contributions in further detail below.

Contributions and Summary of Results

We first consider the problem where the maximum temperature that any machine can attain is given to us and is a hard constraint that cannot be violated. The jobs are assumed to be long lasting, with release time 0, and we consider the thermal profile under steady state assumption. The jobs additionally may have profits associated with them. Our goal is to devise algorithms for an integral assignment of jobs to machines, maximizing the number of jobs scheduled or the total profit of jobs scheduled, without violating the hard thermal constraints. This problem is NP-hard as it is a generalization of the multiple knapsack problem.

In Section IV, we give a $(\frac{1}{2} - O(\epsilon))$ algorithm for this problem for a single rack of multiple machines with cross interference between them.

We then consider multiple racks, where there is cross interference not only between machines belonging to a single rack, but also between machines belonging to different, adjacent racks. In Section III, we give a formal description of each of the models with cross-interferences considered in this work. In Section V, we first develop the structure of the optimum fractional solution for a natural two dimensional generalization of the one-dimensional model, and then we provide a $(\frac{1}{2} - O(\epsilon))$ algorithm for the integral assignment problem in Section VI. We extend our analysis to two other two-dimensional cross interference models for both the fractional and the integral assignment problems. All these models are motivated by heat redistribution effects commonly observed in data centers. In fact, our work can be considered a general framework for analyzing scheduling problems with structured directional cross-interference effects. The analysis of our algorithms is asymptotically tight.

In Section VII, we further consider the problem of minimizing the thermal makespan for a single rack of machines in presence of spatial cross-interference. In other words, we consider the problem of integrally assigning all the jobs such that the maximum temperature attained by any machine is minimized. Again this problem is NP-hard since it generalizes the minimum makespan problem for multiple machines. We show that Graham's Longest Processing Time algorithm [14] gives a constant factor approximation to this problem and the analysis is asymptotically tight. We then consider the online version of the same problem and show that Graham's List Scheduling algorithm [13] gives a constant factor approximation to the optimum solution. The analysis is again asymptotically tight.

II. Related Work

Thermal scheduling in data centers has been an area of active research in recent years. Spatial cross-interference effects are well-documented (see, e.g., Schmidt and Cruz [28]). Moore et al. [23] suggest a set of heuristics for workload placement and scheduling for controlling hot spots. Tang et al. [31] proposes a cross-interference model to capture heat recirculation in a data center, and that model and its impact on task scheduling has been explored in a series of works since then [30], [32], [26], [33]. In [25], where we introduce the thermal scheduling problem allowing fractional assignments, we follow the thermal model proposed by them. Shi and Srivastava [29] also use a similar model, but focus on the storage units (disks) instead of the compute units (processors). Pakbaznia and Pedram [27], using a similar model, argue that server consolidation (choosing which servers are on) is critical in minimizing the power consumption, and address the combined problem of task scheduling and server consolidation.

Similar to our work here, much of the above work also makes a steady state assumption leading to a stationary temperature profile that is optimized. Some work has considered different cooling models and their impact on task scheduling [33]. Zhang et al. [35] take time into account and give an approximation algorithm for voltage frequency scaling. Fisher et al. [10] use the Fourier's cooling model to model cooling and heating phenomena, and develop algorithms for frequency scaling. Choi et al. [6] observe that the rise- and fall-times of on-chip temperatures are typically an order of magnitude larger than the OS scheduler ticks, and develop an OS-level scheduler to balance the heat and avoid formation of hotspots. Extending our approach to formally analyze temporal effects is a rich area for future work that we are planning to explore.

There have been several algorithmic and theoretical papers exploring the power on and off strategies [3], [16]. Yao et al. [34] and Irani et al. [16] consider the problem of reducing the total energy consumed by controlling processor speed for a single processor. Bansal et al. [1] and Bansal and Pruhs [2] also consider the problem of minimizing energy and temperature for a single processor using speed scaling techniques. Our work is fundamentally different since we consider multiple processors with cross-interference and there is no speed control.

With increasing power density on multi-core chips and a trend toward 3D chip architectures [22] that tend to exhibit high temperatures, micro-level thermal management has seen much work in last few years. In addition to investigations into better cooling technologies, techniques have also been proposed to either reduce the power consumption locally through frequency and voltage scaling, or by dynamically redistributing the workload to handle hotspots. Gomaa et al. [12] propose a technique called *heat-and-run* that uses intelligent thread assignment, and thread migration to address the problem. Coskun et al. [8] use both voltage/frequency scaling and task migration to reduce the frequency of hotspots. Ge et al. [11] propose using local task swaps between neighboring cores to achieve thermal balancing; their thermal model looks very similar to the cross-interference model that we use in this work. Liu et al. [21] also give heuristic solutions to the problem we study. Li et al. [19] also look at thermal management in micro chips, but when jobs have precedence constraints. Liu et al. [20] look at thermal management in presence of stochastic workloads. In an earlier work, Kursun et al. [18] examine the effects of task scheduling on thermal behavior and

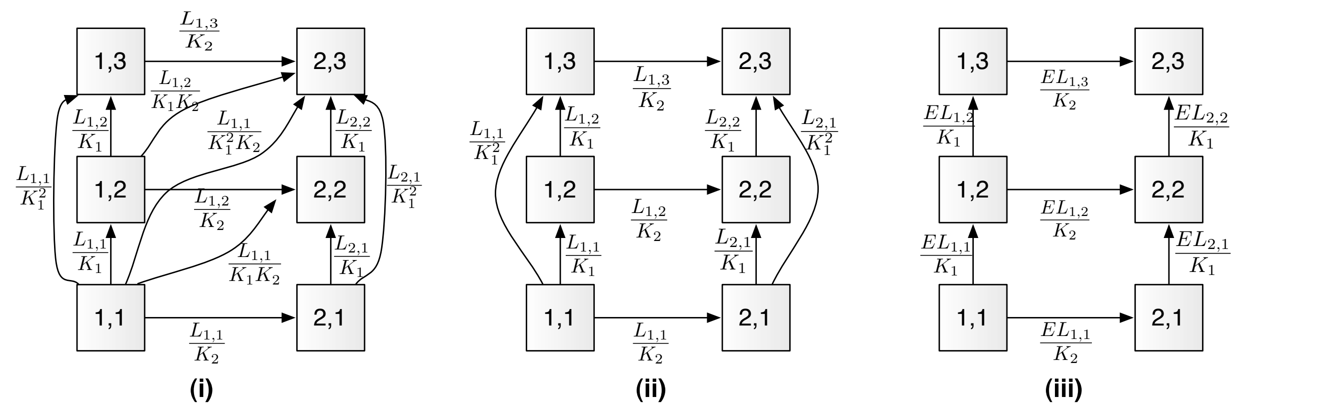

Fig. 1. Models of heating effect shown for two racks each with three machines: (i) General model, (ii) Horizontal Sibling Model, (iii) Indirect Sibling Model.

experimentally show that thermal-aware scheduling policies can alleviate on-chip temperatures. In recent work, Zhou et al. [36] propose and analyze several heuristics for thermal-aware scheduling in 3D chips.

Unlike our work, prior work in spatial cross-interference has either presented heuristics or suggested using ILP solvers to solve the problem, and has not attempted to exploit the structure of the cross-interference matrix to design algorithms with worst case guarantees.

III. Problem Formulation and Model

We formulate the problem of thermal scheduling in terms of "effective load" on a machine [25]. For completeness, we begin with summarizing the key ideas underlying our problem definition here.

For each job to be scheduled, we assume that we can estimate the power that will be consumed to execute it on a machine[1]; this can be computed using the estimated resources required to execute the job, its time duration, and standard system power modeling techniques [9].

We base our thermal model on the abstract heat circulation model suggested by Mukherjee et al. [26]. As with much of the prior work on thermal scheduling, we assume that the system is in steady state; i.e., we assume the jobs are long-lived, and analyze the system state when all the jobs have arrived and the temperatures have stabilized. According to the model, the temperature of a machine i is given by: $T_i = T_{sup} + D_i\mathbf{L}$, where T_{sup} is the supply temperature, D_i is the i^{th} row of the heat distribution matrix D, and $\mathbf{L}$ is the load vector. The vector $\mathbf{L} = \{L_1, \cdots, L_m\}$, where m is the number of machines, denotes the loads on the machines in terms of the *power* consumed by the jobs assigned to the machine. The matrix D represents how the heat or load of any machine j affects machine i (called *cross-interference*). Since the temperature of no machine should exceed T_{red}, we have that: $T_{sup} + \max_{i\in[1,\ldots,m]} D_i\mathbf{L} \leq T_{red}$.

Given this, *effective load* on a machine is computed as a linear combination of the load on the machine itself and loads on other machines. Specifically, given that the load on machine i is L_i, and the effect of machine j's load on machine i is captured through the cross-interference coefficient $D_{i,j}$, the effective load EL_i is computed as follows: $EL_i = \sum_j D_{i,j}L_j$ where $0 \leq D_{i,j} \leq 1$ and $D_{i,i} = 1$. Under a steady state assumption, the effective load on a machine serves as a proxy for the temperature of the machine [25].

[1]We use the terms *machines* and *processors* interchangeably.

A. Single rack of machines: One Dimensional Model

In the simplest case, the machines are arranged in a linear array with cold air blowing from one end. This models the behavior of a single rack with the cold air blowing from the floor [28] and the warm air moving into a vent in the ceiling and back to the HVAC unit.

In this simple model, which we call the *one-dimensional model*, heat is recirculated in one direction. We number machines from bottom to top, in increasing order from the cold air source. The i^{th} machine is affected only by the heat recirculated from the machines $j \leq i$ located below it.

We assume the heat falls off in an exponential manner. Specifically, the heat felt by a machine i due to machine j is a fraction $\frac{1}{K^d}$ of the load (heat) of j, where $d = |i - j|$ is the distance between i and j and K is a constant > 1. More formally, $D_{i,j} = \frac{1}{K^{|i-j|}}$. For technical reasons we assume that $K \geq 2$. In practice, we expect $K \gg 2$ – a value of $K < 2$ would indicate an unacceptably tight thermal coupling between two adjacent machines and is unlikely to be observed in a data center.

B. Multiple racks of machines: 2-D Models

Here we consider a two dimensional grid, consisting of several adjacent racks, such that there is heat distribution among the machines of one rack, as well as heat distribution laterally between adjacent racks. We assume that a cold air source is located at the bottom of the racks, so the heat flows upward along each rack. We also assume there is a cold air source at one end of the series of racks, so that the heat also flows laterally from one rack to another. The heating effect is felt by a machine from the machines located below it in the same rack, as well as from the machines located on the racks to its left, at the same or lower position on their respective racks.

The racks are numbered in an increasing order from the cold air source and in each rack, machines are numbered from bottom to top. In previous works [12], [21] the lateral heat redistribution is considered much weaker than the vertical one. So, the temperatures of machines located in the same rack are more strongly coupled than those across racks. To capture this effect we define *three* models of heat recirculation in the lateral direction (see Figure 1). In all these models, the number of machines in a single rack is $m_1 \geq 2$ and the number of racks is $m_2 \geq 1$. The total number of machines m is therefore $m_1 m_2$. The heat recirculation coefficient in the vertical direction is called K_1 and in the lateral direction as K_2. We assume $K_1 \geq 2$ and $K_2 \geq 1 + \frac{K_1}{K_1 - 2}$. We now define the $D_{i,j}$ as a function of (i, j, K_1, K_2). The specific formulation depends on the corresponding model.

2-D General Model: In the first model, which we call the *general model*, a machine located at the j^{th} row of the i^{th} rack, where $1 \leq j \leq m_1$ and $1 \leq i \leq m_2$, is affected by all machines located in rows j or lower, of all racks numbered $[1 \dots i]$. The heat redistribution effect from the load on the $(i', j')^{th}$ machine on the $(i, j)^{th}$ machine is $\frac{L_{i'j'}}{{K_1}^{j-j'}\ {K_2}^{i-i'}}$, i.e., the effective load on the $(i, j)^{th}$ machine is:

$$EL_{i,j} = \sum_{i'=1}^{i} \sum_{j'=1}^{j} \frac{L_{i',j'}}{{K_1}^{j-j'}\ {K_2}^{i-i'}}$$

2-D Horizontal Sibling Model: Here the heat redistribution in the vertical direction is as before, but the lateral redistribution is restricted to a single level. The vertical effect falls off exponentially as K_1^d, and the lateral effect falls off as K_2. The effective load on the $(i, j)^{th}$ machine is given as follows.

$$EL_{i,j} = \frac{L_{i-1,j}}{K_2} + \sum_{\ell=1}^{j} \frac{L_{i,\ell}}{{K_1}^{j-\ell}}$$

2-D Indirect Sibling Model: In the third model, which we call the *indirect sibling model*, the effective load on the $\left(i^{th}, j^{th}\right)$ machine is defined as

$$EL_{i,j} = L_{i,j} + \frac{EL_{i,j-1}}{K_1} + \frac{EL_{i-1,j}}{K_2}$$

The definition of this model is inspired by the effective load behavior in the one dimensional case [25].

Fig. 1 illustrates the heating effect in the three models.

C. Thermal Scheduling Problem

Given this background, we consider the following problem. We are given a set of jobs, with job j having size s_j – the size of a job denotes the (thermal) load caused by it on the assigned machine. The jobs have release times 0. We are also given a set of machines, and a cross-interference model depending on the geometric configuration of the data center. For the first problem, our goal is to maximize the number of jobs that can be scheduled, given upper bounds on the maximum effective load that any machine can handle (we call this the **maximum cardinality problem**). A variation of this problem is where each job is assigned a profit, and our goal is to maximize the total profit instead (we call this the **maximum profit problem**). For the second problem, we have to assign all jobs, and our goal is to minimize the maximum effective load on any machine (called the **minimum thermal makespan problem**).

IV. Single Rack of Machines: 1-D model

We begin with analyzing the simple one-dimensional model and develop algorithms for the integral version of the problem.

A. Fractional Assignments

Recall that in the fractional assignment case, a job may be arbitrarily split among a set of machines. For this case for the one-dimensional model, the following claims about the behavior of effective load were shown in [25]. We list these claims here as they will be used later for developing algorithms for integral assignments and for 2-D case. Claim 1 relates the effective load on adjacent machines, whereas Claim 2 gives us an analytical expression for the effective load on any machine due to an optimal load distribution strategy.

Claim 1: $EL_i = L_i + \frac{EL_{i-1}}{K}$ for all $i > 1$.

Claim 2: An optimal strategy for minimizing the maximum effective load for fractional assignments, with total load L, would result in uniform effective load of $EL = \frac{L}{m - \frac{m-1}{K}}$.

Using these claims, we can also prove the following claim.

Claim 3: Let c be the capacity constraint for the processors. An optimal strategy packs the first machine up to c and all others up to $c(1 - \frac{1}{K})$ until it runs out of machines or load.

In other words, the optimal capacity distribution (in terms of load assigned to the machines) is $c, c\left(1 - \frac{1}{K}\right), c\left(1 - \frac{1}{K}\right), \dots$.

B. Integral Assignments

Here jobs need to be integrally assigned to machines and our goal is to maximize the number of jobs scheduled without violating thermal constraints.

The optimum fractional strategy (Claim 3) packs the first machine to c and other machines to $c\left(1 - \frac{1}{K}\right)$. However for *integral* assignments, that may not be achievable. Suppose in an optimum packing, machine 1 is occupied up to $c' < c$. The available capacity in machine 2 would be $c - \frac{c'}{K} > c(1 - \frac{1}{K})$ that might allow machine 2 to fit an extra job. So the optimal capacity distribution is not a straightforward pattern of $c, c(1 - \frac{1}{K}), \dots c(1 - \frac{1}{K})$ anymore. Suppose in the optimal packing, the i^{th} machine is filled up to c_i. We refer to the sequence $c_1, c_2, ..., c_m$, as a "pattern" or "layout" of capacities, interchangeably. If $c_1 < c$, we refer to $\delta_1 = c - c_1$ as the *gap* in machine 1. Similarly, if $c_i < c(1 - \frac{1}{K})$ for $i > 1$, $\delta_i = c(1 - \frac{1}{K}) - c_i$ is the gap in machine i.

Lemma 1: There exists an optimal packing with the property that if machine i has an effective load $EL_i \leq c - \Delta$ where Δ is the size of the largest job, then the effective load in all machines j $(j > i)$ is $EL_j \leq c - \Delta$.

Proof: Suppose this is not true. In other words, in every optimal packing for a given instance of the problem, there exists a machine i such that $EL_i \leq c - \Delta$ and there is a machine j ($j > i$) such that $EL_j > c - \Delta$. Consider one such optimum packing. Let i be the machine with $EL_i \leq c - \Delta$. Select the smallest numbered machine j, where $EL_j > c - \Delta$. We can assume now that $j = i + 1$. Move jobs greedily from machine $i+1$ to machine i until $EL_i > c - \Delta$. The assumption on Δ ensures that the packing on i remains feasible at each step. If the total shifted load is s then the effective load on i goes up by s, and the load on $i+1$ goes down by s. However the effective load on $i+1$ goes up by $\frac{s}{K}$, but the drop in the real load of s compensates for this and the new effective load is only lower than the effective load initially. In a similar way, the effective load of all machines with higher indices also decreases. In other words, we now have an optimal packing that satisfies the property of the lemma, contradicting the initial assumption. ■

NOTE: This lemma holds more generally for *any function* where the effect of i on j is monotonically decreasing as j gets further away from i.

However, this does not mean that for a set of jobs with arbitrary sizes, the optimal packing capacity "pattern" is monotonically decreasing or of any regular form, as in the fractional case. The optimal strategy might have a capacity pattern that is quite irregular, decreasing in the middle, and again rising at the ends, or any other arbitrary pattern, and this is the case even if we assume a limit on the sizes of the objects.

The following example shows the optimal pattern might be complicated having "gaps" (less than the size of any object) in two consecutive machines that helps to accommodate an extra object in the third one. These gaps get created because objects need to be assigned integrally. However, as can be seen in the example, these gaps were necessary in order to fit all the objects without thermal violation.

Example: Let $K = 4$. There are 4 processors, we have one object of size c, 3 objects of size $0.2292c$, 3 objects of size $0.2135c$, and 4 objects of size $0.195c$. An optimal arrangement fitting them all is the object of size c in processor 1, 3 objects of size $0.2292c$ in processor 2, 3 objects of size $0.2135c$ in processor 3, and 4 objects of size $0.195c$ in processor 4. This gives an actual load capacity layout of $c, 0.6876c, 0.6405c, 0.78c$. With a little effort it can be seen that this is an optimal packing. The optimal capacity pattern can therefore be quite arbitrary.

We consider a further generalization of the problem where jobs additionally have profits associated with them. We want to maximize the profit of jobs assigned integrally without violating the effective load capacity or thermal constraint. Henceforth we refer to this problem as the *maximum profit* problem. We next describe an algorithm (Algorithm 1) for maximizing profit.

Lemma 2: Algorithm 1 produces a thermally feasible packing.

Proof: The proof follows from Claim 3. ■

Let Δ be the size of the largest job. Let μ be a lower bound on the number of items that can be packed in any machine i under the fixed capacity pattern. We assume $\mu = \lfloor \frac{c\left(1-\frac{1}{K}\right)}{\Delta} \rfloor \geq$ 1, i.e., we assume that the largest job can fit on any machine under modified capacities. In practice μ is likely to be much greater than 1 and the approximation guarantee would increase monotonically with μ.

Algorithm 1 Algorithm for the One Dimensional Model

```
1. Fix the capacity pattern as $c$ for machine 1, and $c\left(1 - \frac{1}{K}\right)$ for machines $[2 \ldots m]$.

2. Run a PTAS for multiple knapsack [5] [17] using the modified machine capacities on the instance I.
```

Theorem 1: Algorithm 1 produces a $\frac{\mu}{\mu+1}(1 - O(\epsilon))$ approximation to the maximum profit problem for a fixed $\epsilon > 0$ in polynomial time. For $\mu \geq 1$, $\frac{\mu}{\mu+1} \geq \frac{1}{2}$. The factor of $\frac{1}{2}$ is asymptotically tight.

To prove Theorem 1, we first prove the following lemma

Lemma 3: There exists a packing of jobs that would lose at most one job per machine compared to an optimum solution, when we fix the actual load capacities of the machines as: $c_1 = c$ and $c_i = c\left(1 - \frac{1}{K}\right),\ i \in [2 \ldots m]$.

Proof: An optimal solution OPT, packs some machines i to an extent lower than c_i. We refer to such machines as *underpacked*. Similarly, OPT packs some machines j to an extent greater than c_j. We refer to such machines as *overpacked*. If there are no overpacked machines, then there is nothing further to prove in this lemma.

Let i be the first overpacked machine and the sum of sizes of jobs packed in i in OPT is c'_i ($c'_i > c\left(1 - \frac{1}{K}\right)$).

First we claim that if there exists a machine i which is overpacked in OPT, then there exists a non-empty set of underpacked machines $j < i$. Suppose there are no underpacked machines $j < i$. That means, the first machine has been packed to c, and machines $[2 \ldots (i-1)]$ are all packed up to $c\left(1 - \frac{1}{K}\right)$. Therefore, from Claim 1, the effective load on every $j < i$ is c, and therefore the effective load on i is $L_i + \frac{c}{K}$. However, i being overpacked, $L_i > c\left(1 - \frac{1}{K}\right)$. Hence, the effective load on i in OPT would be $> c$ which is infeasible.

We order the jobs in machine i in non-decreasing order of size. repacking them in underpacked machines $j < i$ using some heuristic such as *First-Fit*, such that the thermal constraints are not violated. A job k is moved to a machine j, only if $s_k \leq \delta_j$ where δ_j is the size of the gap in machine j before the reassignment. This repacking does not violate thermal feasibility for any machine as proved in Claim 4.

Claim 4: The above repacking does not violate thermal feasibility for any machine.

Proof: The assignment of a job k to a machine j does not violate thermal feasibility by the repacking rule. We prove that the thermal feasibility in other overpacked machines is also not violated by this reassignment.

In machines located at $j + d$, the effect of the gap in j was $\frac{\delta_j}{K^d}$ before the reassignment. Note that $d \geq i - j$, since i is the first overpacked machine. After the reassignment of s_k from i to j, for any overpacked machine $j+d$ the feasibility of the packing is not violated since a job affecting it only moved further away. ∎

Proof of Lemma 3 continued: After repacking some jobs, if machine i is no longer overpacked then we move to the next overpacked machine. Otherwise, the total size of jobs assigned to i still exceeds $c\left(1-\frac{1}{K}\right)$. In other words, the load on i is $L_i = c\left(1-\frac{1}{K}\right)+\varepsilon$, where $\varepsilon > 0$. If after having repacked or reassigned some jobs, the smallest among the remaining jobs in machine i: $s_{i,min}$, must be larger than any of the gaps in any machine j ($j<i$). That is, $s_{i,min} > \delta_{max}$, where δ_{max} is the maximum gap in any $j<i$ after the reassignments. For thermal feasibility, $\varepsilon \le \delta_{max}\left(\frac{1-\frac{1}{K^i}}{1-\frac{1}{K}}-1\right)$. Since $K \ge 2$, $\varepsilon < \delta_{max} < s_{i,min}$. Hence, if we remove at most one job from i (where all the jobs are of size $\ge s_{i,min}$), i will no longer be overpacked.

We then repeat the same procedure for the next overpacked machine, until no machine is overpacked. The thermal feasibility in the succeeding machines is not affected by the repacking of jobs as already proved. Hence, we convert the optimum packing to a packing with the modified capacity pattern, losing at most one job per machine in the process. ∎

Proof of Theorem 1:

Proof: For the maximum cardinality problem where we want to maximize the number of jobs assigned without violating thermal capacity constraints, just arbitrary assignment of the jobs in the machines with the modified capacity pattern would give $\frac{\mu}{\mu+1}$ approximation, since by definition, at least μ jobs can be fitted in every machine with the modified capacity pattern in the given instance. When the objects have profit, if we have access to a multiple knapsack oracle, the same approximation guarantee is achieved. After repacking, we choose the least profit item to discard from the overpacked machines. If the item discarded from machine i has profit p_i, the remaining objects in the machine will have profit $\ge p_i$ and by the assumption on size, there will be at least μ such items in it. We use the multiple knapsack PTAS due to Chekuri and Khanna [5] or Jansen [17] which gives a $1-O(\epsilon)$ approximation to the optimum packing for a given capacity pattern, for a fixed ϵ in polynomial time. ∎

The factor of $\frac{1}{2}$ is asymptotically tight as can be seen in the following example. Let the thermal constraint limit the effective load of any machine to 1 and there are m machines and $2m$ jobs. Suppose the optimum solution packs 2 objects of size $\frac{1}{2}-\varepsilon$ in machine 1, 2 objects of size $\frac{1-\frac{1}{K}}{2}+\frac{\varepsilon}{4K}$ in machine 2, 2 objects of size $\frac{1-\frac{1}{K}}{2}+\frac{\varepsilon}{8K^2}$ in machine 3 and so on, till on machine m it packs 2 jobs of size $\frac{1-\frac{1}{K}}{2}+\frac{\varepsilon}{2^m K^{m-1}}$. Algorithm 1 would miss one object from each of machine $[2\ldots m]$, hence it will give an approximation $\frac{m+1}{2m}=\frac{1}{2}+\frac{1}{m}\approx\frac{1}{2}$ when m is large.

V. Multiple Racks of Machines: Fractional Assignments

In this section we discuss generalizations of the thermal model for a single rack of machines to multiple racks of machines. Specifically, we discuss the three different two-dimensional models, emulating different heat flow effects. We first derive optimal load distribution strategies for fractional assignments. From this analysis it is easy to derive the corresponding three theorems that bound: (a) how much extra load can be assigned due to a thermally aware strategy compared to a naive load distribution strategy subject to a thermal constraint, (b) the reduction in maximum effective load on a machine for a given total load, and (c) the savings in cooling costs. We have omitted some of the more technically involved proofs from this paper due to space constraints. However they can be found in the full version [24].

In Section VI we discuss how to develop approximation algorithms for assigning jobs by fixing the effective load capacities for each machine. This method lets us now completely ignore thermal constraints and reduces the problem (as before) to a multiple knapsack problem. However, the proof technique and analysis are both different and more involved than the one-dimensional case.

A. Two Dimensional General Model

In this model, the heat redistribution effect of the load on the $(i',j')^{th}$ machine on the $(i,j)^{th}$ machine is $\frac{L_{i'j'}}{{K_1}^{j-j'}\ {K_2}^{i-i'}}$ and the effective load on the $(i,j)^{th}$ machine is

$$EL_{i,j}=\sum_{i'=1}^{i}\sum_{j'=1}^{j}\frac{L_{i',j'}}{{K_1}^{j-j'}\ {K_2}^{i-i'}}.$$

The following claim relates the effective load on neighboring machines analogous to the 1D case.

Claim 5: The effective load $EL_{i,j}$ in the general model can be given as $EL_{i,j}=L_{i,j}+\frac{EL_{i,j-1}}{K_1}+\frac{EL_{i-1,j}}{K_2}-\frac{EL_{i-1,j-1}}{K_1\ K_2}$.

Lemma 4 gives the maximum effective load that results due to a thermally aware strategy when fractional assignments are allowed.

Lemma 4: Any optimal strategy for minimizing the maximum effective load for fractional assignments, with total load L, would result in uniform effective load of $\frac{L}{\left(m_1-\frac{m_1-1}{K_1}\right)\left(m_2-\frac{m_2-1}{K_2}\right)}$.

On the other hand, if the load is naively split among all machines equally, the maximum effective load in the system can be much higher and is given by the following lemma.

Lemma 5: A naive strategy minimizing the maximum load for total load L, would split the load uniformly and result in maximum effective load $EL=\frac{L}{m_1\ m_2}\ \frac{1-\frac{1}{{K_1}^{m_1}}}{1-\frac{1}{K_1}}\ \frac{1-\frac{1}{{K_2}^{m_2}}}{1-\frac{1}{K_2}}$.

The corresponding lemmas for the other two models are given in the following sections. The following table gives a comparison of the thermal savings of the three different heat flow models. The parameters used are $K_1=3$, $K_2=5$, $m_1=10$ and $m_2=2$ and load $L=100$. for all three models. The first column gives the maximum effective load due to thermally aware load splitting, the second column gives the maximum effective load due to naive load splitting, and the third column gives the % savings. We note that, even a few degrees reduction in the maximum temperature across the machines may result in significant savings in the cooling costs.

Model	EL-aware	EL-oblivious	% Savings
General	7.936	8.999	11.81
Horizontal	3.279	8.4998	61.4
Indirect	8.333	9.749	14.52

TABLE I
TABLE COMPARING PERCENTAGE SAVINGS IN EFFECTIVE LOAD: $K_1 = 3$, $K_2 = 5$, $m_1 = 10$, $m_2 = 2$, $L = 100$

B. Two Dimensional Horizontal Sibling Model

In this model, the effective load on the $(i,j)^{th}$ machine is given as follows, assuming the machines in each rack are numbered starting from 0 and the racks are also numbered starting from 0.

$$EL_{i,j} = L_{i,j} + \sum_{\ell=0}^{j-1} \frac{L_{i,\ell}}{{K_1}^{j-\ell}} + \frac{L_{i-1,j}}{K_2}.$$

Claim 6 relates the effective load on machines.

Claim 6: The effective load on the $(i,j)^{th}$ machine can be expressed as $EL_{i,j} = L_{i,j} + \frac{EL_{i,j-1}}{K_1} - \sum_{l=0}^{i-1} \left(-\frac{1}{K_2}\right)^{i-l} \sum_{p=0}^{\min(j,i-l+1)} \left(-\frac{1}{K_1}\right)^p \binom{i-l+1}{p} EL_{l,j-p}$.

Lemma 6 gives the maximum effective load that results due to a thermally aware strategy when fractional assignments are allowed.

Lemma 6: Any optimal strategy for minimizing the maximum effective load for fractional assignments, with total load L, would result in uniform effective load $EL = \frac{L}{P}$, where
$P = \sum_{i=0}^{m_2-1} \left(-\frac{1}{K_2}\right)^i (m_2 - i)$
$\left((m_1 - i - 1)\left(1 - \frac{1}{K_1}\right)^{i+1} + \sum_{j=0}^{i} (m_1 - j) \binom{i+1}{j} \left(-\frac{1}{K_1}\right)^j\right)$.

On the other hand, naively splitting load among machines can result in a higher maximum effective load in the system, as given below.

Lemma 7: A naive strategy splitting load uniformly between machines will result in a maximum effective load $EL_{max} = \frac{L}{m_1 m_2}\left[\frac{{K_1}^{m_1}-1}{{K_1}^{m_1-1}(K_1-1)} + \frac{1}{K_2}\right]$.

C. Two Dimensional Indirect Sibling Model

In this model, unlike the above two models, we define the effective load on a machine in terms of the *effective* loads of their neighbors, not the actual loads.

$$EL_{i,j} = L_{i,j} + \frac{EL_{i,j-1}}{K_1} + \frac{EL_{i-1,j}}{K_2}$$

where $i > 1, j > 1$.

$$EL_{1,j} = L_{1,j} + \frac{EL_{1,j-1}}{K_1}$$

where $j > 1$.

$$El_{i,1} = L_{i,1} + \frac{EL_{i-1,1}}{K_2}$$

where $i > 1$, and $EL_{1,1} = L_{1,1}$.

This is a natural way of defining effective load of machine. The effective load is a measure of the temperature of a machine, and the temperature of machine is affected by the temperature of its two immediate neighbors in this model. This recursive definition already takes into account the heat recirculated from previous machines. In fact, this generalizes the one-dimensional model in a way, because, in the 1-D model with a single rack, we had proved in Claim 1 that $EL_i = L_i + \frac{EL_{i-1}}{K}$.

The following claim relates the effective load on machine (i,j) to the actual loads of other machines.

Claim 7: The effective load on machine (i,j) in terms of the *loads* of the other machines is $EL_{i,j} = \sum_{p=0}^{i-1} \sum_{q=0}^{j-1} \binom{p+q}{p} \frac{L_{i-p,j-q}}{{K_1}^p\ {K_2}^q}$.
The following lemma gives the maximum effective load that will result due to a thermally aware strategy when fractional assignments are allowed.

Lemma 8: An optimal strategy minimizing the effective load across all machines when the total load is L would result in an uniform effective load: $EL = \frac{L\ K_1\ K_2}{m_1 K_1 + m_2 K_2 + m_1 m_2 (K_1 K_2 - K_1 - K_2)}$.
If the load is split equally among machines, then the maximum effective load in the system can be quite high, as given by the following lemma.

Lemma 9: A naive strategy that splits load uniformly across all machines results in a maximum effective load $EL_{m_1,m_2} = \frac{L}{m_1\ m_2} \sum_{i=0}^{m_2-1} \sum_{j=0}^{m_1-1} \frac{\binom{i+j}{i}}{{K_1}^j\ {K_2}^i}$.

VI. Multiple Racks of Machines: Integral Assignments

We want to maximize the number or profit of jobs integrally assigned with respect to the thermal constraint, which is the effective load capacity of the machines. Let this capacity be c for all machines. Here we provide a $\left(\frac{1}{2} - O(\epsilon)\right)$ algorithm for the 2D General Model. We also give algorithms with the same approximation ratio for 2D Horizontal and Indirect Sibling Models.

A. General Model

We assume that the maximum object size $\Delta \le c\left(1 - \frac{1}{K_1}\right)\left(1 - \frac{1}{K_2}\right)$. Let us call the instance of jobs as I. Hence, $\mu = \lfloor \frac{c\left(1-\frac{1}{K_1}\right)\left(1-\frac{1}{K_2}\right)}{\Delta} \rfloor \ge 1$. In practice it is much larger for data centers.

Lemma 10: The packing produced by Algorithm 2 is thermally feasible for the general model.

Proof: The proof follows from Claim 5. ■

Theorem 2: When $K_2 \ge 1 + \frac{K_1}{K_1 - 2}$, Algorithm 2 gives a $\left(\frac{1}{2} - O(\epsilon)\right)$ approximation under the general model in time polynomial in input size and any fixed $\epsilon > 0$.

For proving Theorem 2, we initially assume that we have access to a multiple knapsack oracle that returns the optimum packing maximizing the number of jobs when called with

Algorithm 2 Algorithm for General Model

```
1. Fix a capacity pattern:
   $c$ for the machine (1,1),
   $c\left(1-\frac{1}{K_1}\right)$ for machines $(1,j),\ j\in[2\dots m_1]$,
   $c\left(1-\frac{1}{K_2}\right)$ for machines $(i,1),\ i\in[2\dots m_2]$, and
   $c\left(1-\frac{1}{K_1}\right)\left(1-\frac{1}{K_2}\right)$ for the all other machines.
2. Run a PTAS for multiple knapsack [5], [17] using the
   modified machine capacities on the instance $I$.
```

machines of certain fixed capacities. For the one-dimensional case we had shown how a packing, losing at most one object per machine, can be derived from an optimal packing in a polynomial number of operations [25]. However, it is *not* necessary that we show how such a repacking can be done in polynomial time; our purpose is to prove that such a packing *exists* for proving the rest of the theorem.

Before proving the theorem, we first prove the following lemma.

Lemma 11: There exists a packing that loses at most one object per machine, as compared to an optimum solution, when we fix the actual load capacities of the machines as $c_{1,1}=c$, $c_{1,j}=c\left(1-\frac{1}{K_1}\right),\ j\in[2\dots m_1]$, $c_{i,1}=c\left(1-\frac{1}{K_2}\right),\ i\in[2\dots m_2]$, $c_{i,j}=c\left(1-\frac{1}{K_1}\right)\left(1-\frac{1}{K_2}\right),\ i\in[2\dots m_2],\ j\in[2\dots m_1]$.

Proof: Let OPT be an optimum packing. We call the machines (i,j) in which the sum of sizes of assigned objects exceed the assigned capacity $c_{i,j}$ in OPT as *overpacked*, and those in which the sum of sizes is lower than $c_{i,j}$, as *underpacked*. Let O be the set of overpacked machines and U be the set of underpacked machines. If $O=\emptyset$, then we have nothing to prove. Let us therefore assume that there exists a machine $(i,j)\in O$. We first prove the following claim about the sets O and U before proceeding with the proof of the Lemma 11.

Claim 8: If $(i,j)\in O$, then there exists some machine $(i',j')\in U$, such that $i'\le i$ and $j'\le j$, and $(i',j')\neq(i,j)$.

Proof: Suppose for contradiction that there is no such machine (i',j'). Therefore, all the machines $(i',j')\in([1\dots i],[1\dots j])$ are packed such that $L_{i',j'}\ge c_{i',j'}$ for all such (i',j'). However, from Claim 5, it can be verified by induction that the effective load on each of these machines is at least c. But in that case, the effective load in (i,j) is $>c$ (this again follows from Claim 5), which is infeasible. ■

Proof of Lemma 11 continued:
Let $J^t_{i,j}$ denote the set of jobs that are accommodated in an overpacked machine (i,j) in OPT. Since (i,j) is overpacked, $S(J^t_{i,j})=\sum_{\ell\in J^t_{i,j}}s_\ell>c_{i,j}$. Let $S(J^t_{i,j})=c_{i,j}+S_{i,j}$, where $S_{i,j}$ is the "extra capacity" used by (i,j). Among the set of jobs $J^t_{i,j}$, let $J'_{i,j}$ denote a maximal set of jobs such that their total size is $\le c_{i,j}$. Since we assume $\mu\ge 1$, there will be at least one job in the set $J'_{i,j}$ and hence it is well-defined [2]. Let $J''_{i,j}=J^t_{i,j}-J'_{i,j}$.

Our goal is to find the maximal (in terms of cardinality) set of jobs from (i,j) which will completely fit in $S_{i,j}$. In case $S(J''_{i,j})>S_{i,j}$, note that if we discard any job from $J''_{i,j}$, the remaining jobs would fit in $S_{i,j}$. Otherwise, the set $J'_{i,j}$ was not maximal, since we can add at least one more job to it. Let us therefore discard a job from $J''_{i,j}$ to create our new set $J''_{i,j}$. This set is define for every overpacked machine. Now we define a set J as the union of the sets $J''_{i,j}$ for all $(i,j)\in O$. The rest of the proof shows that the set of jobs can be accommodated elsewhere in the packing when we force the machines to have the capacities defined by us.

Let us consider the "gap" $\delta_{i',j'}$ in every underpacked machine $(i',j')\in U$ as a knapsack of capacity $\delta_{i',j'}$. Now we have a set of empty knapsacks of fixed capacity. Let our items in this multiple knapsack problem be the set of jobs in J. Considering each job $\ell\in J$ to be an item of size s_ℓ and profit 1, we call our multiple knapsack oracle to pack these bins or knapsacks optimally. If all the items or jobs have been successfully packed then we have our required packing where each machine (i,j) is packed to an extent $\le c_{i,j}$, with the loss of at most one job per overpacked machine. Note that this repacking would not violate the thermal constraints by definition since the total size of jobs reassigned to a machine (i',j') is $\le\delta_{i',j'}$. Let us call the gaps in each of these knapsacks (i',j'), *(not the machines)*, after the packing to be $\epsilon_{i',j'}$.

If all items or jobs could not be packed, let J_{rem} be the set of jobs that could not be assigned to any of the knapsacks. Let s_{min} be the smallest size of any job in J_{rem}; we know $s_{min}>\epsilon_{max}$, where $\epsilon_{max}\ge\epsilon_{i',j'}$ for $(i',j')\in U$. The total contribution by all the gaps in U is $S\ge S(J)=S(J_{rem})+S(J\setminus J_{rem})$, hence $S(J_{rem})\le S-S(J\setminus J_{rem})$. The contribution of each $\delta_{i',j'}$ in S is
$C_{i',j'}=\delta_{i',j'}\left(\sum_{p=i'}^{m_2}\sum_{q=j'}^{m_1}\frac{1}{{K_1}^{q-j'}{K_2}^{p-i'}}\ -\ 1\right)$
since the contribution of $\delta_{i',j'}$ can only be on machines located higher up in the same rack, or on the same row or higher for racks to the right. Obviously, $S=\sum_{(i',j')\in U}C_{i',j'}$. However, $S(J\setminus J_{rem})=\sum_{(i',j')\in U}\delta_{i',j'}-\epsilon_{i',j'}$ Writing $C_{i',j'}$ in terms of $\delta_{i',j'}$ and $\epsilon_{i',j'}$, we get

$$C_{i',j'}=(\delta_{i',j'}-\epsilon_{i',j'})\left(\sum_{p=i'}^{m_2}\sum_{q=j'}^{m_1}\frac{1}{{K_1}^{q-j'}{K_2}^{p-i'}}-1\right)+$$
$$\epsilon_{i',j'}\left(\sum_{p=i'}^{m_2}\sum_{q=j'}^{m_1}\frac{1}{{K_1}^{q-j'}\ {K_2}^{p-i'}}-1\right).$$

Therefore,

$$S=\sum_{(i',j')\in U}\left((\delta_{i',j'}-\epsilon_{i',j'})\left(\sum_{p=i'}^{m_2}\sum_{q=j'}^{m_1}\frac{1}{{K_1}^{q-j'}\ {K_2}^{p-i'}}-1\right)\right.$$
$$\left.+\epsilon_{i',j'}\left(\sum_{p=i'}^{m_2}\sum_{q=j'}^{m_1}\frac{1}{{K_1}^{q-j'}\ {K_2}^{p-i'}}-1\right)\right).$$

[2]Since this is for the purpose of analysis, we do not need to define a polynomial procedure to identify the set $J'_{i,j}$ from $J^t_{i,j}$.

Since $K_2 \geq 1 + \frac{K_1}{K_1 - 2}$, it can be verified that $\frac{K_1}{K_1-1} \frac{K_2}{K_2-1} - 1 \leq 1$.
Substituting $S(J \setminus J_{rem})$ and S in $S - S(J \setminus J_{rem})$,

$$\begin{aligned} S(J_{rem}) &\leq \sum_{(i',j') \in U} \left(\epsilon_{i',j'} \left(\sum_{p=i'}^{m_2} \sum_{q=j'}^{m_1} \frac{1}{{K_1}^{q-j'} \ {K_2}^{p-i'}} - 1 \right) \right) \\ &\leq \sum_{(i',j') \in U} \epsilon_{max} \left(\sum_{p=i'}^{m_2} \sum_{q=j'}^{m_1} \frac{1}{{K_1}^{q-j'} \ {K_2}^{p-i'}} - 1 \right) \\ &\leq \epsilon_{max} \end{aligned}$$

where the last inequality follows from the relation between K_1 and K_2. However, $s_{min} > \epsilon_{max}$ and $s_{min} \leq S(J_{rem})$, which is a contradiction. Therefore, $J_{rem} = \emptyset$. Hence we have proved that there exists a packing with the fixed capacity pattern, which, if packed optimally would lose no more than one job per overpacked machine. ■

Proof of Theorem 2:

Proof: Due to assumption on size, we know a machine that loses one object, has at least one other object packed in it. In fact, arbitrarily placing the jobs in the machines with reduced capacity gives a $\frac{\mu}{\mu+1} \geq \frac{1}{2}$ approximation to the maximum cardinality problem.

If the objects have different profits, we would first choose J to be the set of least profit jobs that could be accommodated due to the extra space. This can be done by listing out the least profit jobs from each machine (i, j) in an optimal packing, till the first job when the sum of the sizes of the jobs listed out from (i, j) exceeds the extra space $S_{i,j}$. Note that we need not drop the largest job from $J_{i,j}$ to make the remaining jobs fit in $S_{i,j}$. If we drop any single job from $J_{i,j}$, the remaining would fit in $S_{i,j}$. So let us drop the least profit job from $J_{i,j}$, to find the maximal set of jobs that are completely accommodated in $S_{i,j}$. If the profit of the job dropped from (i, j) is $p_{i,j}$, then the profit of the remaining jobs would be $\geq \mu p_{i,j}$. Hence we get at least half the profit from every machine that loses an object compared to its original contribution to the total profit. Since this is a thermally feasible packing, any optimum packing strategy would therefore give $\geq \frac{1}{2}$ the profit, and using the multiple knapsack packing PTAS we lose at most $O(\epsilon)$ of the profit. ■

B. Horizontal Sibling Model

We now consider the horizontal sibling model. Again there are m_2 racks, and m_1 machines in each rack. Each machine has effective load capacity c.

Algorithm 3 Algorithm for Horizontal Sibling Model

```
1. Fix a capacity pattern $c_{i,j} = c\left(1 - \frac{1}{K_1} + \sum_{\ell=0}^{i-2} \sum_{p=0}^{\min j-1, i-\ell} \left(-\frac{1}{K_2}\right)^{i-\ell-1} \left(-\frac{1}{K_1}\right)^{p} \binom{i+\ell}{p}\right)$.
2. Run the multiple knapsack PTAS on instance $I$ using the machines with modified capacities.
```

Lemma 12: The packing produced by the Algorithm 3 is thermally feasible for the horizontal model.

Proof: This follows from the proof of Claim 3. ■

Let c_{min} be the minimum value of any $c_{i,j}$ computed as above, and $\mu = \lfloor \frac{c_{min}}{\Delta} \rfloor$.

We assume $\mu \geq 1$ as before which is reasonable for data centers. As already stated, μ is much larger in practice.

Theorem 3: When $K_2 \geq 1 + \frac{1}{K_1 - 2}$, Algorithm 3 gives a $\left(\frac{1}{2} - O(\epsilon)\right)$ approximation to the optimal solution for any fixed $\epsilon > 0$ in polynomial time both for the maximum profit and cardinality of integrally assigned jobs.

C. Indirect Sibling Model

We now consider the indirect sibling model. As before, there are m_2 racks, and m_1 machines in each rack. The effective load capacity is c for every machine.

Algorithm 4 Algorithm for Indirect Sibling Model

```
1. Fix the capacities of the machines as $c_{1,1} = c$, $c_{1,j} = c\left(1 - \frac{1}{K_1}\right)$ for $j \geq 2$, $c_{i,1} = c\left(1 - \frac{1}{K_2}\right)$ for $i \geq 2$, and $c_{i,j} = c\left(1 - \frac{1}{K_1} - \frac{1}{K_2}\right)$ for $i \geq 2, j \geq 2$.
2. Run the multiple knapsack PTAS on instance $I$ using the machines with modified capacities.
```

Lemma 13: The packing produced by Algorithm 4 is thermally feasible for the indirect sibling model.

Proof: This follows from the definition of the indirect sibling model. ■

Let $\mu = \lfloor \frac{c\left(1 - \frac{1}{K_1} - \frac{1}{K_2}\right)}{\Delta} \rfloor) \geq 1$.

Theorem 4: When $\sum_{i=0}^{m_2-1} \sum_{j=0}^{m_1-1} \frac{\binom{i+j}{i}}{{K_1}^j \ {K_2}^i} \leq 2$, Algorithm 4 gives a $\left(\frac{1}{2} - O(\epsilon)\right)$ approximation to the optimal solution for a fixed $\epsilon > 0$ in polynomial time to the maximum cardinality and profit problems.

VII. Minimizing Thermal Makespan

In this section we consider the dual problem where we need to assign all jobs and the objective is to minimize the maximum effective load or *thermal makespan* over all the machines. In [25], we had analyzed this problem where fractional assignments are allowed, while here we only allow integral assignments. We analyze both offline and online variants of this problem for a single rack of machines (1D model).

A. Single Rack of Machines

We consider here the 1D model or a single rack of machines. Formally, there are m machines and and n jobs. The objective is to *minimize* $\max_{i \in [1 \ldots m]} EL_i$ while scheduling all the jobs. This problem is obviously NP-hard for general m and n as it is a generalization of the minimum makespan problem.

Note that the maximum effective load may be on a machine that may not have the maximum load assigned among all machines. For example, suppose we have $m = n$ and we have $(m-1)$ unit jobs, and one job of size $1-\epsilon$, and the unit jobs have been assigned to machines $[1 \ldots (m-1)]$ and machine m gets the job of size $1-\epsilon$. The machine m therefore has the lowest load among all machines. However, the effective load on m is $\frac{1-\frac{1}{K^m}}{1-\frac{1}{K}} - \epsilon$. It can be verified that if $\epsilon < \frac{1}{K^{m-1}}$, then machine m has the largest effective load.

1) NP-hardness: The problem of minimizing maximum effective load obviously generalizes the minimum makespan problem, which is strongly NP-hard. However, when the number of jobs is less than the number of machines, the minimum makespan problem is not NP-hard, and any arbitrary arrangement of the jobs will result in the same value of optimum objective function value, equal to the length of the longest job.

However, the minimum effective load problem does not have any such trivial solution for the case of $n \leq m$. Chrobak et al. [7] proved the NP-hardness for a different version of the thermal problem. They proved the NP-hardness of maximizing throughput in thermal aware scheduling for a single machine, where there is a temporal drop off in temperature. Specifically, in their model, the heat contribution of a job h is known. The jobs are unit length. If at the time the job is scheduled, the temperature of the machine is τ, then after the execution of the job, the temperature of the machine is $\frac{\tau+h}{2}$. They have a hard constraint on the maximum temperature that the machine can reach at any time step. This temperature constraint is normalized to 1, and the goal is to schedule as many jobs as possible without violating the temperature constraint at any time step. They show that even when all the jobs have release times 0 and equal deadlines, it is NP-hard to maximize the number of jobs scheduled. Their reduction can be modified to prove the NP-hardness of minimizing the effective load when all jobs have to be scheduled and there is no release time or deadline constraint.

The crucial observation is that the time axis can be interpreted to be the space axis; specifically, every time unit on a single machine can be interpreted as a separate machine. After that minor modifications are required. They show the reduction for an instance where all jobs have release time 0 and deadlines equal to some integer, say m and variable thermal sizes. This can be interpreted in our model as jobs with variable sizes (all jobs available at the beginning), and the number of machines is equal to m which is the deadline of all of the jobs. The model of Chrobak et al. considers the temperature only after every time step, after the execution of a job scheduled on the machine. However, in our model, when a job is assigned to a machine, we do not consider any time lag. Hence, the sizes of the jobs as considered by their reduction, need to be reduced accordingly. Finally, the decision question asked needs to be modified. Their reduction assumes an exponential decay factor of 2. In our case, that can be interpreted as $K = 2$. The reduction is from numerical 3D matching. We do not repeat the construction here since the modifications are minor other than the crucial interpretation of time as space.

The following theorem follows from the work of Chrobak et al. [7].

Theorem 5: The offline problem of minimizing the maximum effective load for the one-dimensional case is NP-hard even when the number of jobs n is equal to the number of machines m.

2) Offline Algorithm: We show that applying Graham's LPT scheduling [14] algorithm gives a $\max\left(\frac{K}{K-1}, \frac{4K-3}{3K-3}\right)$ approximation. This analysis is tight both for $K = 2$ which is the minimum value of K, as well as asymptotically, since for no cross-effects, or $K \to \infty$, it is well known that LPT gives a $\frac{4}{3}$ approximation.

The algorithm is formalized below. The intuitive reason for favoring lower indices is that these machines are closer to the source of cold air. Let L_k be the load on machine k.

Algorithm 5 Algorithm using load as the decision metric

```
1. Sort and order jobs in a list in non-increasing sizes.

2. Assign the next job on the list to machine $k$ such that,
   $L_k \leq L_j \ \forall j \in [1 \ldots m]$ and $L_k < L_p \ \forall p < k$.
```

Theorem 6: Algorithm 5 achieves an approximation ratio of $\max(\frac{K}{K-1}, \left(\frac{4K-3}{3K-3} + \frac{1}{3m}\right))$ for $2 \leq K < 3$ and $\frac{4K-3}{3K-3}$ for $K \geq 3$.

Proof: Let us consider the machine with the maximum effective load after all jobs have been assigned. Let this machine be i. Machine i was assigned $p \geq 1$ jobs. Let the size of the last job assigned to this machine be $s_{i,p}$.

For proving the theorem, we consider two cases separately.

Case 1: $s_{i,p} > \frac{OPT}{3}$

In this case, our solution is $\leq \frac{K}{K-1} OPT$. When $s_{i,p}$ was placed on i, $L_i \leq L_j \ \forall j \in [1 \ldots m]$. All the jobs constituting these loads were larger in size than $s_{i,p}$. So, if $p \geq 3$, obviously $OPT \geq 3 s_{i,p}$. Hence $p = 1$ or $p = 2$.

Suppose $p = 2$. Consider the iteration when $s_{i,2}$ was placed on i. The following claims are true at that iteration.

Claim 9: $L_j = s_{j,1} \geq s_{i,1} \ \forall j \leq i$

Proof: Algorithm 5 places $s_{i,2}$ on the machine with the minimum load, favoring lower indices. Hence, $L_j > s_{i,1} \ \forall j < i$. Further, each such j could have only received a single job so far, since Algorithm 5 considers jobs in non-increasing size order, and assigns them to lowest load machines, favoring lower indices. Hence, jobs larger than $s_{i,1}$ were placed on machines $j < i$. ■

Claim 10: Either $L_k = s_{k,1} = s_{i,1}$ or $L_k \geq 2 s_{i,2} \ \forall k > i$.

Proof: Since Algorithm 5 placed $s_{i,2}$ on i, all machines $L_k \geq s_{i,1} \ \forall k > i$. However, for each such machine, $s_{k,1} \leq s_{i,1}$. If some $k > i$ has only one job, then $s_{k,1} = s_{i,1}$, otherwise,

$s_{i,2}$ would have been placed on k. On the other hand, if k has ≥ 2 jobs, since these jobs were placed earlier than $s_{i,2}$, $L_k \geq 2s_{i,2}$. ■

Lemma 14: If $p = 1$ or $p = 2$, $L_i \leq OPT$.

Proof: It is obvious for $p = 1$. For $p = 2$, we use the above claims to prove the lemma. If $i = m$ then obviously $L_i = L_m \leq OPT$. This follows from Claim 9. Hence, let us assume $i < m$. Let the number of machines $k > i$ with single jobs be ℓ. Therefore, from Claim 9, we know that, in OPT, there are $i + \ell$ jobs of size $\geq s_{i,1}$. Let us call them *big* jobs.

We know from Claim 10 that there are at least $2(m - (i + \ell)) + 1$ jobs of size $\geq s_{i,2}$ that OPT would need to assign. Let us call this set as *small* jobs. Further, we know that in OPT none of the big jobs were paired with any of the small jobs, since that would result in load $\geq L_i > OPT$. Hence, the small jobs in OPT were distributed among $(m - (i + \ell))$ machines. However, that means, that at least one of these machines would get $\geq 3s_{i,2}$ for $(i + \ell) < m$, and hence $OPT \geq 3s_{i,2}$. But that gives a contradiction to the assumption $s_{i,2} > \frac{OPT}{3}$. On the other hand, if $i + \ell = m$, then again from Claim 9, $L_i \leq OPT$. Hence, we have proved $L_i \leq OPT$. ■

From Claim 1, we know $EL_i = L_i + \frac{EL_{i-1}}{K}$. Since by assumption, $EL_i \geq EL_j \forall j$, $EL_{max} \leq L_i + \frac{EL_{max}}{K}$. Applying $L_i \leq OPT$, we get, $EL_{max} \leq \frac{K}{K-1} OPT$.

Case 2: $s_{i,p} \leq \frac{OPT}{3}$

Let the load on i when $s_{i,p}$ was assigned be L_i. Since LPT assigned $s_{i,p}$ to i for job $s_{i,p}$, $L_i \leq L_j \forall j$. Hence, $L_i \leq \frac{\sum_{j=1}^{m} L_j}{m}$. If the total load is L, we have $L_i \leq \frac{L - s_{i,p}}{m}$. The effective load on i is $EL_{max} \leq L_i + s_{i,p} + \frac{EL_{max}}{K}$. Substituting for L_i, we get

$$\begin{aligned} EL_{max} &\leq \frac{K}{K-1}\left(\frac{L - s_{i,p}}{m} + s_{i,p}\right) \\ &= \frac{K}{K-1}\left(\frac{L}{m} + s_{i,p}\left(1 - \frac{1}{m}\right)\right). \end{aligned}$$

We know $s_{i,p} \leq \frac{OPT}{3}$. From Claim 2, the minimum effective load for the one-dimensional system when jobs can be distributed fractionally is $EL = \frac{L}{m - \frac{m-1}{K}}$. Therefore this is a lower bound on OPT. Applying these lower bounds on OPT, we get

$$\begin{aligned} EL_{max} &< \frac{K}{K-1} OPT\left(1 - \frac{1}{K} + \frac{1}{mK} + \frac{1}{3} - \frac{1}{3m}\right) \\ &= OPT\left(1 + \frac{K}{3(K-1)} - \frac{K-3}{3m(K-1)}\right). \end{aligned}$$

Therefore, $EL_{max} \leq OPT\left(\frac{4K-3}{3K-3} + \frac{1}{3m}\right)$ for $2 \leq K < 3$ and $EL_{max} \leq OPT\left(\frac{4K-3}{3K-3}\right)$ for $K \geq 3$. ■

For $K = 2$, $\frac{K}{K-1} = 2 \geq \left(\frac{4K-3}{3K-3} + \frac{1}{3m}\right)$. For this case, there is a tight example. Let instance I have a very large number of machines $m \to \infty$. Let the number of jobs be very large n, however, $n << m$ and all jobs are of unit size. The optimal strategy would space out the jobs with one job on the first machine, one on the last machine, and the rest distributed sparsely such that, the effective load on any machine is $\leq (1 + \epsilon)$, where $\epsilon \to 0$. This will be possible if $n << m$. However, our algorithm will place the jobs on the n consecutive machines, resulting on a maximum effective load on the n^{th} machine which is $\frac{1 - \frac{1}{K^n}}{1 - \frac{1}{K}} \approx \frac{K}{K-1} = 2$ for very large n. Hence the approximation is $\approx 2 - o(\epsilon)$.

For $K > 3$, $\frac{K}{K-1} \leq \frac{4K-3}{3K-3}$. For higher values of K, the approximation tends to $\frac{4}{3}$, which is a tight approximation factor for minimum makespan problem as well. Hence this analysis is tight.

3) Online Algorithm: Here we consider the above problem in an online setting. Specifically, we have m machines, and the jobs arrive in an online fashion. Once a job arrives, we have to assign it to a machine and the decision is irrevocable. The objective is to minimize the maximum effective load or the thermal makespan. We assume the jobs are long-lasting, and hence ignore any temporal effects.

We show Graham's List Scheduling algorithm gives a $\frac{2K-1-\frac{1}{m}}{K-1}$ approximation to the online problem of minimizing thermal makespan. The algorithm is simple. When a job arrives, assign it to the machine with the minimum load.

Theorem 7: Graham's list scheduling algorithm gives a $\frac{2K-1-\frac{1}{m}}{K-1}$ approximation to the online problem of minimizing thermal makespan.

Proof: Let the machine with the largest effective load EL_{max} be i. Let the last job assigned to this machine be s_i and the load on i before assigning s_i be L_i. We know $EL_{max} \leq L_i + s_i + \frac{EL_{max}}{K}$, or, $EL_{max} \leq \frac{K}{K-1}(L_i + s_i)$. Obviously, $s_i \leq OPT$. When s_i was assigned to i, $L_i \leq L_j \forall j \in [1 \dots m]$. Hence, $L_i \leq \frac{\sum_{j \in [1 \dots m]} L_j}{m}$. If the total load to be assigned is L, we have $L_i \leq \frac{L - s_i}{m}$. We know, from Claim 2, $OPT \leq \frac{L}{m - \frac{m-1}{K}}$. Applying the lower bounds on OPT, we get,

$$\begin{aligned} EL_{max} &\leq \frac{K}{K-1} OPT\left(1 - \frac{1}{K} + \frac{1}{mK} + 1 - \frac{1}{m}\right) \\ &\leq OPT\left(\frac{2K - 1 - \frac{1}{m}}{K-1}\right). \end{aligned}$$

■

The analysis is essentially tight, since it is well known that for no cross effects, or $K \to \infty$, list scheduling gives a $2 - \frac{1}{m}$ approximation, which is what we get asymptotically.

VIII. Conclusion

In this work we have considered the thermal scheduling problem in a formal setting for several different models. We have derived approximation algorithms for maximizing profit of jobs scheduled without violating thermal constraints by first solving the fractional problem. We have also shown constant factor algorithms for the problem of minimizing the maximum temperature or equivalently, effective load on any machine when all jobs need to be scheduled. Of course, the derived expressions depend on the precise thermal heating model that one considers; however we have paved the way by showing how to deal with three different models, and one could use a different model and do the analysis in a similar manner.

The thermal scheduling problem opens up several new research directions and interesting questions. These problems are different than what has been considered in the literature so far, in fact, they are generalizations of existing problems and may require new techniques and ideas. Further we have not considered any temporal effects which may give rise to another paradigm of problems.

IX. Acknowledgment:

This work has been partially supported by NSF Grants CCF-1217890 and CCF-0937865.

References

[1] N. Bansal, T. Kimbrel, and K. Pruhs. Dynamic speed scaling to manage energy and temperature. In *Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 520–529, 2004.

[2] N. Bansal and K. Pruhs. Speed scaling to manage temperature. In *Proceedings of the 22nd annual conference on Theoretical Aspects of Computer Science (STACS)*, pages 460–471, 2005.

[3] P. Baptiste. Scheduling unit tasks to minimize the number of idle periods: a polynomial time algorithm for offline dynamic power management. In *Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithms (SODA)*, pages 364–367, 2006.

[4] C. Belady. In the data center and power and cooling costs more than the it equipment it supports. *Electronics Cooling*, 13(1):24, 2007.

[5] C. Chekuri and S. Khanna. A ptas for the multiple knapsack problem. In *Proceedings of the eleventh annual ACM-SIAM Symposium on Discrete algorithms (SODA)*, pages 713–728, 2000.

[6] J. Choi, C. Cher, H. Franke, H. Hamann, A. Weger, and P. Bose. Thermal-aware task scheduling at the system software level. In *Proceedings of the 2007 international symposium on Low power electronics and design*, pages 213–218, 2007.

[7] M. Chrobak, C. Durr, M. Hurand, and Jul Robert. Algorithms for temperature-aware task scheduling in microprocessor systems. *Sustainable Computing: Informatics and Systems*, 1(3):241 – 247, 2011.

[8] A.K. Coskun, J.L. Ayala, D. Atienza, T.S. Rosing, and Y. Leblebici. Dynamic thermal management in 3d multicore architectures. In *Proceedings of the Conference on Design, Automation and Test in Europe*, pages 1410–1415, 2009.

[9] D. Economou, S. Rivoire, and C. Kozyrakis. Full-system power analysis and modeling for server environments. In *Workshop on Modeling Benchmarking and Simulation*, pages 70–77, 2006.

[10] N. Fisher, J.J. Chen, S. Wang, and L. Thiele. Thermal-aware global real-time scheduling on multicore systems. In *Real-Time and Embedded Technology and Applications Symposium, 2009. RTAS 2009. 15th IEEE*, pages 131–140, 2009.

[11] Y. Ge, P. Malani, and Q. Qiu. Distributed task migration for thermal management in many-core systems. In *Proceedings of the 47th Design Automation Conference*, pages 579–584, 2010.

[12] M. Gomaa, M.D. Powell, and T.N. Vijaykumar. Heat-and-run: leveraging smt and cmp to manage power density through the operating system. In *ACM SIGARCH Computer Architecture News*, volume 32, pages 260–270, 2004.

[13] R.L. Graham. Bounds for certain multiprocessing anomalies. *Bell System Technical Journal 45*, 45(9):1563–1581, 1966.

[14] R.L. Graham. Bounds on multiprocessing timing anomalies. *SIAM Journal of Applied Mathematics*, 17(2):416–429, 1969.

[15] M.K. Herrlin and C. Belady. Gravity-assisted air mixing in data centers and how it affects the rack cooling effectiveness. In *The Tenth Intersociety Conference on Thermal and Thermomechanical Phenomena in Electronics Systems (ITHERM)*, pages 434–438, 2006.

[16] S. Irani, S. Shukla, and R. Gupta. Algorithms for power savings. In *Proceedings of the fourteenth annual ACM-SIAM symposium on Discrete algorithms (SODA)*, pages 37–46, 2003.

[17] K. Jansen. Parameterized approximation scheme for the multiple knapsack problem. *SIAM Journal on Computing*, 39(4):1392–1412, 2009.

[18] E. Kursun, C.Y. Cher, A. Buyuktosunoglu, and P. Bose. Investigating the effects of task scheduling on thermal behavior. In *Third Workshop on Temperature-Aware Computer Systems (TACS'06)*, 2006.

[19] J. Li, M. Qiu, J. Hu, and E.H.M. Sha. Thermal-aware rotation scheduling for 3d multi-core with timing constraint. In *2010 IEEE Workshop on Signal Processing Systems (SIPS)*, pages 323–326, 2010.

[20] S. Liu and M. Qiu. Thermal-aware scheduling for peak temperature reduction with stochastic workloads. In *16th IEEE Real-Time and Embedded Technology and Applications Symposium, Stockholm: WIP*, pages 59–62, 2010.

[21] S. Liu, J. Zhang, Q. Wu, and Q. Qiu. Thermal-aware job allocation and scheduling for three dimensional chip multiprocessor. In *2010 11th International Symposium on Quality Electronic Design (ISQED)*, pages 390–398, 2010.

[22] G. Loh, Y. Xie, and B. Black. Processor design in 3d die-stacking technologies. In *IEEE Micro*, pages 31–48, 2007.

[23] J. Moore, J. Chase P. Ranganathan, and R. Sharma. Making scheduling cool: Temperature-aware workload placement in data centers. In *Proceedings of the annual conference on USENIX Annual Technical Conference*, pages 61–75, 2005.

[24] K. Mukherjee, S. Khuller, and A. Deshpande. Algorithms for the thermal scheduling problem. http://www.cs.umd.edu/~samir/grant/thermal-full.pdf, 2012.

[25] K. Mukherjee, S. Khuller, and A. Deshpande. Saving on cooling: The thermal scheduling problem. In *Proceedings of the 12th ACM SIGMETRICS/PERFORMANCE joint international conference on Measurement and Modeling of Computer Systems*, pages 397–398, 2012.

[26] T. Mukherjee, A. Banerjee, G. Varsamopoulos, S.K.S. Gupta, and S. Rungta. Spatio-temporal thermal-aware job scheduling to minimize energy consumption in virtualized heterogeneous data centers. *Computer Networks*, 53(17):2888–2904, 2009.

[27] E. Pakbaznia and M. Pedram. Minimizing data center cooling and server power costs. In *Proceedings of the 14th ACM/IEEE International Symposium on Low Power Electronics and Design*, pages 145–150, 2009.

[28] R. Schmidt and E. Cruz. Raised floor computer data center: effect on rack inlet temperatures of chilled air exiting both the hot and cold aisles. In *The Eighth Intersociety Conference on Thermal and Thermomechanical Phenomena in Electronic Systems*, pages 580–594, 2002.

[29] B. Shi and A. Srivastava. Thermal and power-aware task scheduling for hadoop based storage centric datacenters. In *greencomp: International Conference on Green Computing*, pages 73–83, 2010.

[30] Q. Tang, S. Gupta, and G. Varsamopoulos. Thermal-aware task scheduling for data centers through minimizing heat recirculation. In *IEEE International Conference on Cluster Computing, 2007*, pages 129–138, 2007.

[31] Q. Tang, S.K.S Gupta, D. Stanzione, and P. Cayton. Thermal-aware task scheduling to minimize energy usage of blade server based datacenters. In *2nd IEEE International Symposium on Dependable, Autonomic and Secure Computing*, pages 195–202, 2006.

[32] Q. Tang, S.K.S. Gupta, and G. Varsamopoulos. Energy-efficient thermal-aware task scheduling for homogeneous high-performance computing data centers: A cyber-physical approach. *IEEE Transactions on Parallel and Distributed Systems*, pages 1458–1472, 2008.

[33] G. Varsamopoulos, A. Banerjee, and S.K.S Gupta. Energy efficiency of thermal-aware job scheduling algorithms under various cooling models. *Contemporary Computing*, pages 568–580, 2009.

[34] F. Yao, A. Demers, and S. Shenker. A scheduling model for reduced cpu energy. In *Proceedings of the 36th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 374–382, 1995.

[35] S. Zhang and K.S. Chatha. Approximation algorithm for the temperature-aware scheduling problem. In *Proceedings of the 2007 IEEE/ACM international conference on Computer-aided design*, pages 281–288, 2007.

[36] X. Zhou, J. Yang, Y. Zhang Y. Xu, and J. Zhao. Thermal-aware task scheduling for 3d multicore processors. *IEEE Transactions on Parallel and Distributed Systems*, pages 60–71, 2010.

[37] C. Zhu, Z. Gu, Li Shang, R.P. Dick, and R. Joseph. Three-dimensional chip-multiprocessor run-time thermal management. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pages 1479–1492, 2008.