Matching with sizes (or scheduling with processing set restrictions)

Péter Biró and Eric McDermid

Department of Computing Science
University of Glasgow
Glasgow G12 8QQ
UK

Technical Report
TR-2010-307
January 2010

Matching with sizes (or scheduling with processing set restrictions)[*]

Péter Biró[†] and Eric McDermid

Department of Computing Science, University of Glasgow, Glasgow G12 8QQ, UK
Email: {pbiro,mcdermid}@dcs.gla.ac.uk.

Abstract

Matching problems on bipartite graphs where the entities on one side may have different sizes are intimately related to scheduling problems with processing set restrictions. We survey the close relationship between these two problems, and give new approximation algorithms for the (NP-hard) variations of the problems in which the sizes of the jobs are restricted. Specifically, we give an approximation algorithm with an additive error of one when the sizes of the jobs are either 1 or 2, and generalise this to an approximation algorithm with an additive error of $2^k - 1$ for the case where each job has a size taken from the set $\{1, 2, 4, \dots, 2^k\}$ (for any constant integer k). We show that the above two problems become polynomial time solvable if the processing sets are nested.

Keywords: couples, Hospitals/Residents problem, scheduling, processing set restrictions, computational complexity, approximation algorithms

1 Introduction

In this paper we investigate bipartite matching problems where the entities on one side may have different sizes. This type of problem can arise in at least two interesting problem domains. The first is the realm of matching markets in which *couples* are present, where a couple is an inseparable pair of agents. The second significant application area is the realm of scheduling problems in which jobs of different lengths need to be allocated to machines for which they are eligible.

The US Navy assignment process involves hundreds of thousands of sailors to be assigned, commands seeking sailors and detailers who advise both sailors and commands. According to recent studies [24, 19, 25] the three most important requirements to be satisfied, in order to achieve an optimal assignment of sailors to billets, are the following:

Size of the matching: All the sailors should be matched, and there are certain critical billets that cannot go unfilled. Stability: Sailors should not be forced into assignments that they have neither asked for nor desire. Moreover, there should be no sailor who would prefer to be matched to a particular billet if the commanding officer responsible for that billet would also request this sailor. Presence of couples: The need to assign married couples to the same location.

If we relax the stability condition by requiring only that a sailor should not be matched to a billet which is not acceptable for her/him, then we get a setting that fits our basic

[*]This work was supported by EPSRC grant EP/E011993/1
[†]This work was supported by OTKA grant K69027.

problem, *Matching with Couples*, that is the problem of finding a maximum size matching in a bipartite graph where each vertex on one side has size 1 or 2 and each vertex on the other side has an integer capacity. We will define this problem more precisely in Section 2.

The presence of couples is quite natural in other two-sided job markets as well. In fact, in some matching schemes, such as the National Resident Matching Program [18] and the Scottish Foundation Allocation Scheme [22] (schemes that allocate junior doctors to hospitals in the US and in Scotland, respectively), couples are allowed to apply for pairs of positions. The reason for this possibility is that a couple may be prepared to accept a pair of positions only if these are in the same hospital or geographically close, for example. The detailed description of the mechanism underlying the current matching scheme of NRMP can be found in the paper of Roth [21], and some remaining problems caused by the presence of couples were studied by Klaus *et al.* [11].

In the above two-sided markets, where the participants on both sides have preferences and they may reach a private agreement outside the matching scheme (in contrast with the military services), the stability of the solution is considered as a first priority in most existing matching schemes. If couples are not present in the market, we have the classical *Hospitals/Residents* (or *College Admissions*) problem, for which a stable matching can always be found in linear time by the algorithm of Gale and Shapley [4]. However Ronn [20] showed that the problem of deciding whether a stable matching exists, given a Hospitals/Residents problem with couples, is NP-complete. Manlove and McDermid [14] proved that the hardness result still holds even if each couple accepts an assignment only if they are both allocated to the same hospital.

In other applications, such as the allocation of students to dormitories or the assignment of papers to reviewers [5], the preferences may be on one side only. Here also we can imagine that the entities to be matched have different sizes, since a couple (or an even larger group) may want to be allocated to the same place, and the organisers of a conference may call for different kinds of papers (e.g. short and regular). In both of the applications mentioned it is natural to seek a complete (or maximum size) solution, such that the *loads* of the dormitories or the reviewers are *balanced*, leading again to our main problem of Matching with Couples. Or we obtain the more general problem *Matching with Sizes* in case the sizes are positive numbers (not only from the set $\{1, 2\}$). We note that if the organisers of the matching scheme take the preferences into account then as a second priority they may try to find a maximum weight or a *rank-maximal* matching (among the maximum cardinality load balanced matchings). Further details of the latter appear in [9].

Finally, in many assignment problems there may be no preferences at all. A major area of applications is scheduling where we need to allocate, say, jobs of different lengths to machines. In this paper we only take into account the eligibility requirements, that is for each job we suppose that there is a set of machines that are capable of processing that job. In this case we get the *parallel machine* (or multiprocessor) *scheduling problem with processing set* (or machine eligibility, or job assignment) *restrictions*. The problem that we focus on is to minimise the makespan (i.e., the time at which the processing of the last job is finished). We will define the problem more precisely in Section 2. Meanwhile we refer to a survey [13] on scheduling with processing set restrictions and two papers [6, 7] that contain results closely related to those presented in this paper (we will specify these in Section 2).

The contribution of this paper is the following. In Section 2, we survey the correspondence between the problem Matching with Sizes and the parallel machine scheduling problem with processing set. In Section 3, we show that the problem Matching with

Couples is NP-complete even if each capacity is equal to 2. In Section 4, we give an approximation algorithm for the problem Matching with Couples, which finds a solution in $O(ne)$ time (n is the number of jobs and e is the number of edges in the graph) with an additive error 1. Also, we extend our algorithm for the more general problem where the lengths of the jobs are from the set $\{1, 2, 4, \ldots, 2^k\}$ for some constant integer k, and we show that in this case a solution with an additive error $2^k - 1$ is guaranteed to be found in $O(2^k ne)$ time. Finally in Section 5, we give polynomial time algorithms to solve the above two problems in the case when the processing sets form a nested set system.

2 Problem statements and related results

In this section we first define the problem Matching with Sizes and the parallel machine scheduling problem with processing set restrictions. Next, we describe the correspondence between these problems and we survey the related literature.

Matching with Sizes (MS):
We are given a bipartite graph $G(U \cup V, E)$,where $U = \{u_1, u_2, \ldots, u_n\}$, $V = \{v_1, v_2, \ldots, v_m\}$ and $e = |E(G)|$. Suppose that each $u_i \in U$ has a *size* $s(u_i) \in \mathbb{R}^+$ and each $v_j \in V$ has a *capacity* $c(v_j) \in \mathbb{R}^+$. A matching M is a set of edges such that the capacity constraints are satisfied, i.e., $\sum_{u_i:\{u_i,v_j\}\in M} s(u_i) \leq c(v_j)$ for each $v_j \in V$. The *size of a matching* M is $s(M) = \sum_{\{u_i,v_j\}\in M} s(u_i)$. The problem is to find a matching of maximum size. The decision problem related to an instance of MS, denoted by D-MS, is to decide whether there exists a *feasible* matching M, that is a matching which covers U.

Furthermore, given an instance I of MS and a matching M, let $l_M(v_j) = \sum_{u_i:\{u_i,v_j\}\in M} s(u_i)$ be the *load* of v_j. The *load vector* of M is defined as $\bar{l}_M = (l_M(v_1), l_M(v_2), \ldots, l_M(v_m))$. The quality of the feasible matching may be measured by the L_p norm of the load vector, that is

$$||\bar{l}_M||_p = \left(\sum_{v_j \in V} |l_M(v_j)|^p \right)^{1/p}$$

Note that $||\bar{l}_M||_1$ is equal to the size of M, and $||\bar{l}_M||_\infty$ is the maximum load in M. Normally we want to maximise the former and minimise the latter.

Scheduling jobs to machines with processing set restrictions:
We have a set of jobs $\mathcal{J} = \{J_1, J_2, \ldots, J_n\}$ and a set of parallel machines $\mathcal{M} = \{M_1, M_2, \ldots, M_m\}$. Each job J_j has processing time p_j and a set of machines $\mathcal{M}_j \subseteq \mathcal{M}$ to which it can be assigned, the *processing set* of J_j. (So we suppose that the machines are *identical* in the sense that a job J_j requires the same amount of time on each machine which is eligible for J_j.) In a *schedule* with the above processing set restrictions, each job J_j is assigned to one of the machines in $\mathcal{M}_j$. In this paper we will focus on the problem of finding a schedule such that the *makespan* C_{max} (i.e., the time when the last job is completed) is minimised. The standard notation used for this problem in the scheduling literature is $P|\mathcal{M}_j|C_{max}$ (P stands for identical machines, $\mathcal{M}_j$ indicates that we have processing set restrictions and C_{max} denotes that the task is to minimise the makespan.)

The relation between the above two problems is the following. Given an instance I of MS with infinite capacities, the problem of finding a maximum size matching (i.e. a matching that covers U) with minimum L_∞-norm is equivalent to $P|\mathcal{M}_j|C_{max}$ for an instance where $\mathcal{J} = U$, $\mathcal{M} = V$ and the set of machines $\mathcal{M}_j \subseteq \mathcal{M}$ suitable for a job

$J_j \in \mathcal{J}$ corresponds to the set of neighbours of $u_j \in U$ in G. Furthermore, D-MS for an instance where each capacity is equal to a constant T is equivalent to the decision problem related to the corresponding instance of $P|\mathcal{M}_j|C_{max}$ with makespan T.

Given the close correspondence, henceforth we refer to the sets of U and V in an instance of MS as jobs and machines, respectively, and we will use the term length when referring to the size of a job. By adding restrictions on the lengths of the jobs we get the following special cases.

Tutorial Allocation (TA):
This is a special case of MS where $s(u_i) = 1$ for every $u_i \in U$ and $c(v_j) \in \mathbb{Z}^+$. The terminology of TA was introduced by Abraham [1]. This problem is closely related to the scheduling problem with unit-length jobs, denoted by $P|\mathcal{M}_j, p_j = 1|C_{max}$, studied in, e.g., [2] and [7]. Finally, the problem of TA with infinite capacities is also known as the *semi-matching* problem [8].

Alon *et al.* [2] gave an $O(n^3 e)$ algorithm to solve $P|\mathcal{M}_j, p_j = 1|C_{max}$. Moreover, they showed that the output of their algorithm is a so-called *strongly-optimal assignment*, which is a maximum matching that minimises the L_p-norm of the load vector for every $p > 1$. Abraham [1] gave a similar algorithm with an improved $O(ne)$ running time that also finds such a maximum matching with minimum L_p-norm for every $p > 1$ for TA (even if we have positive integer capacities). He called this solution a *balanced matching.* Independently, Harvey *et al.* [8] investigated the same problem in the context of semi-matchings (which is equivalent to $P|\mathcal{M}_j, p_j = 1|$ or TA with infinite capacities). They also gave an $O(ne)$ time algorithm to find a so-called *optimal semi-matching* along the same lines.

Matching with Couples (MC):
This problem is obtained if $s(u_i) \in \{1, 2\}$ for every $u_i \in U$ and $c(v_j) \in \mathbb{Z}^+$. The corresponding scheduling problem is denoted by $P|\mathcal{M}_j, p_j = \{1, 2\}|C_{max}$. Both decision problems are NP-complete, even if $c(v_j) = 2$ for all $v_j \in V$ [3] or when we have to decide whether $C_{max} \leq 2$ or $C_{max} \geq 3$ for the related instance of $P|\mathcal{M}_j, p_j = \{1, 2\}|C_{max}$ [6]. Note that the latter result implies that we cannot give a polynomial-time approximation for $P|\mathcal{M}_j, p_j = \{1, 2\}|C_{max}$ with a factor better than 1.5 (unless P=NP). Both proofs are based on a reduction similar to the one used by Lenstra et al. [12] for a closely related scheduling problem. In Section 3, we include the proof from [3] to show that MC is NP-complete.

As a closely related result, it is worth mentioning that McCormick *et al.* [16] showed that the problem of multiprocessor scheduling with two job lengths and no eligibility constraints, which includes $P|p_j = \{1, 2\}|C_{max}$ as a special case, is polynomial-time solvable. Finally we remark that Marx and Schlotter [15] studied the parameterised complexity of MC. They gave a randomised FPT algorithm by considering the number of couples as the parameter. (This is in contrast with their result on the Hospitals/Residents problem with couples, where they showed that the problem of finding a stable solution with the same parameter is W[1]-hard.)

3 MC is NP-complete

Theorem 1. MC *is NP-complete, even if every machine has capacity 2.*

Proof. We reduce from the 3-dimensional matching problem (3DM). Here, we are given a set $\mathcal{F} \subseteq M \times W \times D$ of ordered triples (families), where M, W and D (men, women and dogs) are disjoint sets of cardinality n. The problem is to decide whether there is a perfect matching F (i.e. a set $F \subseteq \mathcal{F}$ of disjoint families of cardinality n). 3DM is known to be NP-complete [10].

Given an instance I of 3DM as described above, we create an instance I' of MC on a graph $G(U,V)$ as follows. Suppose that $U = U^0 \cup U^1$, where U^0 is the set of unit-length jobs and U^1 is the set of jobs of length two, and we assume that each machine has capacity two. Let $u_k^1 \in U^1$ correspond to a dog d_k for every $d_k \in D$. For every (man, woman) pair $(m_i, w_j) \in M \times W$, we create a machine $v_{i,j}$ and we add the edge $\{u_k^1, v_{i,j}\}$ to G whenever $(m_i, w_j, d_k) \in \mathcal{F}$. Furthermore, we create two unit-length jobs $u_{i,j}^m$ and $u_{i,j}^w$ for each machine $v_{i,j}$ and we add both the edges $\{u_{i,j}^m, v_{i,j}\}$ and $\{u_{i,j}^w, v_{i,j}\}$ to G. Finally, we create some other machines v_i^m and v_j^w together with some corresponding unit-length jobs u_i^m and u_j^w. First, we add the edges $\{v_i^m, u_i^m\}$ to G for every i and $\{v_j^w, u_j^w\}$ for every j, and then we also add the edges $\{v_i^m, u_{i,j}^m\}$ to G for every j and $\{v_j^w, u_{i,j}^w\}$ for every i.

Suppose first that we have a matching F in I of cardinality n. We create a matching M in G that covers U as follows. As every u_i^m and u_j^w has degree one, every edge $\{u_i^m, v_i^m\}$ and $\{u_j^w, v_j^w\}$ should obviously belong to M. Let $\{u_k^1, v_{i,j}\}$, $\{v_i^m, u_{i,j}^m\}$ and $\{v_j^w, u_{i,j}^w\}$ be in M if $(m_i, w_j, d_k) \in F$, and $\{v_{i,j}, u_{i,j}^m\}, \{v_{i,j}, u_{i,j}^w\} \in M$ if $(m_i, w_j, d_k) \notin F$ for any $d_k \in D$. Here, M is matching in G that covers U.

Suppose now, that we have a matching M in G that covers U. We note again, that every edge $\{u_i^m, v_i^m\}$ and $\{u_j^w, v_j^w\}$ should obviously belong to M. Let (m_i, w_j, d_k) be in the matching F of I if $\{u_k^1, v_{i,j}\} \in M$. Here, $|F| = n$, since u_k^1 is covered by M, thus each d_k is in a family of F. To see that F is a disjoint set of families it is enough to observe that $\{u_k^1, v_{i,j}\} \in M$ implies $\{v_i^m, u_{i,j}^m\}, \{v_j^w, u_{i,j}^w\} \in M$, so v_i^m (and v_j^w) cannot have another unit-length job $u_{i,j'}^m$ (or $u_{i',j}^w$) in M, thus m_i (and w_j) can belong to at most one family in F. □

4 Approximation algorithms

In this section we give approximation algorithms for MC and MS.

Matching with couples

Theorem 2. *Let I be an instance of D-MC on graph $G(U \cup V, E)$ with capacities c. If there is a feasible matching M of I then we can find a feasible matching M' for the instance I', where $c'(v_j) = c(v_j) + 1$ for each $v_j \in V$, in $O(ne)$ time.*

Proof. Suppose that $U = U^0 \cup U^1$ where $s(u_i^0) = 1$ for each $u_i^0 \in U^0$ and $s(u_i^1) = 2$ for each $u_i^1 \in U^1$. We start with a relaxed problem. Let I^r be an instance of TA such that for each $u_i^1 \in U^1$ in G we create two copies $\hat{u}_i$ and $\check{u}_i$ in the graph G^r of I^r both of length 1 having the same processing set as u_i^1 had. If there is a feasible matching M for I then there must be a feasible matching M^r for I^r too. We can find such a matching in $O(ne)$ time with the algorithm of Abraham [1].

Let $G_B(V, E_B)$ be an undirected graph such that if $M^r(\hat{u}_i) = v_j \neq v_k = M^r(\check{u}_i)$ then we create an edge $\{v_j, v_k\}$ in E_B. Nash-Williams [17] showed that there exists a so-called *balanced orientation* of any undirected graph, that is a directed graph $D_B(V, \overline{E_B})$ where the in-degree of any vertex $v \in V$ is at most the out-degree of v plus 1. We can find such a balanced orientation in $O(e)$ time.

Now we construct a feasible matching M' for I' in the following way. Let $M'(u_i^1) = v_k$ if $M^r(\hat{u}_i) = v_j \neq v_k = M^r(\check{u}_i)$ and $(v_j, v_k) \in D$. Further, $M'(u_i^1) = M(u_i^1)$ if $M^r(\hat{u}_i) = M^r(\check{u}_i)$, and $M'(u_i^0) = M(u_i^0)$ for each $u_i^0 \in U^0$. It is immediate that M' is a feasible matching for capacities c'.

□

Corollary 1. *We can approximate $P|\mathcal{M}_j, p_j = \{1,2\}|C_{max}$ with an additive error 1, (giving a polynomial-time approximation with ratio $\frac{3}{2}$.)*

Proof. If the optimal schedule corresponding to the matching M has makespan C_{max} for the related instance of $P|\mathcal{M}_j, p_j = \{1,2\}|C_{max}$ then there must exist a schedule M^r in the relaxed instance of $P|\mathcal{M}_j, p_j = 1|C_{max}$ (unit-length scheduling problem) with makespan at most C_{max}. Therefore the schedule corresponding to matching M' has makespan at most $C_{max} + 1$ for the original problem. □

We note that Glass and Kellerer [6] gave an algorithm resulting in a 1.5-approximation for $P|\mathcal{M}_j, p_j = \{1,2\}|C_{max}$, however, our algorithm is simpler than theirs, and has a faster running time.

Matching with sizes

Theorem 3. *Let I be an instance of* D-MS *on graph $G(U \cup V, E)$ with integer capacities c and sizes in the set $\{1, 2, 4, \ldots, 2^k\}$ for any constant integer k. If there is a feasible matching M of I then we can find a feasible matching M' for the instance I' where $c'(v_j) = c(v_j) + (2^k - 1)$ for each $v_j \in V$ in $O(2^k ne)$ time.*

Proof. Let $U = U^0 \cup U^1 \cup U^2 \cup \ldots \cup U^k$ where $s(u_i) = 2^t$ for each $u_i \in U^t$, $t = \{0, 1, \ldots, k\}$. We create $k+1$ new instances of MS starting from $t = k$ down to $t = 0$ in the following way. Let I^k be essentially the same as I with the following minor differences. The graph $G^k(U_k, V)$ of I^k is the same as G, only $U = U^0 \cup U^1 \cup U^2 \cup \ldots \cup U^k$ is renamed as $U_k = U_k^0 \cup U_k^1 \cup U_k^2 \cup \ldots \cup U_k^k$. The capacities are increased by $2^k - 1$, so $c^k(v_j) = c(v_j) + 2^k - 1$ for each $v_j \in V$.

For every t, $t = \{k-1, \ldots, 1, 0\}$, let I^t be an instance of MS with sizes $\{1, 2, 4, \ldots, 2^t\}$ as follows. Let the graph of I^t be $G^t(U_t, V)$ where $U_t = U_t^0 \cup U_t^1 \cup U_t^2 \cup \ldots \cup U_t^t$. Let $U_t^l = U_{t+1}^l$ for every index $l < t$ and $U_t^t = U_{t+1}^t \cup \hat{U}_{t+1}^{t+1} \cup \check{U}_{t+1}^{t+1}$ where $\hat{U}_{t+1}^{t+1}$ and $\check{U}_{t+1}^{t+1}$ are two copies of U_{t+1}^{t+1}, i.e., each job $u_i \in U_{t+1}^{t+1}$ of length 2^{t+1} is replaced by two copies $\hat{u}_i \in \hat{U}_{t+1}^{t+1}$ and $\check{u}_i \in \check{U}_{t+1}^{t+1}$ both of length 2^t, keeping the same processing set. Finally, let each capacity be increased by $2^t - 1$, so $c^t(v_j) = c(v_j) + 2^t - 1$ for each $v_j \in V$.

By the above construction, for $t = 0$ we get an instance I^0 of TA where the number of (unit-length) jobs is at most $2^k n$ (the number of machines remains the same). It is obvious that if there is a feasible matching for I then there must be a feasible matching for I^0 too. We solve this problem with the algorithm of Abraham, thereby obtaining matching M^0. After this we will create M' by repeatedly reuniting the separated groups as follows. Starting with $t = 0$, given a feasible matching M^t for instance I^t we create an undirected graph $G_B^t(V, E_B^t)$ such that $\{v_j, v_k\}$ is in E_B^t if $M^t(\hat{u}_i) = v_j \neq v_k = M^t(\check{u}_i)$ where $\hat{u}_i \in \hat{U}_{t+1}^{t+1}$ and $\check{u}_i \in \check{U}_{t+1}^{t+1}$. We orient the edges of G_B^t in a balanced way, getting directed graph $D^t(V, \overline{E_B^t})$, and then we construct M^{t+1} from M^t as follows. Let $M^{t+1}(u_i) = M^t(u_i)$ for every $u_i \in U_{t+1} \backslash U_{t+1}^{t+1}$. If $u_i \in U_{t+1}^{t+1}$ and $M^t(\hat{u}_i) = M^t(\check{u}_i)$ then also let $M^{t+1}(u_i) = M^t(u_i)$. Finally, if $u_i \in U_{t+1}^{t+1}$ and $M^t(\hat{u}_i) = v_j \neq v_k = M^t(\check{u}_i)$ with $(v_j, v_k) \in D^t(V, \overline{E_B^t})$ then let $M^{t+1}(u_i) = M^t(\check{u}_i)$.

If M^t is a feasible matching for I^t then M^{t+1} is a feasible matching for I^{t+1} since $c^{t+1}(v_j) = c^t(v_j) + 2^t$ for each $v_j \in V$ and the load of a machine may be increased by at most 2^t in M^{t+1}.

At the end of this process we get a feasible matching $M' = M^k$ for instance $I' = I^k$ where $c'(v_j) = c(v_j) + 1 + 2 + 4 + \ldots + 2^{k-1} = c(v_j) + 2^k - 1$.

□

Corollary 2. *We can approximate $P|\mathcal{M}_j, p_j = \{1, 2, 4, \ldots, 2^k\}|C_{max}$ for any positive constant k with an additive error $2^k - 1$, (giving a polynomial-time approximation with ratio $2 - \frac{1}{2^k}$.)*

Proof. If an optimal schedule corresponding to a feasible matching M has makespan C_{max} for the related instance of $P|\mathcal{M}_j, p_j = \{1, 2, 4, \ldots, 2^k\}|C_{max}$ then there must exist a schedule for the relaxed unit-length scheduling problem with makespan at most C_{max}, therefore M' corresponds to a schedule with makespan at most $C_{max} + 2^k - 1$ for the original problem. □

Here, the best known approximation-ratio in the scheduling context has been 2, proved by Lenstra *et al.* [12] (for a more general setting). This was improved by Shchepin and Vakhania [23] to $2 - \frac{1}{m}$ (where m is the number of machines). But the ratio achieved by our algorithm ($2 - \frac{1}{2^k}$) is better for any constant k. For example, for $k = 2$ (i.e., for job lengths $\{1, 2, 4\}$) we have a 1.75-approximation. Also, our performance ratio is better for any $k \leq \log_2 m$.

5 Nested processing sets

Given an instance I of MS on a graph $G(U, V)$, let $N(u_i)$ denote the processing set of a job u_i, i.e., the neighbours of u_i in G. The processing sets form a *nested set system* if, for any two jobs u_i and u_i', $N(u_i) \cap N(u_i') \neq \emptyset$ implies either $N(u_i) \subseteq N(u_i')$ or $N(u_i) \supseteq N(u_i')$. Henceforth we will use the notations MS-N and MC-N for MS and MC with nested set systems, respectively, and D-MS-N and D-MC-N for the corresponding decision problems. In the scheduling literature the standard notation for nested set systems is PN instead of P at the beginning of the abbreviations. The case of $PN|\mathcal{M}_j|C_{max}$ has been studied by Glass and Kellerer [6] who gave an approximation algorithm with performance ratio $2 - \frac{1}{m}$ (where m is the number of machines). For further descriptions of and motivation for nested set systems in scheduling problems see [13] and [6].

Theorem 4. D-MC-N *is solvable in $O(e)$ time, where e is the number of eligible job-machine pairs in I.*

Proof. Suppose that we are given an instance I of D-MC-N on a graph $G(U \cup V, E)$, where $U = U^0 \cup U^1$, $U^0 = \{u_1^0, \ldots u_{n_1}^0\}$ are the jobs of length 1, $U^1 = \{u_1^1, \ldots u_{n_2}^1\}$ are the jobs of length 2, and $V = \{v_1, \ldots v_m\}$ is the set of machines with capacities $c : V \to \mathbb{Z}^+$. Suppose also that the processing sets $\mathcal{V} = \{V_1, V_2, \ldots V_l\}$ are ordered in such a way that $V_i \not\supseteq V_j$ for any $i < j$. Note that $l \leq 2m - 1$. Let I_p denote the subinstance of I which is the restriction of I to the set of machines V_p and to the jobs $u_i \in U$ such that $N(u_i) \subseteq V_p$ (i.e., the jobs which are eligible for some subset of machines in V_p).

For any matching M, let $r_M(v_j)$ denote the remaining capacity of machine v_j, i.e., $r_M(v_j) = c(v_j) - \sum_{u_i : \{u_i, v_j\} \in M} s(u_i)$. We refer to $2\left\lfloor \frac{r_M(v_j)}{2} \right\rfloor$ as the *remaining even capacity* of machine v_j. We say that feasible matching M for I_p is *economical* if $\sum_{v_j \in V_p} 2\left\lfloor \frac{r_M(v_j)}{2} \right\rfloor$ (i.e., the *total remaining even capacity* of the set of machines V_p with respect to M) is maximal.

In the algorithm we take the processing sets according to the above defined order of $\mathcal{V}$ and we consider the corresponding jobs one by one (in any order). We start with the empty matching, $M = \emptyset$, and we enlarge M whenever we allocate a job to a machine. We use the following greedy rule to choose an eligible machine for the job considered. For each job $u_i^1 \in U^1$ we allocate u_i^1 to any machine v_j in $N(u_i^1)$ for which $r_M(v_j) \geq 2$ if such a machine exists (and in case $r_M(v_j) < 2$ for every machine $v_j \in N(u_i^1)$ we report that no feasible solution exists). For each job $u_i^0 \in U^0$ either we allocate u_i^0 to any machine v_j in $N(u_i^0)$ for which $r_M(v_j) \geq 1$ and $r_M(v_j)$ is odd, if there exists such a machine, or (in case $r_M(v_j)$ is even for every machine $v_j \in N(u_i^0)$) we allocate u_i^0 to any machine v_j in $N(u_i^0)$ such that $r_M(v_j) \geq 1$. Finally we report that no feasible solution exists in case $r_M(v_j) = 0$ for every machine $v_j \in N(u_i^0)$.

We claim that if I is solvable then our algorithm finds a feasible matching M. Moreover, we will show that M is economical for I_p for every $1 \leq p \leq l$ and for V as well. We prove this by induction on p. Let M_{p-1} and M_p denote the matchings obtained by the algorithm before and after considering set V_p, respectively.

It is easy to verify the assumption for each set V_p which has no subset in $\mathcal{V}$. In this case the subgraph of G induced by I_p is complete and no job in I_p is allocated in M_{p-1}. Therefore the subinstance I_p is solvable if and only if the total length of the jobs is less than or equal to the total capacity of the machines, and the total length of the jobs of length 2 is less than or equal to the total even capacity (i.e., $\sum_{v_j \in V_p} 2 \left\lfloor \frac{c(v_j)}{2} \right\rfloor$). It is straightforward to verify that if I_p is solvable then our algorithm will produce a feasible matching M_p that is economical for I_p (and the final matching M remains economical for I_p, since we do not reallocate these jobs later on).

Let us now consider a set V_q that is a superset of some set in $\mathcal{V}$ (that we have already considered in our algorithm). We call a subset $V_p \subset V_q$ a *child* of V_q if there exists no $V_{p'}$ such that $V_p \subset V_{p'} \subset V_q$. According to our inductive assumption, M_{p-1} is economical for each child of V_q. Let us consider I_q without the new jobs, i.e., without the jobs $\{u_i : N(u_i) = V_q\}$. For this subinstance M_{q-1} is economical, since M_{q-1} is economical for each child of V_q and the processing sets of the children do not intersect with each other.

Therefore a feasible solution exists for I_q if and only if the total length of the new jobs is less than or equal to the total remaining capacity of the machines with respect to M_{q-1}, and the total length of the new jobs of length 2 is less than or equal to the total remaining even capacity with respect to matching M_{q-1} (i.e., $\sum_{v_j : N(v_j) = V_q} 2 \left\lfloor \frac{r_{M_{q-1}}(v_j)}{2} \right\rfloor$). Moreover, if I_q is solvable then our algorithm finds such a feasible matching M_q that is economical for I_q (and so the final matching M is also economical for I_q).

Finally the claim that M is economical for I (if I is solvable) comes from the fact that, if V is not a processing set itself, then the processing sets of $\mathcal{V}$ with no superset can be considered as the children of V, thus if M is economical for each of the subinstances induced by these sets then M is economical for I too.

□

Corollary 3. *$PN|\mathcal{M}_j, p_j = \{1, 2\}|C_{max}$ is solvable in $O(e \log n)$ time, where e is the number of eligible job-machine pairs.*

Proof. It is obvious that the makespan is within the range of $[1, \ldots, 2n]$, hence we can find it by a standard binary search by running our algorithm at most $\lceil \log_2 n \rceil + 1$ times. □

Theorem 5. D-MS-N *with sizes $\{1, 2, 4, \ldots, 2^k\}$ is solvable in $O(e)$ time for any constant k, where e is the number of eligible job-machine pairs.*

Proof. We generalise the algorithm and proof given in the proof of Theorem 4 in the following way. Here let $U = U^0 \cup U^1 \cup \ldots \cup U^k$, where U^r is the set of jobs of length 2^r (for $0 \leq r \leq k$). Again, we consider the processing sets one by one according to the same ordering $\mathcal{V} = \{V_1, V_2, \ldots V_l\}$, and we allocate each job $u_i^r \in U^r$ with the actual processing set (in any order) using the following greedy rule. If, for the current matching M, there is a machine with remaining capacity at least 2^r then we allocate u_i^r to any machine

$v_j \in N(u_i^r)$ such that the remaining capacity of v_j satisfies the following strict inequality with the smallest possible s (where $r \le s < k$):

$$2^{s+1}\left\lfloor \frac{r_M(v_j)}{2^{s+1}} \right\rfloor < 2^s \left\lfloor \frac{r_M(v_j)}{2^s} \right\rfloor .$$

Otherwise, if $2^k \left\lfloor \frac{r_M(v_j)}{2^k} \right\rfloor = r_M(v_j)$ then we allocate u_i^r to any eligible machine v_j with $r_M(v_j) \ge 2^r$ (and we report that no feasible solution exists if $r_M(v_j) < 2^r$ for every machine v_j in $N(u_i^r)$).

In this setting, we say that a feasible matching is *economical* for a processing set V_p if $\sum_{v_j \in V_p} 2^k \left\lfloor \frac{r_M(v_j)}{2^k} \right\rfloor$ is maximal, and subject to $\sum_{v_j \in V_p} 2^{k-1} \left\lfloor \frac{r_M(v_j)}{2^{k-1}} \right\rfloor$ being maximal and subject to $\ldots \sum_{v_j \in V_p} 2 \left\lfloor \frac{r_M(v_j)}{2} \right\rfloor$ being maximal.

We claim that if I is solvable then our algorithm finds a feasible matching M. Moreover M is economical for each I_p (subinstance of I corresponding to processing set $V_p \in \mathcal{V}$) and economical for I as well. We will prove this by induction on p.

Again, it is easy to prove that the assumption is true for each subinstance I_p where the corresponding processing set, V_p, has no subset in $\mathcal{V}$. Let $G_p(U_p, V_p)$ be the induced subgraph of I_p and let M_{p-1} and M_p denote the matchings obtained by our algorithm before and after considering set V_p, respectively. G_p is a complete graph, obviously, and M_{p-1} has no edge in G_p. Therefore I_p is solvable if and only if the following inequality holds for every r, $0 \le r \le k$.

$$\sum_{u_i \in U_p \cap (U^r \cup U^{r+1} \cup \ldots \cup U^k)} s(u_i) \le \sum_{v_j \in V_p} 2^r \left\lfloor \frac{c(v_j)}{2^r} \right\rfloor .$$

Note that for $r = 0$ this means that the total length of the new jobs must be less than or equal to the total capacity of the new machines; for $r = 1$ we get that the total length of the new jobs of length 2 or more must be less than or equal to the total even capacity of the new machines, and so on. It is straightforward to verify that our algorithm finds a feasible matching for the subinstance I_p if it is solvable (resulting in matching M_p), and that M_p is economical for I_p.

Let us now consider a set V_q that is a superset of some set in $\mathcal{V}$. Here again, M_{q-1} is economical for the subinstance I_q without the new jobs ($\{u_i : N(u_i) = V_q\}$), since M_{q-1} is economical for each child of V_q according to our inductive assumption. Therefore a feasible solution exists for I_q if and only if the following inequality holds for every r, $0 \le r \le k$.

$$\sum_{u_i \in \{u_i : N(u_i) = V_q\} \cap (U^r \cup U^{r+1} \cup \ldots \cup U^k)} s(u_i) \le \sum_{v_j \in V_p} 2^r \left\lfloor \frac{r_{M_{q-1}}(v_j)}{2^r} \right\rfloor .$$

That is, for $r = 0$ we get that the total length of the new jobs is less than or equal to the total remaining capacity in V_q; $r = 1$ implies that the total length of the new jobs of length at least 2 is less than or equal to the total remaining even capacity in V_q, and so on. It is straightforward to verify that our algorithm finds a feasible solution for I_q if I_q is solvable, and that M_q is economical for I_q. We can also show that the final matching M is economical for I, in a way similar to the corresponding result in the proof of Theorem 4. □

Corollary 4. *$PN|\mathcal{M}_j, p_j = \{1, 2, 4, \ldots, 2^k\}|C_{max}$ is solvable in $O(e \log n)$ time, where e is the number of eligible job-machine pairs.*

Proof. It is obvious that the makespan is within the range of $[1, \ldots, 2^k n]$, hence we can find it by a standard binary search by running our algorithm at most $k\lceil \log_2 n \rceil$ times. □

6 Further notes

We conjecture that there is a polynomial time approximation algorithm for MS with sizes $\{1, 2, 3, \dots, k\}$ with an additive $(k-1)$ error. One might start to consider MS with sizes $\{1, 2, 3\}$ first. In this case, an approximation algorithm with additive error 2 would be a novel result. On the other hand, if it turns out that it is NP-hard to decide between $C_{max} \leq 3$ and $C_{max} \geq 5$ for $P|\mathcal{M}_j, p_j = \{1, 2, 3\}|C_{max}$, then this would lead to a $\frac{5}{3}$-inapproximability result (breaking the $\frac{3}{2}$ inapproximability ratio, which is the best known even for some more general settings as well).

Finally, we note that the gap is even larger at the moment for nested set systems. It is unknown whether MS-N is polynomial time solvable with sizes $\{1, 2, 3, \dots, k\}$ for any constant k larger than 2. Similarly, $PN|\mathcal{M}_j, p_j = \{1, 2, 3, \dots, k\}|C_{max}$ is open for any constant k larger than 2, whilst the best approximation ratio for $PN|\mathcal{M}_j|C_{max}$ is still $2 - \frac{1}{m}$.

References

[1] D.J. Abraham. Algorithmics of two-sided matching problems. Master's thesis, University of Glasgow, Department of Computing Science, 2003.

[2] N. Alon, T. Azar, G. Woeginger, and T. Yadid. Approximation schemes for scheduling. In *Proceedings of the 8th ACM-SIAM Symposium on Discrete Algorithms*, pages 493–500. ACM-SIAM, 1997.

[3] P. Biró. Matching with couples is NP-complete. Unpublished manuscript, 2007.

[4] D. Gale and L.S. Shapley. College admissions and the stability of marriage. *American Mathematical Monthly*, 69:9–15, 1962.

[5] N. Garg, T. Kavitha, A. Kumar, K. Mehlhorn, and J. Mestre. Assigning papers to referees. Unpublished manuscript, 2009.

[6] C.A. Glass and H. Kellerer. Parallel machine scheduling with job assignment restrictions. *Naval Research Logistics. A Journal Dedicated to Advances in Operations and Logistics Research*, 54(3):250–257, 2007.

[7] C.A. Glass and H.R. Mills. Scheduling unit length jobs with parallel nested machine processing set restrictions. *Computers & Operations Research*, 33:620–638, 2006.

[8] N.J.A. Harvey, R.E: Ladner, and L. László. Semi-matchings for bipartite graphs and load-balancing. *Journal of Algorithms*, 59:53–78, 2006.

[9] R.W. Irving, T. Kavitha, K. Mehlhorn, D. Michail, and K. Paluch. Rank-maximal matchings. *ACM Transactions on Algorithms*, 2(4):602–610, 2006.

[10] R.M. Karp. Reducibility among combinatorial problems. In R.E. Miller and J.W. Thatcher, editors, *Complexity of Computer Computations*, pages 85–103. Plenum Press, 1972.

[11] B. Klaus, F. Klijn, and J. Massó. Some things couples always wanted to know about stable matchings (but were afraid to ask). *Review of Economic Design*, 11:175–184, 2007.

[12] J.K. Lenstra, D.B. Shymoys, and E. Tardos. Approximation algorithms for scheduling unrelated parallel machines. *Mathematical Programming*, 46:259–271, 1990.

[13] J.Y.-T. Leung and C.-L. Li. Scheduling with processing set restrictions: A survey. *International Journal of Production Economics*, 116:251–262, 2008.

[14] D.F. Manlove and E. McDermid. Keeping partners together: Algorithmic results for the Hospitals / Residents problem with couples. *Journal of Combinatorial Optimization*, (forthcoming), 2009.

[15] D. Marx and I. Schlotter. Stable assignment with couples: Parameterized complexity and local search. In *Proceedings of the 4th IWPEC: International Workshop on Parameterized and Exact Computation*, Lecture Notes in Computer Science. Springer, 2009.

[16] S. Thomas McCormick, Scott R. Smallwood, and Frits C. R. Spieksma. A polynomial algorithm for multiprocessor scheduling with two job lengths. *Math. Oper. Res.*, 26(1):31–49, 2001.

[17] C.St.J.A. Nash-Williams. On orientations, connectivity and odd vertex pairings in finite graphs. *Canadian Journal of Mathematics*, 12:555–567, 1960.

[18] `http://www.nrmp.org/about_nrmp/how.html` (National Resident Matching Program website).

[19] P.A. Robards. Applying two-sided matching processes to the United States Navy enlisted assignment process. Master's thesis, Naval Postgraduate School Monterey CA, 2001.

[20] E. Ronn. NP-complete stable matching problems. *Journal of Algorithms*, 11:285–304, 1990.

[21] A. E. Roth. The economist as engineer: Game Theory, Experimentation, and Computation as tools for design economics. *Econometrica*, 70(4):1341–1378, 2002.

[22] `http://www.nes.scot.nhs.uk/sfas/` (Scottish Foundation Allocation Scheme website).

[23] E.V. Shchepin and N. Vakhania. An optimal rounding gives a better approximation for scheduling unrelated machines.

[24] M.M. Short. Analysis of the current navy enlisted detailing process. Master's thesis, Naval Postgraduate School Monterey CA, 2000.

[25] W. Yang, J.A. Giampapa, and K. Sycaratech. Two-sided matching for the U.S. Navy detailing process with market complications. Technical Report CMU-RI-TR-03-49, Robotics Institute, Carnegie Mellon University, 2003.