The modular tool switching problem

Csaba Raduly-Baka[a,*], Olli S. Nevalainen[a]

[a]*Department of Information Technology, 20014, University of Turku, Finland*

Abstract

This article analyzes the complexity of the modular tool switching problem arising in flexible manufacturing environments. A single, numerically controlled placement machine is equipped with an online tool magazine consisting of several changeable tool feeder modules. The modules can hold a number of tools necessary for the jobs. In addition to the online modules, there is a set of offline modules which can be changed to the machine during a job change. A number of jobs are processed by the machine, each job requiring a certain set of tools. The sequence of jobs is given as part of the input and fixed. Tools between jobs can be switched individually, or by replacing a whole module containing multiple tools. We consider the complexity of the problem of arranging tools into the modules, so that the work for module and tool loading is minimized. Tools are of uniform size and have unit loading costs. We show that the general problem is NP-hard, and in the case of fixed number of modules and fixed module capacity the problem is solvable in polynomial time.

Keywords: Complexity theory, Flexible Manufacturing Systems, Tool loading, Set-up optimization, Printed circuit board

1. Introduction

In the assembly of printed circuit boards (PCB), flexible component placement machines are used to mount components onto a bare PCB. The placement machines are highly automatized, configurable and suitable for the assembly of a wide range of PCB product types. The flexibility and configurability of these machines results in various efficiency problems. The planning and control of PCB assembly machines is a task that consist of multiple interconnected problems (see [4], [8] and [2]).

One of these problems arises when several different PCB product types are manufactured on a single machine, with each product type requiring a specific set of component types to be placed on the boards. The component placement machines are equipped with a feeder unit that can hold a sufficient number of component input reels to manufacture a single product type. Due to the capacity constraints of the feeder unit, it is not possible to load at once all the component reels required for all the PCB jobs.

In the case of multiple PCB types, as one PCB type batch ends, the content of the feeder unit must be reconfigured, by loading the component reels required for the next PCB type. This replacement can occur only between processing jobs, because the machine must be stopped. Replacing one component reel causes extra delay in production, and the total delay between two jobs is determined by the number of component reel swaps.

The above discussion deals with the organization of job switching in the context of PCB assembly manufacturing. The same situation occurs in many other Flexible Manufacturing Systems (FMS), where tools (above component reels) are stored in tool magazines (feeder units). Minimizing the setup delay is known in literature as the *Tool Switching Problem* (TS) (see [19]), in this more general context.

Many variants of the tool switching problem have been extensively studied in literature. The order in which the jobs are processed may be fixed or arbitrary. If the job order is arbitrary, the tool switching cost can be further reduced by finding a better job sequence (this is also known as the *job sequencing problem*). The cost of switching tools may be

*Corresponding author. Tel: +358 45 1167977
Email address: `csraba@utu.fi` (Csaba Raduly-Baka)

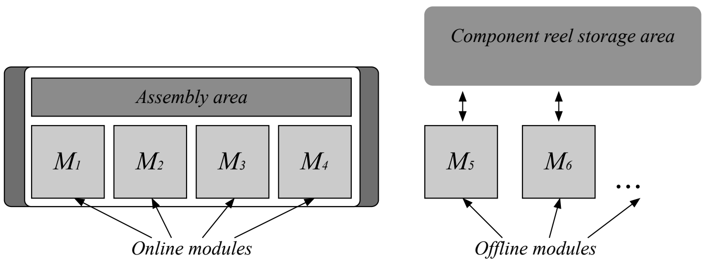

Figure 1: A placement machine with modular component feeder unit. Individual modules (M_i) can be replaced in one step with offline modules. The content of offline modules can be prepared in parallel with the processing of the previous assembly batch.

uniform or tool specific. Further, the size of the tools may be uniform, or tool specific, causing fragmentation of the feeder unit in the latter case. The job sequencing problem is NP-hard even in the case of uniform tool sizes and equal switching costs ([6]).

When the job sequence is fixed, the tool switching problem with uniform tool sizes and equal change costs can be solved optimally in polynomial time by the means of the KTNS procedure of Tang and Denardo [19]. When tool sizes are non-uniform, the problem is NP-hard even for the fixed job sequence (as shown in [5], where the problem is called the tool loading problem).

Over the past decade, component placement machines have increasingly employed modular feeder units to improve versatility, flexibility and efficiency of these equipments. Such examples are Fuzion from Universal, iFlex from Assemblon, and BM/NPM from Panasonic. These machines employ several feeder module units that are online at the same time. The modules contain component reels, and can be swapped with readily available offline modules, that have been prepared in parallel to the processing of the current assembly job.

The present work focuses on the practical problem of switching component reels (tools) in a single placement machine containing multiple removable feeder modules of equal capacity. Each feeder module contains a number of component reels (typically 40), and one or more online modules can be replaced by another module which has been prepared to contain a different set of component reels. Swapping a feeder module with another one incurs a much smaller delay (cost) than swapping some individual component reels. Therefore, switching components in groups can reduce the overall setup time required when transitioning between PCB job types. Component reels can also be swapped individually in the online modules if that is preferred (in case of small setup changes), making the problem more difficult than the basic tool switching problem.

The problem of switching tools in groups to improve production efficiency is not considered by the tool switching literature to our knowledge. Let us call this problem the *modular tool switching (MTS) problem*, where in the present context, tools are component reels required to manufacture PCB jobs (see Fig. 1.). In the present work, the tools have *uniform sizes* and the delay caused by switching tools to online modules is also uniform across tool types. The delay of switching a feeder module is a constant, different from the tool switching delay. We show that by introducing online and offline modules into the tool switching problem, the problem becomes NP-hard even in the case of a fixed job sequence, uniform tool sizes and equal tool switching delays.

1.1. Literature review

The tool switching literature focuses mainly on solving two problems. The first one deals with a fixed job sequence (see Tzur and Altman [13]). The second problem also includes the task of finding an order of the jobs that minimizes the tool switching costs. The general tool switching problem was introduced by Tang and Denardo [19]. They proposed a polynomial time, optimal algorithm called Keep Tool Needed Soonest (KTNS), for the case of a fixed job sequence, uniform tool sizes and tool switching costs.

The case of fixed job sequence, uniform tool size, and tool specific changeover cost was discussed by Privault et. al. in [14]. They show that if the changeover costs are of the form d_{ik}, where tool i is inserted after removing tool k, then the problem can be solved optimally by formulating it as a min-cost flow problem.

For non-uniform tool sizes and arbitrary magazine capacity C (introduced by Stecke et. al. in [18]), the problem becomes NP-hard even for a fixed job sequence, as shown by Crama et. al. in [4]. Heuristic methods for solving the problem were given in [12] and in [20]. However, when the magazine capacity C is fixed, Crama et. al. [4] show that the problem admits a polynomial-time optimal algorithm, albeit with a very large exponent, making it unpractical for industrial application.

For the case of job sequencing with uniform tool sizes, Crama et. al. [6] show that the problem is NP-hard, even for $C = 2$. Heuristics to solve the problem have been proposed in [3], [11], [9], [16], [17] and [21]. The approximability of the problem was discussed by Crama et. al. [7]. Fawzan et. al. [1] use a tabu search approach to find better job sequences, that minimize the number of tool switches.

For the case of job sequencing with non-uniform tool sizes, two NP-hard problems (tool switching and job sequencing) are combined into a single optimization objective. Heuristics to solve the combined problem have been discussed in [13] and [15].

While the case of modular tool magazine and extra offline modules is an important practical application, previous research on it seems to be missing.

1.2. Problem definition

Next, the definition and assumptions of the *modular tool switching problem* are given, in the context of loading component reels into feeder modules of a PCB assembly machine. Suppose, that a list J of n PCB assembly jobs is given. The jobs are processed by a single component placement machine. The order in which these jobs are processed is fixed by the list J. Each job j ($j \in [1...n]$) requires the insertion of a set T_j of different component types (these are considered tools) and the components of a job are supplied by the means of a modular feeder unit (tool magazine).

The feeder unit has sufficient capacity to hold all the component reels required for a job. The feeder unit contains a number of F changeable modules (typically from 2 to 6). Each module has the same fixed capacity C to hold the component reels. A number of E offline modules are available in the vicinity of the machine. The capacity of each offline module is the same C, and their content can be manually rearranged, in parallel to the processing of the current job, having no impact on the production time.

The available capacity in all the online modules is $F \cdot C$, and it is not sufficient to hold all the component reels for all the PCB types: $|\cup_{j\in[1...n]} T_j| > F \cdot C$. Therefore, component reel changes are possibly required between jobs. Reels can be changed individually, incurring a delay of t_c for each change, or in group by changing a whole module, incurring a delay of t_m for each module change. By changing a module, one can change a total of C component reels in one step (C is typically 20 to 40) and incur a single t_m delay that is much smaller than a $C \cdot t_c$ delay of changing all reels of a module while changing jobs. (In practice the t_m/t_c ratio tends to be between 2 and 4, and it is 2.5 for the machine types studied here). Therefore, arranging component reels into offline modules, so that they can be swapped in a group when transitioning to the next job, is advantageous in terms of the incurred delay.

Next, the parameters and assumptions of the MTS problem are listed:

C	the capacity of a feeder module. All feeder modules have equal capacity.
F	the number of online feeder modules.
E	the number of offline modules. It is assumed that at least one offline module is available. Otherwise the problem simplifies to the well known tool switching problem which can be solved optimally by the means of the KTNS procedure.
n	the number of PCB assembly jobs. The order in which jobs are processed is given and fixed by list J.
T_j	the set of component types required for job j. It is assumed that a single job will always fit into the available online capacity ($|T_j| \leq F \cdot C$), and that there is no sufficient online capacity for all the jobs: ($|\cup_{j\in[1...n]} T_j| > F \cdot C$). All components are of uniform size, i.e. demand a unit space (single slot) in the feeder module.
t_c	the delay incurred when swapping one component reel in an online module.
t_m	the delay incurred when swapping one online module with an offline module.

It is assumed that the completion time of each job is sufficiently large that offline modules can be prepared before the job completes. This simplifying assumption is valid mainly because jobs are typically batches of identical PCB units.

Extra copies of component reels may or may not be available. If extra copies are available, these make it easier to setup offline modules, while the current job is being processed. The number of extra copies may be limited per component reel type, or it may be that no extra copies are available due to budgetary constraints. There are F online modules and there is no need to place more than one reel of a type into a module. Therefore, it is sufficient to have $F+1$ copies available of each component type reel, so that at least one reel is available offline, and there is no need to wait for a job to complete when building offline modules. Otherwise, some component reel types may be locked into online modules, so an offline module can be completed only after the previous job completes, and at that moment all the component reel switchings will be counted in the total delay.

2. Complexity

The problem of switching component reels (tools) having equal sizes, equal switching costs, and no offline modules, can be solved by the means of the KTNS procedure. In the case of modular feeder units, with at least one offline module, the problem becomes more complicated by the fact that switching modules incurs a smaller cost than switching individually several component reels of a module.

Next, we show that if the capacity constraints F and C are arbitrary, the MTS problem is NP-hard. In fact there are four cases of the MTS problem in terms of the capacity constraints. Either F or C can be fixed, or can be arbitrary parameters. In the case of either F or C are arbitrary, the problem is NP-hard. Only when both F and C are fixed, the problem becomes polynomially solvable.

The 4 cases of the MTS problem are:

- The *General MTS Problem* (or simply the *MTS* problem) is the problem of minimizing the delay incurred by component reel and module switches, when both the number of online modules F and their capacity C are arbitrary.
- The *MTS-F* problem is the special case of MTS problem, when the number of online modules F is fixed, but the capacity C of these modules is arbitrary.
- The *MTS-C* problem is the special case of MTS problem, when capacity of the modules C is fixed, but the number of online modules F is arbitrary.
- The *MTS-FC* problem is the special case of MTS problem, when both the number of online modules and their capacity are fixed.

Next, we show that the MTS, MTS-C and MTS-F problems are NP-hard, while the MTS-FC problem can be solved in polynomial time, albeit with very large exponent.

2.1. MTS: The general case

In the General MTS (or just MTS) problem, F and C are arbitrary. We show that this problem is NP-hard, by showing that its decision version (*Decision-MTS* problem) is NP-complete. The proof is inspired by the complexity proof for the traditional tool switching problem with non-uniform tool sizes by Crama et. al. [5].

In the Decision-MTS problem, a fixed integer L is given and it is asked whether there is a module and component reel switching policy so that the total delay is less or equal than L. We use a reduction from 3-Partition which is known to be strongly NP-complete ([10]). The 3-Partition problem is defined as follows:

2.1.1. 3-Partition

Given a multi-set S of $3n$ positive integers w_k, and a positive integer B, where $B/4 < w_k < B/2$ for each w_k, and $\sum_{k \in S} w_k = nB$. The problem asks to partition S into n disjoint triples $S_1, S_2, ..., S_n$, so that for each $1 \leq i \leq n$, the sum of integers in S_i is B: $\sum_{k \in S_i} w_k = B$.

2.1.2. Complexity of the MTS problem

Lemma 1. Decision-MTS *is NP-Complete, with at least one offline module and with or without extra component reels.*

Proof. Clearly, Decision-MTS is in NP: given a module setup and component reel switching policy, it is easy to sum up the setup delay incurred between jobs. Next, we show that the 3-Partition problem can be reduced into a Decision-MTS problem.

In the Decision-MTS problem, one or more offline modules can help to reduce the cost of component reel switches. First, we consider the case where at least one extra copy is available for each component reel type. Then, with a small change, we show that the reduction works even if no extra copies of component reels are available.

One extra copy for each component type. Given an arbitrary instance I of the 3-Partition problem, a Decision-MTS problem instance will be constructed in the following way:

- Set the component reel switching cost $t_c = 6n + 1$ and the module switching cost $t_m = 1$. The component reel switching cost can be any number equal or greater than $6n + 1$, the goal is to force the Decision-MTS problem to avoid switching component reels.
- For each value w_k ($k = 1, 2, ..., 3n$) of I, define a set of basic component types $T_k = \{t_{k1}, t_{k2}, ..., t_{kw_k}\}$, $|T_k| = w_k$. There are a total of $n \cdot B$ different component types.
- Define an *extra* set of component types X, $|X| = B$. These component types are different from the ones in the basic sets T_k.
- Let J_a be a *basic* job requiring all the basic component types from $\cup_{k=1}^{3n} T_k$. There are a total of nB component reels required by job J_a.
- Let J_k be an *extra* job containing the component types $(\cup_{i=1}^{3n} T_i) \backslash T_k$, and w_k component types from X. Job J_k also requires exactly $n \cdot B$ component reels. Component types from X are reused between various jobs J_k, by selecting a w_k number of component types from X.
- Each of the $3n$ component type set T_k induces a different job J_k.
- Create a job sequence of $6n + 1$ jobs: $J_a, J_1, J_a, J_2, J_a, ..., J_a, J_{3n}, J_a$.
- Each of the $6n+1$ jobs requires exactly nB reel types.
- Let the number of online modules be $F = n$, the capacity of a module $C = B$, and use at least one offline module of capacity C. The total available online capacity is thus $n \cdot B$.

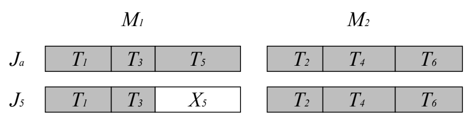

Figure 2: The setup for job J_5 after J_a requires swapping module M_1. The replacement module for M_1 can be prepared offline if extra component reel copies are available.

- Clearly, each job fully uses the available online capacity.
- Each pair of consecutive jobs (J_a, J_k or J_k, J_a) differ by a number of w_k component types, and these reels must be switched between each job (see Fig. 2).
- There are $6n$ transitions between the $6n + 1$ jobs. It is assumed that the machine starts with all the component reels available online for job J_a.
- Since each transition must change at least w_k component reels, the optimal solution must have at least one module change incurring a cost of $t_m = 1$ per transition.
- Set $L = 6n$, the Decision-MTS problem will ask if there is a module and component reel swapping policy with a delay no more than L.
- All component reels have an extra copy available, so that offline modules can be prepared before the current job completes. One extra copy is sufficient, since there will be no need to duplicate copies in online modules. Each component type will have at most one reel online and at least one reel offline.

The above transformation creates a Decision-MTS problem instance that has a size exponential in the size of the 3-Partition problem, in case of binary representation. This is because for each value w_k we create w_k different components. If w_k is binary encoded, the number of components will be exponential in the size of w_k. Since 3-Partition is a strongly NP-complete problem, it is still NP-complete if unary coding of w_k is used. Therefore, the unary coded representation of the 3-Partition is considered, and it is transformed into a Decision-MTS problem of similar size.

1. To show NP-Completeness (with one extra copy for each component type), we show that a YES-instance of Decision-MTS problem translates into a YES-instance of 3-Partition and vice versa.

Consider that there is a YES-instance for Decision-MTS problem with $L = 6n$. A consecutive pair of jobs is either of the form (J_a, J_k) or (J_k, J_a). The difference between jobs J_k and J_a is exactly w_k components, as the set of T_k components required by job J_k are replaced by w_k components from the set X. Therefore, there must be some reel changes between two consecutive jobs.

There are $6n$ job transitions, and for $L = 6n$ the answer of the Decision-MTS problem was a YES. Clearly, this can happen if and only if there is exactly one module swap between each consecutive jobs (the delay of the module swap being $t_m = 1$). But this implies that all the w_k component reels (basic or extra) are located in one module. That is, the $(2k)$th and $(2k + 1)$th job transition cannot be done with a delay of 1 if the replaced w_k components of T_k or X are spread over multiple modules.

A transition from job J_a to job J_k can be done by a single module in the following way. Let M_i be the module containing reels from the set T_k while J_a is processed. Prepare a new module M_i' offline, that has the same content as M_i except the reels of T_k are replaced with w_k reels from X. M_i' can be prepared offline because there is at least one extra offline copy for each reel (the other $B - w_k$ reels of module M_i are also required for module M_i').

Each job requires exactly $n \cdot B$ reels. Each module holds at most B reels. Each set of T_k reels is fully contained in a module. A module cannot contain 4 or more T_k sets of reels (because $B/4 < w_k$). There are $3n$ sets of reels loaded into the n online modules, then each module contains exactly three sets of reels. Since the total online capacity $n \cdot B$ is fully used by each job, each module contains exactly B reels. Thus, when a job J_a is being processed, for each online module M_i we have $\sum_{T_k \in M_i} |T_k| = B$. This directly translates into a YES-instance of the 3-Partition problem, where S_i corresponds to module M_i and $\sum_{k \in S_i} w_k = B$.

2. Now we show that a YES-instance of 3-Partition corresponds to a YES-instance of Decision-MTS problem with $L = 6n$. Given a solution S_i to 3-Partition, assign modules M_i ($i = 1, 2, ..., n$) with reels from T_k for all $k \in S_i$. The capacity of a module is B, and from 3-Partition we will have $\sum_{k \in S_i} |T_k| = B$. Thus, all modules are completely filled with reels, and all $3n$ reel sets are made available online for job J_a.

At the transition from J_a to J_k there is exactly one module M_i that contains the set of reels T_k. Prepare an offline module M_i' containing the other $B - w_k$ reels of M_i and w_k reels from X. Swap M_i with M_i' at cost $t_m = 1$. A similar transition can be made from J_k to J_a. Thus, the given job list can be processed with no more than $6n$ module transitions, with each transition at a cost of $t_m = 1$. This results in a YES-instance for the Decision-MTS problem with $L = 6n$.

No extra copies of component reels. Consider the case where no extra reels are available to build offline modules. We show that a reduction exists from 3-Partition even in this case.

If there are sufficient extra copies of each component reel, a complete offline module can be prepared without using reels from online modules. This is a relaxed assumption of the problem. If there are no such extra copies, the offline module cannot be prepared completely and the above reduction will not hold. With a small change to the job list, the reduction can be modified so that it holds for no extra copies as well.

Define a new set Y containing B different component types, these are also different from component types in T_k and X. Define a new job J_y, that contains exactly B component types from Y. Now define a job sequence of the following form:

$$J_a, J_y, J_1, J_y, J_a, J_y, J_2, J_y, J_a, ..., J_a, J_y, J_{3n}, J_y, J_a.$$

There are a total of $12n + 1$ jobs, with $12n$ transitions. The J_a, J_y transition will unload all reels from a module M_i, and load one module containing only reels from Y. The reels of module M_i then are available offline, so that a new module can be constructed for job J_k (while job J_y is being processed) see Fig. 3. The transition back to job J_a is similar, a job J_y is inserted in-between in order to unload all reels that are required to construct an offline module for J_a.

In this instance of the Decision-MTS problem, the limit is set to $L = 12n$, and the reel switching cost is $t_c = 12n + 1$. Similarly to the earlier reduction, it is clear that a $12n$ delay can be achieved if and only if there is a 3-Partition of the instance.

□

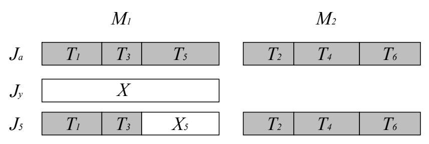

Figure 3: No extra copies of component reels are available. Inserting job J_y between jobs J_a and J_5 will cause the unloading of component sets T_1, T_3 and T_5. These become available to build an offline module, while J_y is being processed, and load the module in one step when transitioning to J_5.

2.2. MTS-F: fixed number of online modules

The above reduction for the General MTS problem assumed that both the number of online modules and their capacity are arbitrary. In the MTS-F problem the number of online modules is fixed and the capacity of each module is arbitrary. Showing the complexity of $MTS-F$ problem requires a different reduction, since in the earlier case the variable n of 3-Partition was mapped to the number of online carriages F.

In the case of MTS-F problem, a reduction from the Bin-Packing problem is used. Then the Bin-Packing problem is also known to be strongly NP-complete ([10]). As before, the following reduction considers the unary representation of the values in Bin-Packing. Since the Bin-Packing problem is strongly NP-complete, it remains NP-Complete even in the case of unary representation of the problem.

2.2.1. Bin-Packing

Given a multi-set of numbers $A = \{a_1, a_2, ..., a_n\}$ and a bin capacity V, find a minimum number of bins m and a partition of the numbers of A into the bins $B_1, B_2, ..., B_m$ such that $\sum_{k \in B_i} \leq V$ for all $1 \leq i \leq m$. The decision version of Bin-Packing (b-Bin-Packing) asks if it is possible to pack the given input A into b bins each of the capacity V.

2.2.2. MTS-F complexity

Lemma 2. Decision-MTS-F *problem is NP-Complete, with at least one offline module and with or without extra component reels.*

Proof. Clearly, the Decision-MTS-F is in NP. We will show that an arbitrary instance of the b-Bin-Packing problem can be reduced to the Decision-MTS-F problem. Given an instance of the b-Bin-Packing problem, construct the following Decision-MTS-F problem instance:

- Set the reel switching cost to $t_c = 2n + 1$ and the module switching cost $t_m = 1$.
- For each value a_k of A, define a set of basic component types $T_k = \{t_{k1}, t_{k2}, ..., t_{ka_k}\}$, with $|T_k| = a_k$.
- Define an *extra* set of component types X, $|X| = V$ (i.e. the capacity of a bin). These component types are different from the ones in the basic sets T_k.
- Let the number of online modules $F = b$, and the capacity of a module $C = V$.
- Let $S = \sum_{k=1}^{n} a_k$. Clearly $S \leq b \cdot V$, otherwise the b-Bin-Packing problem is not feasible.
- Define another set of component types Z containing $b \cdot V - S$ different component types.
- Component types of Z are used to complete the jobs so that they fully use the available online capacity (see Fig. 4).
- Let J_a be a *basic* job requiring all the basic component types from all T_k sets and the component types from Z. Then, J_a requires exactly $b \cdot V$ capacity.
- Let J_k be an *extra* job containing the components $(\cup_{i=1}^{n} T_i) \backslash T_k$, all components from Z, and w_k component types from X. Job J_k also requires exactly $b \cdot V$ capacity.
- There are a total of n basic component type sets T_k and each of them induces a different job J_k.
- Create a job sequence of $2n + 1$ jobs: $J_a, J_1, J_a, J_2, J_a, ..., J_a, J_n, J_a$.
- There are a total of $2n + 1$ jobs in the sequence, each requiring exactly $b \cdot V$ component reel types.
- Each consecutive pair of jobs differs by a number of w_k component types, these reels must be switched between each job.
- There are $2n$ transitions between the $2n + 1$ jobs.
- Since each transition must incur at least w_k component reel changes, the optimal solution must have a minimum of one module switch incurring a cost of $t_m = 1$ per transition.

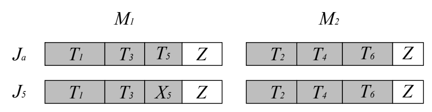

Figure 4: The setup for job J_5 after J_a requires swapping module M_1 only. The components of Z will fill up a module to full capacity.

- Set $L = 2n$, that is the Decision-MTS-F problem will ask whether there is a module and component reel swapping policy with a delay no more than $2n$.
- All component reels have an extra copy available, so that offline modules can be prepared before the current job completes.

To show NP-Completeness, we show that a YES-instance of Decision-MTS-F problem translates into a YES-instance of b-Bin-Packing, and vice versa. The reduction is similar to that of the General MTS problem.

Clearly, in a YES-instance for Decision-MTS-F problem with $L = 2n$, there is exactly one module swapped between two consecutive jobs. This means that each reel set T_k was fully contained by one module.

A module M_i will contain the reel sets T_k that correspond to a bin assignment B_i. The extra reels of Z are used to fill the empty capacity of each module. If b online modules, each of capacity V, are sufficient to achieve one module swap per job transition (i.e. a reel sets T_k is fully contained in a module M_i), then b bins are sufficient to pack all a_k numbers of the Bin-Packing instance. Thus, for a YES-instance of Decision-MTS-F problem we have a YES-instance of Bin-Packing.

Suppose, that the answer to the b-Bin-Packing problem is YES. For each bin B_i containing numbers a_k, assign the corresponding reel sets T_k to module M_i. Since the Bin-Packing instance fits into b bins, each reel set T_k can be placed into exactly one of the b online modules. Because a reel set is fully contained in one module, a single module swap is sufficient between each job J_a and J_k. This results in a total delay of $2n$, implying a YES-instance for the Decision-MTS-F problem.

The reduction in the case of no extra component reel copies is handled in a similar way as in the case of General MTS problem. Let Y be an additional set of V component types, and J_y a job containing the V component types from Y. Job J_y is inserted after each J_a and J_k to allow the unloading of reels required to construct a module offline. Transitions between J_y and other jobs can be handled by exactly one module swap. In order to avoid extra cost, the swapped module must contain the changing reel set T_k.

$\square$

2.3. MTS-C: fixed module capacity

In the MTS-C problem, the capacity of a module C is fixed and the number of online modules F is arbitrary. For MTS-C problem we use a reduction from the *Minimum Makespan Scheduling Problem*, that is also known to be strongly NP-hard ([10]).

2.3.1. Minimum Makespan Scheduling

In the *Minimum Makespan Scheduling* Problem (MMS) n jobs are given, each with an integer processing time p_k, and m identical machines. Find the assignment of the n jobs to the m machines so that the maximum completion time (also called *makespan*) is minimized.

The MMS problem is strongly NP-hard for arbitrary m. The decision version of the MMS problem (Decision-MMS) asks if there is a job/machine assignment so that the makespan is not larger than some fixed integer C. Decision-MMS is NP-complete for an arbitrary value of m.

2.3.2. MTS-C complexity

Lemma 3. Decision-MTS-C *problem is NP-Complete, with at least one offline module and with or without extra component reels.*

The proof of NP-Completeness of the Decision-MTS-C problem follows similar steps as the earlier reductions, the details are given in the Appendix.

2.4. MTS-FC: Fixed online and module capacity

In the MTS-FC problem both the number of online modules F and their capacity C are fixed. We show that the problem can be solved optimally in polynomial time, albeit with a very large exponent. The polynomial-time algorithm follows similar ideas to those given by Crama et. al. [5], for the case of non-uniform tool sizes and single linear tool magazine, except that here we solve the problem by dynamic programming.

Consider all the possible component reel assignments to the online modules. A given assignment to F online modules may or may not be feasible. The assignments that are not feasible are not considered (incur ∞ cost). The transition between two module assignments can be calculated easily and efficiently by either swapping components or replacing modules if such modules can be constructed offline.

Let $m = |\cup_{j\in[1\dots n]} T_j|$ be the total number of different components used in the manufacturing of n jobs. Consider one extra dummy component type, which is used to fill unused slots in a module. Suppose that there is an arbitrary number of copies available of the dummy component. Thus, a module can be filled with $m+1$ components in at most $O((m+1)^C)$ different ways (depending how many extra copies of each reel are available). With F online modules there are at most $O((m+1)^{C\cdot F})$ configurations for online components. An online configuration is feasible for a job j if all the required component types are present, and the number of component reel duplications don't exceed the available component reel copies.

Let D be the set of all the $O((m+1)^{C\cdot F})$ possible online configurations, and $d, d' \in D$ be two such configurations. Then, the transition function $R(d, d')$ calculates the delay incurred by changing from configuration d to d'. The function R can be efficiently calculated for a fixed online capacity F and C.

Given a number of n jobs, the delay for the minimum module/component reels swapping can be calculated by dynamic programming using the following recurrence relation:

The initial online magazine setup consists of loading no more than $C \cdot F$ component reels into the feeder unit:

$$OPT(1, d') \leq C \cdot F \cdot t_c$$

For each $d' \in D$ that contains all the components for job j

$$OPT(j, d') = min_d\{OPT(j-1, d) + R(d, d')\}$$

and let

$$OPT(j, d') = \infty$$

if $d' \in D$ is not feasible for job j.

That is, the minimum delay of reaching job j with state d' is the minimum of the sum of reaching job $j-1$ with a state d and transitioning from state d to d'.

Lemma 4. *The dynamic programming recurrence defined by $OPT(n, d)$ results in an optimal solution to the MTS-FC problem.*

Proof. By induction, assume that $OPT(n-1, d)$ is the optimal value to reach job $n-1$ with a feasible state d. Let $d' \in D$ be a feasible state for job n. Lets assume that this state can be reached by a delay Q, where $Q < OPT(n, d')$.

Let $d'' \in D$ be the state used by job $n-1$, in the solution that obtains delay Q. Then we have

$$Q = OPT(n-1, d'') + R(d'', d')\} <$$

$$min_d\{OPT(n-1, d) + R(d, d')\}$$

since Q is optimal and $R(d'', d')$ can be calculated efficiently. But this is a contradiction, since min_d minimizes over all possible configurations, thus $Q = OPT(n, d')$.

That is, there is no solution that results in a smaller delay than the one given by the recurrence relation $OPT(n, d')$. $\square$

The above dynamic program calculates the minimum setup delay when manufacturing n jobs. The actual module configurations can be recovered, by recording at each step the choice made for d that achieves the minimum. The dynamic program will involve a table of size $O(n \cdot (m+1)^{F\cdot C})$.

Although this is polynomial for a fixed F and C, in practice the values of m, F and C are large enough to make this approach prohibitive. It is typical in industrial cases that $m \geq 100$, while $F = 4$ and $C = 40$. Making the DP table as large as 100^{160} per job.

3. Conclusion

This study considered the complexity of the modular tool switching problem. The problem arises in flexible manufacturing environment, especially in the case of component placement machines in PCB manufacturing, where multiple online and offline feeder modules are available for a single machine. It was shown that the modular tool switching problem is NP-hard in the general case, even for a fixed job sequence, unit loading cost and uniform tool sizes. The problem can be solved optimally in polynomial time if both the number of online feeder modules and the capacity of these modules are fixed.

In practical applications of the MTS problem, both the number of feeder modules and their capacity are constants. These are characteristics of the component placement machines. Nevertheless, the number of different component types and the total capacity of the online feeder modules make an exact solution prohibitive. Thus, heuristic approaches are necessary to find good feeder module arrangements and reel/module swapping decisions. Such heuristics are subject of active research in our group, and will be considered in future for publication.

[1] Al-Fawzan, M. A., Al-Sultan, K. S., 2003. A tabu search based algorithm for minimizing the number of tool switches on a flexible machine. Computers & industrial engineering 44, 35-47.

[2] Ayob, M., Kendall, K., 2008. A survey of surface mount machine optimization: Machine classification, European Journal of Operational Research, Vol. 186, pp. 893-914.

[3] Bard, J. F., 1988. A heuristic for minimizing the number of tool switches on a flexible machine. IIE transactions 20, 382-391.

[4] Crama, Y., van de Klundert, J., Spieskma, F.C.R., 2002. Production planning problems in printed circuit board assembly, Discrete Applied Mathematics, Vol. 123, No 1-3, pp. 339-361.

[5] Crama, Y., Moonen. L.S., Spieskma, F.C.R., Talloen, E., 2007. The tool switching problem revisited, European Journal of Operational Research, 182, pp. 952-957.

[6] Crama, Y., Kolen, A.W.J., Oerlemans, A.G., 1994. Minimizing the number of tool switches on a flexible machine, The International Journal of Flexible Manufacturing Systems, Vol. 6, pp. 33-54.

[7] Crama, Y., van de Klundert, J., 1999. Worst-case performance of approximation algorithms for tool management problems. Naval research logistics 46, 445-462.

[8] Crama, Y., 1997. Combinatorial optimization models for production scheduling in automated manufacturing systems. European Journal of Operational Research 99, 136-153.

[9] Djellab, H., Djellab, K., Gourgand, M., 2000. A new heuristic based on the hypergraph representation for the tool switching problem. International Journal of Production Economics, 64, pp. 165-176.

[10] Garey, M.R., Johnson D.S., 1979. Computers and intractability. A guide to the theory of NP-completeness. W.H. Freeman and Company, New York.

[11] Hertz, A., Laporte, G., Mittaz, M., Stecke, K. E. (1998). Heuristics for minimizing tool switches when scheduling part types on a flexible machine. IIE transactions, 30(8), 689-694.

[12] Hirvikorpi, M., Salonen, K., Knuutila, T., Nevalainen, O., 2006. General Two Level Storage Management Problem - A reconsideration of the KTNS-Rule, European journal of operational research 171, 189-207.

[13] Matzliach, B., Tzur, M., 2000. Storage Management of Items in Two Levels of Availability, European Journal of Operational Research 121, 363-379.

[14] Privault, C., Finke, G., 1995. Modeling a tool switching problem on a single NC-machine. Journal of Intelligent Manufacturing 6, 87-94.

[15] Raduly-Baka, Cs., Knuutila, T., Nevalainen, O., 2005. Minimizing the Number of Tool Switches with Tools of Different Sizes. 5th International Conference on Technology and Automation.

[16] Salonen, K., Raduly-Baka, Cs., Nevalainen, O., 2003. A note on the tool switching problem of a flexible machine, Special Issue of Computers and Industrial Engineering, Selected papers from 32nd ICC&IE in Limerick.

[17] Song, C.Y., Hwang, H., 2002. Optimal tooling policy for a tool switching problem of a flexible machine with automatic tool transporter, International journal of production research, 40, 873-883.

[18] Stecke, K. E., 1983. Formulation and solution of nonlinear integer production planning problems for flexible manufacturing systems. Management Science 29.3, 273-288.

[19] Tang, C.S., Denardo E.V., 1988. Models arising from a flexible manufacturing machine, Part I: Minimization of the number of tool switches, Operations Research, Vol. 36, No. 5, pp. 767-777.

[20] Tzur, M., Altman, A., 2004. Minimization of tool switches for a flexible manufacturing machine with slot assignment of different sizes, IIE Transactions, Vol 36, pp. 95-110.

[21] Zhou, B.H., Xi, L.F., Cao, Y.S., 2005. A beam-search-based algorithm for the tool switching problem on a flexible machine. The International Journal of Advanced Manufacturing Technology 25.9-10, 876-882.

4. Appendix. Proof of Lemma 3

Proof. The reduction of the Decision-MMS problem to the Decision-MTS-C problem follows a similar logic to the earlier reductions. Here we must check that, for some fixed C, processing times p_k can be assigned to the machines so that each machine completes in no more that C time. Observe that this reduction relies on the fact that Minimum Makespan Scheduling is strongly NP-hard, and therefore its unary representation is still NP-complete.

Given an arbitrary instance I of the Decision-MMS problem, construct a Decision-MTS-C problem instance in the following way:

- Set the reel switching cost to $t_c = 2n + 1$ and the module switching cost $t_m = 1$.

- For each processing time p_k of I, define a set of basic component types $T_k = \{t_{k1}, t_{k2}, ..., t_{kp_k}\}$, $|T_k| = p_k$. This transformation is linear when unary representation of the MMS problem is used (per the definition of strongly NP-complete).

- Define an *extra* set of component types X, $|X| = C$. These component types are different from the ones in the basic sets T_k.

- Let the number of online modules be the number of machines $F = m$, and the capacity of a module be C (the makespan of the Decision-MMS).

- A machine of the MMS problem corresponds to a module of the MTS problem. Assigning jobs k in MMS to a machine corresponds assigning reels of the set T_k to a module.
- Let $S = \sum_{k=1}^{n} p_k$. Clearly, $S \leq m \cdot C$ otherwise the problem is not feasible.
- Define another set of component types Z containing $n \cdot C - S$ different component types.
- If a machine (module) is not fully used to its capacity C, components of Z are used to fill the capacity requirement up to nC for each job of the MTS problem.
- The rest of the Decision-MTS-C problem instance is constructed in the same way as in the Bin-Packing reduction.

Clearly, as in the Bin-Packing reduction, a YES-instance of Decision-MTS-C problem can be achieved if and only if, each reel set T_k is entirely contained in a single module. This is because only in such assignment it is possible to have a single module switched between two consecutive jobs. Each module may contain some extra components of Z, and the capacity of a module may not exceed C.

The reduction follows the same logic as earlier reductions. From a YES-instance of Decision-MTS-C problem we get an assignment of reel sets T_k to modules M_i, where each reel set is feasibly assigned to a single module. This corresponds to an assignment of job k in MMS to a single machine, where the machine capacity C is not exceeded.

Conversely, a YES-instance in MMS can easily be converted into a YES-instance of MTS-C problem, requiring only one module swap between two consecutive jobs, by assigning a reel set T_k to a module M_i following the job/machine assignments obtained in MMS.

As in the previous cases, the proof can also be extended to the case where no extra component reel copies are available. A job J_y taking up only one module is inserted between each job J_a, J_k and J_k, J_a, forcing the content of the corresponding module offline, and making the necessary components available offline to construct a new module at no delay.

$\square$