Algorithms for Sheet Metal Nesting

Jeffrey W. Herrmann, David R. Delalio

Abstract—**This paper discusses the problem of minimizing the cost of sheet metal punching when nesting (batching) orders. Although the problem is NP-complete, the solution to a linear programming relaxation yields an efficient heuristic. This paper analyzes the heuristic's worst-case performance and discusses experimental results that demonstrate its ability to find good solutions across a range of cost parameters and problem sizes.**

Keywords—**Sheet metal punching, Batch processing, Manufacturing, Optimization.**

I. Introduction

SHEET metal is a popular material for many types of products. Forming a part from sheet metal includes preparatory and finishing operations such as blanking, deburring, and bending. A blank is an intermediate part that is cut from sheet metal and transformed by other operations to form the finished part. Many manufacturers use numerically controlled (NC) punch presses for creating blanks. Such a press is an extremely flexible machine and can form a wide variety of shapes from different types of sheet metal. It forms a part by using different tools to punch holes in the metal sheet. The metal sheet is clamped to a table that the machine can move. By moving the sheet and cutting holes next to each other (see Figure 1), the press forms the part's outline, leaving small pieces to hold the part in the sheet. During this process, the press follows the instructions listed in a NC program. These instructions tell the press how to punch one or more copies of the part from a sheet sheared to the correct size.

To complete a job (or order) that requires many parts, an operator first loads the NC program into the machine, arranges the clamps, and places the needed tools into the tool carousel, if they are not already in place. Then the operator loads the first sheared sheet and runs the NC program, which punches some parts from that sheet. Then the operator unloads the punched sheet, loads a new sheet, and repeats the process until the order is finished. Sheared sheets are cost-effective when producing large quantities because less material is wasted.

In the modern manufacturing environment, many companies are using smaller lot sizes and trying to decrease costs wherever possible. Nesting is a potential solution for sheet metal punching.

Definition 1: Nesting combines multiple orders that require the same type of sheet metal so that a punch press can complete all of the parts in these orders as one job.

Note that the nested parts must require the same material and the same sheet thickness. Nesting can reduce the time for machine setup and sheet loading and reduce material costs. The operator performs only one setup (instead of two or more). Instead of purchasing specific sheared sheets, nesting uses larger, standard sheets, which cost less per pound because they don't require a shearing operation. Because the sheets are larger, fewer are needed, and the operator spends less time loading sheets. Section II describes nesting in more detail.

Nesting has great cost-saving potential, and we have worked with a manufacturer that is beginning to nest orders regularly. However, it raises some questions. If no orders are nested, it is straightforward to calculate the orders' capacity and material requirements (for example, see [1]). However, nesting the orders requires new methods that model the requirements more accurately. This is especially important when using order release procedures that monitor work-in-process inventory and estimate the machine workloads. Overestimating the workload requirements will starve machines unnecessarily. If these machines are bottlenecks, then this decreases the shop's throughput. (This company is developing such procedures [2, 3].) In addition, the shop needs to monitor requirements for standard metal sheets so that sufficient stock is on-hand when needed. Finally, it is important to calculate accurate costs and to estimate the cost savings, if any, that nesting will bring. Herrmann and Delalio [4] describe models for calculating accurate capacity and material requirements.

Although nesting can save time and money, nesting every order is not necessarily the best policy. When the objective is to minimize setup and material costs, finding the optimal set of orders to nest is not a simple problem.

We will prove that the problem is NP-complete and will present a pseudo-polynomial time dynamic program to find an optimal solution. Although we can use integer programming techniques or the dynamic program to solve the problem optimally, these are not easy to implement in a factory, and an exact algorithm takes too much time to do by hand. Thus, a robust, fast heuristic that is easy to implement would be a very useful tool for dynamic nesting. This paper discusses a linear programming relaxation and derives an efficient heuristic. Finally, we discuss nesting frequency and some related issues.

Various aspects of sheet metal manufacturing have attracted attention in the past. Some writers [5, 6] have described sequencing algorithms that reduce the total punching time due to table movement and tool changes. Sachs [7] introduces the fundamental issues in sheet metal fabrication and classifies sheet metal parts based on the part geometry. Other authors also describe the associated manufacturing processes (for example, see [8]). Previous work on batching includes work on how to batch jobs to create good schedules [9, 10] cutting stock problems that seek to

Jeffrey W. Herrmann is with the Department of Mechanical Engineering and the Institute for Systems Research, University of Maryland. Email: jwh2@eng.umd.edu.

David R. Delalio is with IGEN International, Inc.

minimize waste (starting with Gilmore and Gomory [11]), and multiperiod lot sizing problems that are similar to the joint replenishment problem [12]. Other authors have considered how setup time reductions affect average inventory costs (for example, see [13, 14]). There exist a variety of commercial software packages that can layout parts from multiple orders and generate the necessary NC code. However, we are not aware of any work that considers the problem of forming a batch from a set of orders to minimize setup and material costs.

Section III introduces the notation, formulates the dynamic nesting problem, presents a dynamic programming solution, and discusses a linear programming relaxation. Section IV presents an efficient heuristic and its worst-case error bound. Section V presents some experimental results that demonstrate the heuristic's performance. Section VI concludes the paper and presents some ideas for future work.

II. Dynamic Nesting

This section discusses the context for sheet metal nesting decisions. Although this discussion was motivated by our work with a specific manufacturer, the setting is typical and will apply to other manufacturers and similar manufacturing processes as well.

The manufacturer uses a manufacturing planning and control system that maintains a master production schedule. The planning system uses material requirements planning (MRP) to explode the end-item requirements into work orders for components and subassemblies. Also, the planning system has detailed routings (process plans) that specify the resources required for each product. Thus, production planners can identify, for next week and each week after that, the orders that will require the punch presses in the sheet metal area. (In this setting there are no filler parts available for consuming remaining open areas.)

Because the product mix changes greatly each week, the sheet metal area will use dynamic nesting to nest the orders.

Definition 2: Dynamic nesting periodically considers the specific orders to be punched in the next period and creates customized nests for those orders.

That is, before each week, the production planners will identify and nest the orders that will be released into the shop that week. Orders that require the same NC punch press, the same material type, and the same sheet thickness can form a nest. Then, the production planners will use existing software that arranges the required parts on the minimum number of unsheared sheets, leaving room for the clamping area and leaving room between the parts. This software also creates the required NC program by combining the NC programs for each different part. After the orders are released, the operator loads this NC program onto the punch press and starts punching the nest, loading unsheared sheets and removing punched sheets as required. Punched sheets are sent to a deburring operation, where an employee removes the individual parts from the sheet, removes any burrs from the part, and sorts the parts into the individual workorders for additional processing.

For a simple example, consider Figure 2. Order 1 requires two parts, one on each sheared sheet. Order 2 requires four parts, all on one sheared sheet. Order 3 requires four parts, two on each sheared sheet. All three orders require the same material type and thickness. Thus, they can form a nest. Nesting all three orders creates the nest in the bottom right of the figure. The nest requires two unsheared sheets. If the nest includes only Orders 1 and 2, then the nest requires only one unsheared sheet, as shown in the bottom left of the figure. Order 3 is processed separately.

Dynamic nesting has great cost-saving potential since a nest requires only one setup and fewer, standard sheets. However, finding the optimal nest (which minimizes the setup and material costs) is not a simple problem. The production planners need a tool to help identify good nests. However, they must make this nesting decision before they can generate the actual nest layout (and the associated NC program).

To overcome this obstacle, we simplified the layout problem by considering only the area that each part requires and the total usable area of a sheet. A part's required area, which may have a complex profile, includes the necessary inter-part spacing and any internal area that no other part could possibly use. We assume that parts will fit onto a sheet if the total area is sufficient. If many of the parts were large, then this approximation would be poor, and we would need to consider the part geometry and part layout explicitly in the nesting decision process. However, in practice, the parts were small compared to an unsheared sheet, so the approximation is acceptable. (The largest part area was approximately eleven percent of the usable sheet area.) Note that the nesting decision determines which orders (and parts) the nest should include, but it does not create a layout for the parts on the sheets. If the actual part layout for the nest did require more unsheared sheets than the nesting decision predicted, there would be time to modify the nesting decision. Thus, we used the area approximation for making the nesting decision.

III. Problem Formulation

This section introduces the dynamic nesting problem. This discussion describes a set of orders that need processing on a given machine in a given week.

We limit our analysis to the most significant costs that nesting changes. These are the direct labor for performing machine setups and loading sheets and the cost of sheared and unsheared sheets. The cost for setup labor is determined by the time required and the labor cost (in dollars per hour). Although there is unavoidable variation in setup and sheet loading times, we use the average time in this analysis.

This analysis does not include the time that the machine spends cutting the parts because nesting does not significantly affect the cutting time, which depends upon the perimeter of the parts to be cut, or the time the punch spends moving from one part to another. The machine setup time includes time for tool changes, which are not explicitly considered. As discussed previously, we assume that the number of sheets required for a nest is at least the total area of the nested parts (including the necessary inter-part spacing) divided by the usable sheet area.

A. Notation

For a given time period, there are a set of jobs (orders) to be processed on a given machine. These orders form g disjoint groups $G_1, \ldots, G_g$. The jobs in a group require the same material and same sheet thickness. Thus, they can form a nest. The nesting of each group does not affect the nesting of any other group, so it is possible to decompose the problem into subproblems for each group. Thus, the remainder of this paper will address the problem for one group.

The machine setup time for a non-nested order is u^0. The machine setup time for a nest is u^1. Let D be the direct labor cost rate for setup (and sheet loading) time (in dollars per hour).

An order J_j requires q_j copies of part type i_j. Each part of type i_j has area $A^0(i_j)$, which includes the necessary inter-part spacing. Let $O_j = q_j A^0(i_j)$ be the total area of an order J_j. (We assume that all areas can be expressed as integers in some unit of measurement.)

If not nested, an order J_j requires N_j^0 sheared sheets. The time needed to load each sheared sheet is t_j^0. The cost of each sheet is c_j^0. Let $F_j = N_j^0(c_j^0 + t_j^0 D) + u^0 D$ be the total material and setup cost of not nesting an order J_j.

Now consider the nest for group G. Let $X_j = 1$ if order J_j is in the nest, and $X_j = 0$ otherwise. Let $Y = \max_{j \in G}\{X_j\}$. If there are any nested orders, $Y = 1$. Otherwise, $Y = 0$. Let $S = u^1 D$ be the machine setup cost for a nest. Each unsheared sheet includes clamping area and area that can be used for parts. Let $N(X)$ be the number of sheets required, which depends upon the nesting decision, the total area of the nested orders, and the unsheared sheet's usable area A.

$$N(X) = \left\lceil \frac{\sum_{j \in G} O_j X_j}{A} \right\rceil$$

where $\lceil x \rceil$ is the smallest integer greater than or equal to x.

The time needed to load each unsheared sheet is t^1. The cost of each sheet is c^1. Let $R = c^1 + t^1 D$ be the total material and sheet loading cost of an unsheared sheet.

Then, for a given nesting decision X, the total setup and material cost for the group is the following sum:

$$C(X) = \sum_{j \in G} F_j(1 - X_j) + \left\lceil \frac{\sum_{j \in G} O_j X_j}{A} \right\rceil R + YS$$

For purposes of analysis, identify, for each $j \in G$, the non-negative integer s_j such that $s_j A \leq O_j < (s_j + 1)A$. Then, there exists a value e_j such that $O_j = e_j + s_j A$ and $0 \leq e_j < A$. s_j is the number of whole unsheared sheets that order J_j requires, and e_j is the extra area that the order requires. Then, we can write the dynamic nesting optimization problem as follows:

Minimize

$$C(X) = \sum_{j \in G} F_j + \sum_{j \in G} (s_j R - F_j) X_j + \left\lceil \frac{\sum_{j \in G} e_j X_j}{A} \right\rceil R + YS$$

subject to

$$\begin{aligned} X_j &\leq Y \ \forall\, j \in G \\ X_j &\in \{0,1\} \ \forall j \in G \\ Y &\in \{0,1\} \end{aligned}$$

The decision version of this problem is an NP-complete problem. The proof, which is discussed in the Appendix, uses a transformation from the PARTITION problem. Moreover, this problem remains NP-complete if all of the setup costs are zero or if all of the material costs are zero.

Note especially that, because the objective function is nonlinear, the linear programming relaxation does not yield an optimal solution despite the integer extreme points (see also Section III-C).

B. Dynamic Programming

This section presents a pseudo-polynomial dynamic programming algorithm to solve the nesting problem for each group. Number the orders in the group $1, 2, \ldots, n$. Although the total number of feasible solutions is 2^n, the computational effort of the dynamic program is proportional to nA. Thus, as n increases, the computational effort does not increase exponentially, unlike other enumeration techniques that could be used. The dynamic program presented here is a forward dynamic program that finds the optimal (least-cost) nesting decision. For more information about dynamic programming, see, for example, [15].

The state of the dynamic program is the set of orders that have been considered and the area on the last unsheared sheet. The recursive equations determine, for each state, the cost of the optimal policy that achieves the state. Let $f(j, x)$ be the minimum cost of any nesting decision that has considered the first j orders and requires area x on the last unsheared sheet. (Note that the cost function tallies the cost of the complete unsheared sheets used, which do not otherwise affect the recursion.)

Initialization: $f(0, 0) = 0$. $f(0, A) = S$. $f(0, x) =$ infinity if $0 < x < A$.

Recursion: If $x = 0$, then

$$f(j, x) = f(j-1, x) + F_j$$

In this case, no orders are nested.

If $0 < x \leq e_j$, then

$$f(j, x) = \min\{f(j-1, x) + F_j, f(j-1, A + x - e_j) + (s_j + 1)R\}$$

In this case, either order j was not nested, or order j added its s_j unsheared sheets and the extra area of order j added another unsheared sheet to the nest. In this case, the portion of order j on the new unsheared sheet has area x, and the portion on the last unsheared sheet has area $e_j - x$. Thus, the area on the last unsheared sheet was $A - (e_j - x) = A + x - e_j$ before adding order j to the nest.

If $e_j < x \leq A$, then

$$f(j,x) = \min\{f(j-1,x) + F_j, f(j-1,x-e_j) + s_j R\}$$

In this case, either order j was not nested, or order j added its s_j unsheared sheets but the extra area of order j fit onto the last unsheared sheet of the nest.

Answer:

$$\min_{0 \leq x \leq A} \{f(n,x)\}$$

The complexity of this algorithm is $O(nA)$.

This formulation allows us to identify a special case that can be solved immediately. Note that $(s_j+1)R$ is an upper bound for the cost of nesting order j. If $(s_j+1)R \leq F_j$, then the cost of adding the order to the nest is less than or equal to the cost of excluding it. If $(s_j+1)R \leq F_j \; \forall j$, then nesting no orders is the only solution that could cost less than nesting every order. To decide, compare $\sum_j F_j$ (the cost of nesting no orders) to $S + \lceil \sum_j O_j / A \rceil R$ (the cost of nesting every order). If the first quantity is smaller, then nesting no orders is the optimal solution. Otherwise, nesting every order is an optimal solution.

C. Linear Programming Relaxation

In order to construct an efficient heuristic, we form and solve a linear programming relaxation of the dynamic nesting problem. The variables are now continuous, and the objective function has been made linear.

Minimize

$$C^R(X) = \sum_{j \in G} F_j + \sum_{j \in G} (s_j R + \frac{e_j}{A} R - F_j) X_j + YS$$

subject to

$$\begin{aligned} X_j &\leq Y \; \forall \, j \in G \\ 0 \leq X_j &\leq 1 \; \forall j \in G \\ 0 \leq Y &\leq 1 \end{aligned}$$

This linear program has integer extreme points, and we can determine the optimal solution as follows. Let $T_j = s_j R + \frac{e_j}{A} R - F_j$ be the net cost of nesting order J_j in this linear program. If no $T_j < 0$, then an optimal solution is to nest nothing (all $X_j = 0$ and $Y = 0$).

Otherwise, some $T_j < 0$. Let $W = \{j : T_j < 0\}$. If $\sum_{j \in W} T_j + S \geq 0$, then an optimal solution is to nest nothing. If $\sum_{j \in W} T_j + S < 0$, then the optimal solution is to nest order J_j if and only if $j \in W$ ($X_j = 1$ if and only if $j \in W$ and $Y = 1$). This insight leads to the heuristic described below.

IV. An Efficient Heuristic

This section presents an efficient heuristic for the cost optimization problem. It requires little computational effort and creates cost-effective nests, as we will show. We will also discuss how to implement the heuristic.

A. Description

The heuristic (called "Simple") requires the following steps:

1. Calculate the approximate net cost T_j of nesting order J_j for all $j = 1, \ldots, n$, in group G:

$$T_j = s_j R + \frac{e_j}{A} R - F_j$$

2. Let $W = \{j : T_j < 0\}$.
3. If W is empty or $\sum_{j \in W} T_j + S \geq 0$, then nest nothing. Otherwise, since $\sum_{j \in W} T_j + S < 0$, nest order J_j ($X_j = 1$) if and only if $j \in W$.

The heuristic, which finds an optimal solution to the linear programming relaxation, requires $O(n)$ effort, so it is very efficient. In addition, the heuristic produces near-optimal nesting decisions, as the following result shows.

Theorem 1: If X^o is the solution that the Simple heuristic creates and X^* is an optimal nest for the dynamic nesting problem, then $C(X^o) - C(X^*) < R$.

Proof: For any solution X, $C(X)$ is the cost of the solution. Note the following bound on this cost:

$$C(X) \geq C^R(X) = \sum_{j \in G} F_j + \sum_{j \in G} T_j X_j + YS$$

If all $T_j \geq 0$, then, for any solution X that nests an order, $Y = 1$, and $C(X) > \sum_{j \in G} F_j$, so the optimal solution must be nesting nothing, which the heuristic does, since W will be empty. Thus, $C(X^o) = C(X^*)$.

Now, suppose some $T_j < 0$. $W = \{j : T_j < 0\}$. Define a lower bound B:

$$B = \sum_{j \in G} F_j + \sum_{j \in W} T_j + S$$

B is a lower bound on the cost $C(X)$ of any solution X that nests an order.

Suppose $\sum_{j \in W} T_j + S \geq 0$. Then $B \geq \sum_{j \in G} F_j$, so an optimal solution is to nest nothing, which the heuristic does. Thus, $C(X^o) = C(X^*)$.

Suppose $\sum_{j \in W} T_j + S < 0$. Then $B < \sum_{j \in G} F_j$. Substituting the formula for T_j yields

$$B = \sum_{j \in G} F_j + \sum_{j \in W} (s_j R - F_j) + \frac{\sum_{j \in W} e_j}{A} R + S$$

Let X^o be the heuristic solution: $X_j = 1$ if and only if $j \in W$. $Y = 1$.

$$C(X^o) = \sum_{j \in G} F_j + \sum_{j \in W} (s_j R - F_j) + \left\lceil \frac{\sum_{j \in W} e_j}{A} \right\rceil R + S$$

Because $\lceil x \rceil R < (x+1)R$,

$$C(X^o) < B + R$$

If X^* nests an order, then $B \leq C(X^*)$, so

$$C(X^o) < C(X^*) + R$$

If X^* nests nothing, $C(X^*) = \sum_{j \in G} F_j$, and $B < \sum_{j \in G} F_j$ implies that $B < C(X^*)$. Thus,

$$C(X^o) < C(X^*) + R$$

So, in the worst case, $C(X^o) - C(X^*) < R$. ■

This worst case can be achieved. Consider the following instance, which has one order: Given R and A, select parameters e and d such that $0 < e < A$ and $0 < d < R(1 - e/A)$. Let $F_1 = eR/A + 2d$. Let $S = d$ and $O_1 = e$ (thus $s_1 = 0$ and $e_1 = e$). Then, $T_1 = -2d < 0$, and $T_1 + S = -d < 0$. The heuristic forms a nest. X^o, the optimal solution to the linear programming relaxation, is $X_1^o = 1$ and $Y^o = 1$. $C^R(X^o) = eR/A + d$, and $C(X^o) = R + d$. However, by the constraints on d, $C(X^o) - F_1 = R - eR/A - d > 0$, so the optimal nest X^* is to nest nothing ($X_1^* = 0$, $Y^* = 0$). $C(X^*) = F_1$. As e and d approach zero, $C(X^o)$ approaches R, $C(X^*)$ approaches zero, and $C(X^o) - C(X^*)$ approaches R.

B. Implementation

To implement dynamic nesting the production planners first need to sort the orders to be released each time period into groups based on material type and thickness. The next step is to use the Simple heuristic to make a nesting decision for each group.

To implement the Simple heuristic, the production planners need the following information:

- For each order, F_j, the total setup and material cost of order J_j if it is not nested.
- For each order, O_j, the total area that order J_j requires.
- A, the usable area of an unsheared sheet.
- R, the material and sheet loading cost of an unsheared sheet.
- S, the machine setup cost for a nest.

For an order J_j, calculating F_j requires the following steps: determine the number of sheared sheets (the number of parts in the order divided by the number of parts per sheet), multiply by the material and sheet loading cost of a sheared sheet, and add the machine setup cost for an unnested order. Calculating O_j requires one to multiply the number of parts in the order by the required area (including unusable space and interpart spacing) per part.

After the data is collected, the heuristic can proceed as follows:

1. Calculate the approximate net cost T_j of nesting order J_j for all orders in the group.

$$T_j = \frac{O_j}{A} R - F_j$$

2. Let $W = \{j : T_j < 0\}$.
3. If W is empty or $\sum_{j \in W} T_j + S \geq 0$, then nest nothing. Otherwise, nest the orders in W.

V. Experiments

This section presents some experiments conducted on typical data that we encountered in industry. The manufacturing firm provided the necessary part data. To protect proprietary information, we do not include in this report all of the data that we collected. In addition, all cost values are scaled.

First we identified 11 part types as typical parts. Table I presents some information about the part types and the sheared sheets.

The load time t_j^0 for all sheared sheets is 0.014 hours. The machine setup time for a non-nested order is $u^0 = 0.5$ hours. The part perimeters ranged from 30 cm to 216 cm.

The unsheared sheets (which measure 122 cm by 244 cm) are much larger than the sheared sheets. Their usable area is $A = 26\,381$ cm^2. The time needed to load each unsheared sheet is $t^1 = 0.021$ hours. The machine setup time for a nest is $u^1 = 1.25$ hours.

To compare the average performance of the heuristic, we created a number of problem sets, each defined by a combination of problem parameters, and generated ten random instances within each set. The problem parameters included the usable area of an unsheared sheet, the cost of an unsheared sheet, the cost to load an unsheared sheet, the cost to setup a nest, the cost to load a sheared sheet, the cost to setup an order, the number of orders in the instance, the minimum parts per order, and the maximum parts per order.

For each instance, the instance generation procedure created the appropriate number of orders. For each order, the procedure randomly selected one of the eleven part types and the number of parts in the order. Then, using the cost parameters, it calculated the non-nested cost F_j, the total area O_j, the number of whole unsheared sheets s_j, and the extra area e_j. A complete instance includes this data for each order and the nest setup cost S, the usable unsheared sheet area A, and the unsheared sheet cost R.

Table II lists the combinations of parameters used. Most parameters were unchanged as we were particularly interested in the heuristic's performance as the unsheared sheet cost changed and as orders became smaller but more numerous.

Table III lists the average relative cost of the heuristic on each problem set. For comparison's sake, we also measured the cost of nesting nothing and the cost of nesting everything. For each instance, the relative cost of each solution is the cost of the solution divided by the optimal cost (determined by the dynamic program) for that instance. Then, the relative cost is the averaged over the ten instances in the problem set. We did not measure CPU times as the heuristic is a polynomial-time algorithm and requires very little computational effort. Our primary concern was the quality of the solutions.

We can see from these results that the heuristic is able to find good solutions. The relative cost of extreme solutions (nesting nothing or nesting everything) changes greatly as the unsheared sheet cost changes. However, the relative cost of the heuristic's solutions remains low in all scenarios.

The discussion so far has concerned a given set of orders, and which should be nested. However, the production planners also need to consider how often to form nests. Nesting more frequently would reduce the nest size and allow orders to be released to the shop more frequently, reducing work-in-process inventory. However, it might increase costs. Nesting less frequently, which would create larger nests, should further reduce setup and material costs, although it will increase work-in-process inventory. Thus, we investigated how the nesting frequency affects the total setup and material costs.

To do this, we created, for each of the six problem sets, ten randomly generated instances of 256 orders. We then compared batch sizes of 2, 4, 8, 16, 32, 128, and 256 orders. When the batch size equals b, the instance has $y = 256/b$ batches. We used the Simple heuristic to nest the orders in each batch. (Of course, for some batches, the heuristic may decide to nest nothing.) The total setup and material cost was summed over all y batches. This total cost, relative to the total cost when $b = 2$, was averaged over the ten instances in the problem set. Table IV summarizes the results. Note that increasing batch size corresponds to decreasing nesting frequency.

These results show that changing the dynamic nesting frequency can significantly change the material and setup costs. This change is greatest when the unsheared material cost is low, as in Set A and Set D, because nests are always preferable. Thus, less frequent nesting decreases costs because there are fewer nest setups. When the cost of unsheared material is moderate, as in Set B and Set E, smaller batches are less likely to be nested because the cost of the nest setup outweighs the benefits of nesting the small set of orders. In this case, nesting less frequently reduces nest setup costs, but the reduction is smaller. When unsheared material is expensive, as in Set C and Set F, small batches are rarely nested. Nesting less frequently increases the batch size, and the heuristic nests a few orders, which increases costs due to the expensive unsheared sheets. When the batches are very large, more orders are nested, and the setup costs decrease. Also, note that when orders have fewer parts (Sets D, E, and F), more orders will fit onto an unsheared sheet. Thus, larger nests eliminate more unnested setups, which reduces costs more (compared to Sets A, B, and C).

Thus, we conclude that, when unsheared material is inexpensive and nests are preferred, changing the nesting frequency can significantly affect setup and material costs. However, when unsheared material is expensive and nests are not desirable, changing the nesting frequency does not affect setup and material costs as much.

VI. Conclusions

Dynamic nesting has great cost-saving potential for NC punch presses that create sheet metal blanks. It reduces setups and material requirements. However, finding the most cost-effective nest is an NP-complete problem. This paper addresses that problem and presents a robust, efficient heuristic that is easy to implement.

Specifically, we constructed an integer program that models the nesting decision for a group of orders that require the same material type and thickness. The objective is to minimize the total cost of setup and material. After analyzing a linear programming relaxation, we derived an efficient heuristic and presented a worst-case error bound. Experimental testing based on industry data demonstrated the heuristic's effectiveness over a range of cost parameters and order sizes. In addition, testing illustrated how changing the nesting frequency changes the total setup and material costs.

When beginning to implement dynamic nesting, nesting all orders may seem like a good strategy. Unfortunately, it could lead to poor solutions. The heuristic presented here, however, is guaranteed to find near-optimal solutions. The heuristic's minimal data and computational requirements allow it to run independent of other software packages such as nesting software or optimization engines. Because it is a robust, fast algorithm that is easy to implement, the heuristic should be a very useful tool for dynamic nesting.

Our analysis simplified the layout problem by considering only the required area for each part and the total usable area of a sheet. For making the nesting decision, which precedes part layout, we assume that parts will fit onto a sheet if the total area is sufficient. If many of the parts were large, then this approximation would be poor, and we would need to consider the part geometry and part layout explicitly in the nesting decision process. However, in practice, the parts were small compared to an unsheared sheet, so the approximation is acceptable. In addition, if creating the actual part layout for a nest did require more unsheared sheets than expected, there would be time to modify the nesting decision.

Appendix

This section considers the problem of finding the least expensive nest for a group. Let us formally state the decision version of the nesting optimization problem:

NESTING. Instance: A set G of orders, with costs as given in Section III, and a cost constraint $\mathcal{C}$.

Question: Is there a nest X such that the total cost $C(X)$ is less than or equal to $\mathcal{C}$?

$$C(X) = \sum_{j \in G} F_j(1 - X_j) + \left\lceil \frac{\sum_{j \in G} O_j X_j}{A} \right\rceil R + YS$$

Theorem 2: NESTING is NP-complete.

Proof: To show that this problem is NP-complete, we define a transformation from the NP-complete problem PARTITION.

PARTITION. Instance: A finite set $A = \{d_1, d_2, \ldots, d_n\}$, where each d_i is a positive integer.

Question: Is there a subset $S \subset \{1, 2, \ldots, n\}$ that partitions the set into two equal subsets?

$$\sum_{i \in S} d_i = \sum_{i \notin S} d_i$$

Given any instance of PARTITION, we can construct (in polynomial time) an instance of NESTING and show that there is a solution to PARTITION if and only if there is a solution to NESTING. This will show that NESTING is NP-complete.

For an instance of PARTITION, let $B = \frac{1}{2}\sum_{i\in A} d_i$. Thus, there is a solution to PARTITION if and only if there is a subset S such that $\sum_{i\in S} d_i = B$. Now, construct the following instance of NESTING:

Let $\mathcal{C} = 3B$. Let $G = \{0, 1, 2, \ldots, n\}$. Let $u^0 = 0$. For all orders, let $t_j^0 = 0$. Let $u^1 = 0$ and $t^1 = 0$. Let $c^1 = 2B$ and $A = 3B$. Let $D = 0$. Thus, $S = 0$.

For each order $j > 0$, let $A(i_j) = 1$, $q_j = d_j$, $N_j^0 = d_j$, and $c_j^0 = 1$. Thus, $O_j = d_j$ and $F_j = d_j$, for $j > 0$. Note $d_j < A$, so $s_j = 0$ and $e_j = d_j$. These orders have many small parts, and a sheared sheet holds one part.

For order 0, let $A(i_0) = 2B$, $q_0 = 1$, $N_0^0 = 1$, and $c_0^0 = 2B$. Thus, $O_0 = 2B$ and $F_0 = 2B$. This order has one large part and requires one expensive sheared sheet.

Substitute these values into the equations for $N(X)$ and $C(X)$:

$$\begin{aligned} N(X) &= \left\lceil \frac{2BX_0 + \sum_{j>0} d_j X_j}{3B} \right\rceil \\ C(X) &= 2B(1 - X_0) + \sum_{j>0} d_j(1 - X_j) + N(X)2B \end{aligned}$$

If there is a solution to PARTITION, then there exists a partition S such that $\sum_{i\in S} d_i = B$. Form a solution X to NESTING as follows: nest order 0 and those orders $j \in S$. Don't nest the other orders. That is, $Y = 1$ and $X_0 = 1$; for $j > 0$, $X_j = 1$ if and only if $j \in S$.

$$\begin{aligned} N(X) &= \left\lceil \frac{2B + \sum_{j\in S} d_j}{3B} \right\rceil = 1 \\ C(X) &= 0 + \sum_{j\notin S} d_j + 2B = 3B \\ &= \mathcal{C} \end{aligned}$$

If there is no solution to PARTITION, then, for any subset S, $\sum_{j\in S} d_j \neq B$. Now, we need to show that the cost of any nest is greater than $\mathcal{C}$, so there is no solution to NESTING. For any nest, let $S = \{j > 0 : X_j = 1\}$. There are four cases that we must consider. First, suppose $Y = 1$, $X_0 = 1$, and $B < \sum_{j\in S} d_j \leq 2B$. This nest will require two unsheared sheets and the total cost will exceed $\mathcal{C}$:

$$N(X) = \left\lceil \frac{2B + \sum_{j\in S} d_j}{3B} \right\rceil = 2$$

$$C(X) = 0 + \sum_{j\notin S} d_j + 2 \times 2B \geq 4B > \mathcal{C}$$

Second, suppose $Y = 1$, $X_0 = 1$, and $\sum_{j\in S} d_j < B$. Thus, $\sum_{j\notin S} d_j > B$. This nest will require one unsheared sheet, but the total cost still exceeds $\mathcal{C}$:

$$C(X) = 0 + \sum_{j\notin S} d_j + 2B > 3B = \mathcal{C}$$

Third, suppose $Y = 1$, $X_0 = 0$, and $0 < \sum_{j\in S} d_j \leq 2B$. This nest will require one unsheared sheet, and the total cost will exceed $\mathcal{C}$:

$$N(X) = \left\lceil \frac{\sum_{j\in S} d_j}{3B} \right\rceil = 1$$

$$C(X) = 2B + \sum_{j\notin S} d_j + 2B \geq 4B > \mathcal{C}$$

Finally, suppose no orders are nested: $Y = 0$, $X_0 = 0$, and $\sum_{j\in S} d_j = 0$. Thus, $\sum_{j\notin S} d_j = 2B$. This solution will require no unsheared sheets, but the total cost will exceed $\mathcal{C}$:

$$C(X) = 2B + \sum_{j\notin S} d_j + 0 = 4B > \mathcal{C}$$

This completes the proof, for we have shown that there is no solution to NESTING. ■

Note that this proof shows that NESTING is NP-complete even if all setup costs are zero. This problem remains NP-complete if the setup costs are positive but the material costs are zero. To prove this, construct the following transformation: Let $G = \{0, 1, 2, \ldots, n\}$. For each order $j > 0$, let $A(i_j) = 1$, $q_j = d_j$, $N_j^0 = d_j$, $c_j^0 = 0$, and $t_j^0 = 1$. For order 0, let $A(i_0) = 2B$, $q_0 = 1$, $N_0^0 = 1$, $c_0^0 = 2B$, and $t_0^0 = 2B$. Let $c^1 = 0$, $t^1 = 2B$, and $A = 3B$. Let $u^0 = 0$ and $u^1 = 0$. Let $D = 1$. $S = 0$. Let $\mathcal{C} = 3B$.

This yields the same equations for $N(X)$ and $C(X)$:

$$\begin{aligned} N(X) &= \left\lceil \frac{2BX_0 + \sum_{j>0} d_j X_j}{3B} \right\rceil \\ C(X) &= 2B(1 - X_0) + \sum_{j>0} d_j(1 - X_j) + N(X)2B \end{aligned}$$

Thus, this problem has a solution ($C(X) \leq \mathcal{C}$) if and only if PARTITION has a solution.

References

[1] T.E. Vollmann, W.L. Berry, and D.C. Whybark, *Manufacturing Planning and Control Systems*, 4th ed. New York: Irwin/McGraw-Hill, 1997.

[2] M.C. Fu, J.W. Herrmann, and M. Narayanaswamy, "Setting thresholds for periodic order release," *Journal of Intelligent Manufacturing*, vol. 8, pp. 369–383, 1997.

[3] J.W. Herrmann, I. Minis, M. Narayanaswamy, and P. Wolff, "Work order release in job shops," in *Proceedings of the 4th IEEE Mediterranean Symposium on New Directions in Control and Automation*, 1996, pp. 733–736.

[4] J.W. Herrmann and D.R. Delalio, "Evaluating Sheet Metal Nesting Decisions," Institute for Systems Research, University of Maryland, College Park, MD, Technical Report 98-8, 1998.

[5] R.A. Walas and R.G. Askin, "An algorithm for NC turret punch press tool location and hit sequencing," *IIE Transactions*, vol. 16, pp. 280–285, 1984.

[6] J.A. Svestka, "Imposing precedence constraints on NC punch press sequences," *International Journal of Production Research*, vol. 28, pp. 2309–2319, 1990.

[7] G. Sachs, *Principles and Methods of Sheet-metal Fabricating*, 2nd ed. New York: Reinhold Publishing Corporation, 1966.

[8] E.P. DeGarmo, J.T. Black, and R.A. Kohser, *Materials and Processes in Manufacturing*, 8th ed. Upper Saddle River, NJ: Prentice Hall, 1997.

[9] C.-Y. Lee, S.D. Liman, and A. Wirakusumah, "Product batching and batch sequencing for NC punch presses," *International Journal of Production Research*, vol. 31, pp. 1143–1156, 1993.

[10] K.G. Kempf, R. Uzsoy, and C.-S. Wang, "Scheduling a single batch processing machine with secondary resource constraints," *Journal of Manufacturing Systems*, vol. 17, pp. 37-51, 1998.

[11] P.C. Gilmore, and R.E. Gomory, "Controlling pattern changes in one dimensional trim problems," *Operations Research*, vol. 13, pp. 94–120, 1965.

[12] C.L. Stowers and U.S. Palekar, "Lot sizing problems with strong set-up interactions," *IIE Transactions*, vol. 29, pp. 167–179, 1997.

[13] J. Trevino, B.J. Hurley, and W. Friedrich, "A mathematical model for the economic justification of setup time reduction," *International Journal of Production Research*, vol. 31, pp. 191–202, 1993.

[14] E.A. DuPlaga, C.K. Hahn, and C.A. Watts, "Evaluating capacity change and setup time reduction in a capacity-constrained, joint lot sizing situation," *International Journal of Production Research*, vol. 34, pp. 1859–1873, 1996.

[15] F.S. Hillier and G.J. Lieberman, *Introduction to Operations Research*. New York: McGraw-Hill, 1990.

Jeffrey W. Herrmann is an assistant professor at the University of Maryland, where he holds a joint appointment with the Department of Mechanical Engineering and the Institute for Systems Research. He is the director of the Computer Integrated Manufacturing Laboratory. He is a member of INFORMS and ASME.

He earned his B.S. in applied mathematics from Georgia Institute of Technology. As a National Science Foundation Graduate Research Fellow from 1990 to 1993, he received his Ph.D. in industrial and systems engineering from the University of Florida. His dissertation investigated production scheduling problems motivated by semiconductor manufacturing. He held a post-doctoral research position in the Institute for Systems Research from 1993 to 1995.

His research interests include the design and control of manufacturing systems and the integration of product design and manufacturing system design.

David R. Delalio works at IGEN Incorporated.