Sequencing Bitvectors with Distance Constraints

R. Nickel*, W. Hochstättler†

8. July 2003

Abstract

Motivated by an application in the automobile industry, we present results on the following problem:

Given a finite multi-set $F = \{b_1, ..., b_1, b_2, ..., b_2, ..., b_\alpha, ..., b_\alpha\}$ of β-dimensional 0-1-vectors and bounds $\underline{m}_j, \overline{m}_j \in \mathbb{N}$ with $\underline{m}_j < \overline{m}_j$ $(j = 1..\beta)$. Find a sequencing $b_{i_1}, b_{i_2}, ..., b_{i_n}$ of the vectors of F such that in each component j there are at least $\underline{m}_j$ and at most $\overline{m}_j$ zeroes between two ones.

We show that this problem is $\mathcal{NP}$-hard even for simplified cases but becomes polynomial if we bound β, $\underline{m}_j$ and $\overline{m}_j \quad \forall j = 1..\beta$. After that we present a simple heuristic procedure and extend it to further types of constraints appearing in automobile industry.

1 Introduction

We focus on a problem that arises in automobile industry when painted car bodies are put on an assembly line to build in all options such as engines, transmissions, sun-roofs, etc. Different bodies may have different options to build in, therefore they are grouped in body types b_i with a quantity q_i (for $i = 1..\alpha$ if we have α such types). For each option there is a team at the assembly line to build in this part which takes a known amount of time. During that time no car with the same option should enter the line to avoid conveyor stopping. Furthermore each option should appear as regular as possible in the sequence to minimize idle-time of the teams.

Since cars are put on the line in a regular manner (say one car per minute) we can simplify the problem to sequencing the cars such that between two cars with the option j at least $\underline{m}_j$ and at most $\overline{m}_j$ other cars without option j are sequenced where $\underline{m}_j$ is the time to build in option j and $\overline{m}_j$ some upper bound to avoid longer idle-times of the team for option j. In this way each body type b_i can be simplified to a bitvector of size β if there are β options, having a one in component j if the car body has option j and zero otherwise. This yields a more formal description of the problem.

Distance-Constraint-Bitvector-Sequencing (DCBS):

- Given a finite multiset $F = \{b_1, ..., b_1, b_2, ..., b_2, ..., b_\alpha, ..., b_\alpha\}$ of β-dimensional 0-1-vectors and bounds $\underline{m}_j, \overline{m}_j \in \mathbb{N}$ with $\underline{m}_j < \overline{m}_j$ $(j = 1..\beta)$.
- Find a sequencing $b_{i_1}, b_{i_2}, ..., b_{i_n}$ of the vectors of F such that in each component j there are at least $\underline{m}_j$ and at most $\overline{m}_j$ zeroes between two ones.

*nickel@math.tu-cottbus.de
†hochstaettler@math.tu-cottbus.de

After the problem is formulated as a mixed integer constrained satisfaction problem we focus on the complexity of this problem and therefore formulate it as a decision problem: "Is there such a sequencing?". We will prove $\mathcal{NP}$-completeness in the general case and two simplified cases. Furthermore we present a dynamic program to prove the polynomiality of the problem when β as well as all upper bounds $\overline{m}_j$ can be assumed to be bounded by a fixed constant.

2 A Mixed Integer Program

Let n be the length of the sequence indexed by k and α the number of different bitvectors indexed by i. We introduce decision variables x_{ik} which will be one if and only if b_i is sequenced at position k. If F contains α different bitvectors $b_i \quad (i = 1..\alpha)$ we assign to each b_i it's quantity N_i. Now we can formulate this problem in terms of mixed integer constraint satisfaction:

$$\sum_{i=1}^{\alpha} x_{ik} = 1 \qquad \forall k = 1..n \tag{1}$$

$$\sum_{k=1}^{n} x_{ik} = N_i \qquad \forall i = 1..\alpha \tag{2}$$

$$X_{j,0} = 0 \qquad \forall j = 1..\beta \tag{3}$$

$$X_{j,k} = X_{j,k-1} + \sum_{i=1}^{P} x_{ik} b_{ij} \qquad \forall j = 1..\beta \quad \forall k = 1..n \tag{4}$$

$$X_{j,k} - X_{j,k-\underline{m}_j} \leq 1 \qquad \forall j = 1..\beta \quad \forall k = \underline{m}_j + 1..n \tag{5}$$

$$X_{j,k} - X_{j,k-\overline{m}_j} \geq 2 \qquad \forall j = 1..\beta \quad \forall k = \overline{m}_j + 1..n \tag{6}$$

$$x_{ik} \in \{0, 1\}$$

Equations (1) and (2) make sure that at each position k exactly one vector is sequenced and that each vector b_i appears exactly N_i times in the sequence. The other two equations simply count the occurance of each commodity j up to position k. (5) and (6) are the distance constraints for the problem. In sum we now have a constraint satisfaction problem with about $n+\alpha+3\beta n$ constraints and $\alpha n+\beta n$ variables. Mixed integer constraint satisfaction is known to be $\mathcal{NP}$-hard, therefore we now focus on the complexity of that problem and it's simplifications.

3 Complexity Results

We will analyse (DCBS), prove it's $\mathcal{NP}$-completeness and find simplifications that belong to $\mathcal{P}$. The combinatorial problems used for reduction are taken from [3].

3.1 $\mathcal{NP}$-complete cases

3.1.1 The general case

We give a pseudopolynomial reduction of (DCBS) from the Three-Partition-Problem (3P) which is known to be $\mathcal{NP}$-complete in the strong sense:

- Given $A = \{a_1, ..., a_{3m}\}$ and $b \in \mathbb{Z}$ with $\frac{b}{4} < a_j < \frac{b}{2} \quad \forall j = 1..3m$ and $\sum_{j=1}^{3m} a_j = bm$.
- Is there a partitioning $A_1 \dot{\cup} ... \dot{\cup} A_m = A$ of A with $\sum_{a \in A_i} a = b \quad \forall i = 1..m$?

Given an instance of (3P). Without loss of generality assume b to be a multiple of 4. Otherwise multiplying each number a_j and b yields the same. An instance of (DCBS) is derived in the following way:

- F contains $(2b+6)\,m+1$ vectors with $3m+4$ components:
  - $3m$ vectors represent the a_j and have a 1 in their j^{th} and $(3m+1)^{st}$ component $(j = 1..m)$ and a zero otherwise. These vectors are called *element vectors* (e_j).
  - $m+1$ vectors have a 1 in each component except the $(3m+1)^{st}$ and are called *partition vectors* (p) because they have to separate the partitions.
  - $2m$ vectors have a 1 in each component except the last and the $(3m+1)^{st}$ and are called *separation vectors* (s) because they have to separate two element vectors from each other.
  - The remaining $2bm$ all-zero-vectors only fill the gaps between the other vectors and are therefore called *gap vectors* (g).
- Each component is assigned a lower and upper bound that forces the vectors on the correct positions in a sequence.
  - For the partition vectors to build up correct partitions the $(3m+4)^{th}$ component is assigned the bounds $\underline{m}_{3m+4} = 2b+5$ and $\overline{m}_{3m+4} = 2b+6$.
  - The $(3m+3)^{rd}$ component aims to sequence exactly two separation vectors between two partition vectors. Hence let $\underline{m}_{3m+3} = \frac{b}{2} - 1$ and $\overline{m}_{3m+3} = b-1$.
  - The bounds of component $3m+2$ make sure that all non-zero vectors are sequenced with a safety distance. Let therefore $\underline{m}_{3m+2} = \frac{b}{4} + 1$ and $\overline{m}_{3m+2} = \frac{b}{2} - 1$.
  - For two element vectors not to follow directly after each other (i.e. they are separated by gap vectors only) the $(3m+1)^{st}$ component is sequenced with the bounds $\underline{m}_{3m+1} = \frac{b}{2} + 3$ and $\overline{m}_{3m+1} = b-1$.
  - An element vector e_j is forced to have a distance of a_j to it's neighboring partition or separation vector respectively by the bounds $\underline{m}_j = a_j$ and $\overline{m}_j = b-1$.

Lemma 1. *Given a feasible sequencing of F the following statements hold:*

1. *Partition vectors are sequenced with an exact distance of $2b+5$. Furthermore the sequence starts and ends with a partition vector.*
2. *Exactly two separation vectors are sequenced between two partition vectors.*
3. *Between two vectors of the kind separation or partition vector exactly two element vectors are sequenced. Ignoring the gap vectors the sequence looks as follows:*

$$p, e_{i_1}, s, e_{i_2}, s, e_{i_3}, p, e_{i_4}, s, e_{i_5}, s, e_{i_6}, p, ...$$

4. *An element vector is enclosed by two a_j-blocks of gap vectors.*

Proof. If the vectors are sequenced such that no distance constraint $\underline{m}_j$ or $\overline{m}_j$ for $j = 1..3m+4$ is violated we can conclude:

1. Component $3m+4$ makes sure that two partition vectors are sequenced with a distance of at least $2b+5$. Since there are $m+1$ of those vectors and the sequence has a length of $(2b+5)\,m+m+1$, there's only one possibility.

2. Assume that only one separation vector s is sequenced between two partition vectors. Then s has a distance of more than $\left\lfloor \frac{2b+5}{2} \right\rfloor$ to at least one of the neighboring partition vectors which is a contradiction to the upper bound $\overline{m}_{3m+3} = b-1$. The partition vectors induce m partitions in each of which at least two of $2m$ separation vectors are sequenced. This proves the assumption.

3. Per partition (the space between two partition vectors) exist three intervalls induced by two separation vectors in which the element vectors have to be put. In sum there are $3m$ of these intervalls.
   At first we show that one partition contains exactly three element vectors. In case one partition contains 4 element vectors at least two of those follow directly after each other (separated only by gap vectors) with a distance of at least $\frac{b}{2}+3$ because of component $3m+1$. The remaining five non-zero vectors (3 element vectors and 2 separation vectors) are sequenced with a minimum distance of $\frac{b}{4}+1$ due to component $3m+2$. Furthermore we have to add at least once a minimum distance of $\frac{b}{4}+1$ to the pair of element vectors. It follows that in one partition safety distances of $6\left(\frac{b}{4}+1\right)+\frac{b}{2}+3 = 2b+9 > 2b+5$ have to be kept which leads to a contradiction to the length of one partition.
   Now it is clear that all e_j are uniformly distributed over the partitions (i.e. each partition contains 3 element vectors and 2 separation vectors). We show by contradiction that a partition (ignoring the gap vectors) is of the form $p, e_{i_1}, s, e_{i_2}, s, e_{i_3}, p$ for if two element vectors follow directly after each other (minimum distance of $\frac{b}{2}+3$) two vectors of kind partition or separation vector must follow after each other dirctly, too (minimum distance of $\frac{b}{2}-1$). A partition then would look like: $p, e_{i_1}, e_{i_2}, s, s, e_{i_3}, p$. Between two vectors of differend kind component $3m+2$ ensures the sequencing of at least $\frac{b}{4}+1$ gap vectors. In sum we than have at least $4\left(\frac{b}{4}+1\right)+\frac{b}{2}+3+\frac{b}{2}-1 = 2b+6 > 2b+5$ which contradicts the size of a partition.
   Now we have shown that each partition is of the form $p, e_{i_1}, s, e_{i_2}, s, e_{i_3}, p$.

4. There exist $2bm$ gap vectors $2b$ of which have to be sequenced in each partition ($2b+5$ vectors fit into each partition, two separation vectors, three element vectors and $2bm$ gap vectors). Each element vector e_j has a minimum distance of a_j (j^{th} component) to it's left and right non-zero neighbor (partition or separation vector). Since $\sum_{j=1}^{3m} a_j = bm$ we need $2bm$ gap vectors to fill theese gaps. Considering the overall length of the sequence each e_j is enclosed by two a_j-blocks of gap vectors. If the element vectors $e_{i_1}, e_{i_2}, e_{i_3}$ are sequenced in the same partition then $a_{i_1} + a_{i_2} + a_{i_3} = b$ must hold which completes the proof.

□

Theorem 2. *Given an instance of (3P). There exists a solution if and only if the derived instance of (DCBS) has a solution. (DCBS) is therefore $\mathcal{NP}$-complete.*

Proof. Lemma 1 yields a short argumentation:

- Given an instance of (3P) with a solution $A_i = \{a_{i,1}, a_{i,2}, a_{i,3}\} \quad i = 1..m$. We derive an instance of the modified sequencing problem and order the vectors such that each partition is of the form

$$\begin{array}{ccccccccccccccc} & & a_{i,1} & & & a_{i,1} & & a_{i,2} & & & a_{i,2} & & a_{i,3} & & a_{i,3} \\ p, & g,...,g, & e_{i,1}, & g,...,g, & s, & g,...,g, & e_{i,2}, & g,...,g, & s, & g,...,g, & e_{i,3}, & g,...,g, & p \end{array}$$

  We now check the lower and upper bounds for each component. The distance for the components $j = 1...3m$ is always $\underline{m}_j = a_j$ because before and behind an element vector e_j

are sequenced exactly a_j gap vectors which also ensures the upper bound for $j = 1...3m$. Looking at component $3m+1$ we see that the assumption $\frac{b}{4} < a_j < \frac{b}{2} \quad \forall j = 1..3m$ ensures that the distance between two element vectors is at least $\frac{b}{2} + 3$ and at most $b - 1$ (which exactly are the bounds $\underline{m}_{3m+1}$ and $\overline{m}_{3m+1}$). For the same reason component $3m+2$ cannot violate it's bounds ($\underline{m}_{3m+2} = \frac{b}{4} + 1$ and $\overline{m}_{3m+2} = \frac{b}{2} - 1$). Between two vectors of the kind partition or separation vector one element and $2a_j$ gap vectors are sequenced which conforms with the bounds of component $3m + 3$ ($\underline{m}_{3m+3} = \frac{b}{2} - 1$ and $\overline{m}_{3m+3} = b - 1$). Since $\sum_{j=1}^{3} a_{i,j} = b$ we have exactly $2b$ gap vectors, two separation vectors and three element vectors in each Partition i. Therefore the ones in component $3m + 4$ always have a distance of $2b + 5$. The sequence is feasible and we have found a solution of (DCBS).

- As shown in Lemma 1 a feasible sequence for (DCBS) implies that in each partition i the containing element vectors $e_{i_1}, e_{i_2}, e_{i_3}$ belong to values $a_{i_1}, a_{i_2}, a_{i_3}$ that sum up to b and we have found a solution of (3P).

□

3.1.2 Fixing bounds and forgetting the upper bounds

We will now show that even strong simplifications of (DCBS) remain $\mathcal{NP}$-complete by giving a reduction from the Hamiltonian-Path-Problem:

- Given a graph $G(V, E)$.
- Does G contain a path that uses each node of V?

Theorem 3. *(DCBS) remains $\mathcal{NP}$-complete if we make the following simplifications:*

1. *The bounds for each component are equal and fixed to $\underline{m} = 1$ and $\overline{m} = 3$.*
2. *The upper bounds are ignored and the lower bound is fixed to $\underline{m} = 1$.*

Proof. Let $\overline{G} = G(V, \overline{E})$ with $\overline{E} = (V \times V) \setminus E$ the complementary graph of G and $m := \left|\overline{E}\right|$ the number of edges.

1. For each node $v \in V$ we create a *node vector* $b_v \in \{0, 1\}^{m+2}$with $b_v^e = \begin{cases} 1 & v \notin e \\ 0 & v \in e \\ 1 & e = m + 1 \\ 0 & e = m + 2 \end{cases}$.
   Furthermore we need $|V| - 1$ *gap vectors* g with $g^e = \begin{cases} 1 & e = m + 2 \\ 0 & \text{otherwise} \end{cases}$. Components $m + 1$ and $m + 2$ make sure that node and gap vectors always alternate. Furthermore two node vectors v_1 and v_2 are sequenced after each other (in order $..., g, b_{v_1}, g, b_{v_2}, g, ...$) only if there is an edge from v_1 to v_2 in G which means that there is no edge from v_1 to v_2 in $\overline{G}$ and therefore b_{v_1} and b_{v_2} do not both have a zero in components $e = 1..m$.
   If there is a Hamiltonian Path $v_1, ..., v_{|V|}$ in G then $b_1, g, b_2, g, ..., b_{|V|-1} g, b_{|V|}$ is obviously a feasible sequence. And on the other hand, if $b_1, g, b_2, g, ..., b_{|V|-1} g, b_{|V|}$ is a feasible sequence then $v_1, ..., v_{|V|}$ must be a Hamiltonian Path.
2. The proof is similar to 1. but we can drop off the gap vectors and the components $m + 1$ and $m + 2$. Then a Hamiltonian Path in G induces a sequence which is feasible when considering the bounds $\underline{m} = 0$ and $\overline{m} = 1$. We can now finish the proof by inverting each bitvector logicaly.

□

3.2 Polynomial cases

If we assume $\beta \leq \overline{\beta}$ for a fixed $\overline{\beta}$ (DCBS) becomes "easier". First we show that considering only the lower bound $\underline{m} = 1$ yields a dynamic program to solve the problem in time bounded by a polynomial in $n = |F|$. Then it is easy to show that (DCBS) generaly is in $\mathcal{P}$ for bounded upper bounds.

If β is bounded by $\overline{\beta}$ then α is bounded, too, since $\alpha \in O(2^{\beta})$. The idea for a dynamic program is simple. Consider a node set

$$V = \{0, ..., N_1\} \times ... \times \{0, ..., N_\alpha\} \times \{0, ..., \alpha\} \cup \{s\}$$

and add arcs

$$(n_1, ..., n_i, ..., n_\alpha, l) \rightarrow (n_1, ..., n_i - 1, ..., n_\alpha, i) \quad \forall n_1 = 0...N_1, ..., n_\alpha = 0...N_\alpha, l = 1...\alpha$$

whenever b_l and b_i do not have a 1 in the same component. Adding further arcs

$$s \rightarrow (N_1, ..., N_i - 1, ..., N_\alpha, i) \quad \forall i = 1..\alpha$$

yields a tree rooted by s with at most $O(\alpha n^\alpha)$ nodes. A path from s to a leaf of heigth n represents a feasible sequence of the vectors.

The nodes of height k are called $\gamma^k(...)$. The following dynamic program finds such a path:

create $\gamma^0\,(N_1, ..., N_\alpha, 0)$
for $k = 1..n$ ***do***
  for all γ^{k-1} ***do***
    for $i = 1..\alpha$ ***do***
      if $(\gamma_i^{k-1} > 0$ ***and*** *(b_i and $b_{\gamma_{\alpha+1}^{k-1}}$ do not conflict))* ***then***
        create $\gamma^k = \gamma^{k-1}(..., \gamma_i^{k-1} - 1, ..., i)$
        add arc *from* γ^{k-1} *to* γ^k
        if $(k = n)$ ***then return true***
return false

Here "b_i *and* $b_{\gamma_{\alpha+1}^{k-1}}$ *do not conflict*" means that the two bit vectors do not have a 1 in the same component. This will help us later to extend the program to general upper and lower bounds.

Theorem 4. *If* $\underline{m} = 1$ *for all components and* β *is bounded by a constant* $\overline{\beta}$ *then (DCBS) belongs to* $\mathcal{P}$ *and can be solved in* $O(n^\alpha)$ *using the above dynamic program.*

Proof. It suffices to show by induction that the dynamic program works well. I show the following assumtion:

- In the k^{th} iteration of the outer for-loop the paths from γ^0 to any γ^{k-1} yield feasible sequences of the available vectors of length k (considering the last component of each node). The first α components represent how many items of each vector are still available.

Creating node γ^0 does not cause any confilicts. Each of the first α components show the availability of each vector type and the sequence has length 0. Now assume that each node of height $k - 1$ represents a sequence (following the path to the root and considering the last

component). If a node for the vector type i and an arc is being created in the innermost block of the program $\gamma_i^{k-1} > 0$ ensures that the vector type is still available and that the vectors b_i and $b_{\gamma_{\alpha+1}^{k-1}}$ can be sequenced after each other. The created node then keeps the information that one vector less of type i is available. The program returns false if no node of height n can be constructed and true otherwise. □

Corollary 5. *(DCBS) belongs to $\mathcal{P}$ for bounded β and bounded $\overline{m}_j \quad j = 1..\beta$.*

Proof. The idea is to hold the last $M = \max\{\overline{m}_j \,|j = 1..\beta\}$ vector types in each node to make sure that the next vector b_i does not conflict with the last M sequenced vectors. The node set then is $V = \{0, ..., N_1\} \times ... \times \{0, ..., N_\alpha\} \times \{0, ..., \alpha\}^{\max\{\overline{m}_j+1|j=1..\beta\}} \cup \{s\}$ which is still polynomial in n. □

4 A Parameterized Heuristic Procedure

Returning to practice we say that "b_i has option j" if b_i has a one in component j. The main idea (derived from [2]) is simple and is shown best in a small block of pseudocode:

```
S = ∅
while not all vectors sequenced do
  set p_min = min{p_i | S + b_i violates p_i constraints, i = 1..α}
  set b* to that element of {b_i | S + b_i violates p_min constraints, i = 1..α}
    which ensures the most regular flow of ones in each component
  put b* into the sequence
end while
```

In each iteration of the while-loop the set of those vectors is considered which cause the least conflicts. As in other greedy strategies the problem could arise that "better" vectors are sequenced first and leave the rest of the sequence to vectors that are harder to sequence. To avoid this we try to keep the rate an option enters the sequence as near to the average as possible.

Let N_i the quantity of b_i and $N = \sum_{i=1}^{\alpha} N_i b_i$ the vector containing the quantities of the options $j = 1..\beta$ in F. Then with $X_{j,k}$ being the number of sequenced vectors with option j in the k^{th} step of the while-loop we want to minimize

$$\left\| \frac{X_{k-1}}{k} + b_i - \frac{N}{|F|} \right\| \tag{7}$$

This approach was first introduced by Monden [4] but without further constraints. Here the presented heuristic follows both goals, avoid distance violation and try to keep average option consuming. A similar approach (considering car-sequencing) was made in [1]. They also presented a column-generation approach to solve small instances of the problem up to the size of about 50. The instances that we have to cope with (arising in automobile production) have a size of about 2000.

The approach to calculate optimum sequences of smaller size containing a fraction of each vector type and combining the small sequences afterwards could be used to get aproximate solutions for bigger problems but this would still imply to calculate many small instances since vectors with lower occurance have still to be considered (they cannot be put into the sequence afterwards). Furthermore in automobily production it is hard to ensure a specific input sequence at the assembly line anyway since there could occure many failures during the previous production process which would alter the sequence. Therefore it is suitable to assume that in each

time step a limited number of vectors (say in a choice-free memory) is available for sequencing. The heuristic presented above follows this just-in-time apparoach.

We now extend the distance constraints to a more general class of constraints which arise in car production. Consider the expression

$$\sum_{j=1}^{\beta} W_{j,\rho} \left(X_{j,k} - X_{j,k-L_\rho} \right) \leq M_\rho \qquad \forall k = L_\rho + 1..n$$

where $W_{j,\rho}$ is some weight for option j and restriction ρ and M_ρ is a constant for constraint ρ. We can now cope with more than just distance constraints:

- lower bound distance constraints (W_ρ is the j^{th} unity vector, $L_\rho = \underline{m}_j$ and $M_\rho = 1$)
- upper bound distance constraints (as above with $-W_\rho$ and $M_\rho = -2$ and $L_\rho = \overline{m}_j$)
- at most x_j vectors in every subsequence of length y_j should contain option j, known as Car-Sequencing-Problem (W_ρ is the j^{th} unity vector, $L_\rho = y_j$ and $M_\rho = x_j$)
- sequence at most x_1 cars with option j_1 out of y but if they have option j_2, too only x_2 out of y are allowed
- combine options, i.e. measure the distance between options of a specified set

We remark that even banning restrictions that do not allow vectors in special positions of the sequence are possible but not suitable for the heuristic because too much global knowledge about the already built sequence and the remaining vectors is necessary.

In automobile industry it often occurs that some options should be handled with higher priority. This can be done by a higher penalty for a broken constrained and by replacing the norm in 7 by a weighted norm like

$$\|x\|_* = \sqrt{\sum_{j=1}^{\beta} w_j x_j^2}$$

where more important options j get a higher weight w_j.

Open Issues of the Heuristic

The procedure above causes some problems that we want to point out here

- The restrictions must be chosen relatively soft because otherwise the sequence is not leveled anymore (consider the instance with vector types $\begin{pmatrix} 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \end{pmatrix}$ and $\begin{pmatrix} 1 \\ 1 \end{pmatrix}$ and minimum distance 1. The all-one vectors will always be sequenced last causing double penalty)
- Although the options are leveled over the sequence it can still occur that "better" vector types are sequenced first while others are left for the end of the sequence (see example above)
- Other important constraints for automobile production are not being considered (banning, parallel assembly lines on which other restrictions have to hold)

5 Concluding Remarks

Since the presented heuristic procedure is not completely new the main result of this paper lies in the complexity analysis of (DCBS). It is still left open, how to get results that overcome the open problems of the heuristic. It has been shown that the complexity of the problem is high and that even strong simplifications still remain hard to solve. By loosening the requirement of optimality we get a just-in-time heuristic procedure that copes with sudden variations of the incoming sequence that can occur in car production having a choice-free memory prior to the line which holds the available cars.

References

[1] A. Drexl and A. Kimms, *Sequencing JIT Mixed-Model Assembly Lines Under Station-Load and Part-Usage Constraints*, Managment Science **47** (2001).

[2] J. P. Garcia-Sabater, *The Problem of JIT Dynamic Sequencing. A Model and a Parametric Procedure*, ORP^3, Paris, September 26–29 (2001).

[3] M. R. Garey and D. S. Johnson, *Computers and Intractability. A Guide to the Theory of $\mathcal{NP}$-Completeness*, W. H. Freeman and Company, 1979.

[4] Y. Monden, *Toyota Production System*, Institute of Industrial Engineers Press, 1983.