Cyclogram Unfolding in Dynamic Hoist Scheduling Problem
Krzysztof Kujawski, Jerzy Świątek

To cite this version:

HAL Id: hal-00652639
https://hal.archives-ouvertes.fr/hal-00652639
Submitted on 16 Dec 2011

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
Cyclogram Unfolding in Dynamic Hoist Scheduling Problem

<table>
<thead>
<tr>
<th>Journal:</th>
<th>International Journal of Production Research</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manuscript ID:</td>
<td>TPRS-2009-IJPR-1192.R2</td>
</tr>
<tr>
<td>Manuscript Type:</td>
<td>Original Manuscript</td>
</tr>
<tr>
<td>Date Submitted by the Author:</td>
<td>11-Aug-2010</td>
</tr>
<tr>
<td>Complete List of Authors:</td>
<td>Kujawski, Krzysztof; Wrocław University of Technology, Institute of Informatics in the Faculty of Computer Science and Management Świątek, Jerzy; Wrocław University of Technology, Institute of Informatics in the Faculty of Computer Science and Management</td>
</tr>
<tr>
<td>Keywords:</td>
<td>SCHEDULING, MATERIALS HANDLING, MANUFACTURING SYSTEMS</td>
</tr>
<tr>
<td>Keywords (user):</td>
<td>hoist scheduling problem, DHSP</td>
</tr>
</tbody>
</table>
Electroplating Production Scheduling by Cyclogram Unfolding in Dynamic Hoist Scheduling Problem

K. Kujawski\textsuperscript{a}, J. Świątek\textsuperscript{a}
\textsuperscript{a} Institute of Informatics, Wrocław University of Technology, Wrocław, Poland

Tel. +48 71 7984949, Fax. +48 71 7984948 (K. Kujawski)
Email: krzysztof.kujawski@pwr.wroc.pl
Postal Address: 53-653 Wrocław, Inowroclawska St. 19/11, Poland

The paper presents a scheduling problem that exists in electroplating lines. An electroplating line is an automated manufacturing system, which cover parts with a coat of metal. It consists of a set of tanks that chemically process the items and hoists that transport the items between workstations. Scheduling the movements of these hoists is commonly called a hoist scheduling problem (HSP). The most common approaches to the problem are cyclic hoist scheduling (CHSP) and dynamic hoist scheduling (DHSP). The paper presents a DHSP solution method. The method divides the problem to real time and non-real time. Special schedules, called cyclograms, allow minimizing the length of non-real time calculations. A notion of the problem is introduced, an outline of a scheduling system is presented, as well as the heuristic algorithm itself. The results of described method, referred as a cyclogram unfolding method, are compared to several cases available in literature.

Keywords: hoist scheduling problem, dynamic hoist scheduling problem, electroplating, real time scheduling, flexible manufacturing system.

INTRODUCTION

In electroplating industry a HSP – Hoist Scheduling Problem occurs. It deals with creating a proper schedule for machinery working on a production line. To fulfil customer needs, there is a necessity of flexible production systems. In such systems, there is a possibility to produce many types of products. The production orders change frequently. Such conditions are often in electroplating lines, although printed circuit board production lines and some food processing lines are organized similarly. An electroplating production line most important feature is chemical processing. It implies surface treatment constraints on the production. Products cannot stay in workstations after processing, because they would be destroyed, e.g. melted in acid or rendered too hot. No-buffer rule also applies here. Semi-finished products cannot be stored in order to be used later, since they would lose their qualities. A schedule, which does not meet no-wait and no-buffer constraints, is treated as infeasible.
Mentioned constraints render the hoist transportation the most important subject of scheduling process. No-wait constraint requires a hoist to be present when maximum processing time is reached. Hoist is also only “buffer” available. Only during transportation, a product is not being processed. Additionally, transportation times are comparable to processing times, what makes them important in real-life schedules. A workstations schedule can be determined directly from a hoists schedule.

A production in electroplating lines can be variously organized. One or multiple hoists may be present at the production line, one or multiple types of items production can be scheduled. Cyclic (Cyclic Hoist Scheduling Problem) or dynamic production are also considered. In CHSP schedule is static. It is repeated in order to produce more items. Items must be introduced to line cyclically and the type of produced items is constant over time. In DHSP (Dynamic Hoist Scheduling Problem) schedule is created during production, items may vary both in type and sequence of introduction times.

In this paper, a heuristic scheduling method for the dynamic case is proposed.

HOIST SCHEDULING
An electroplating line covers processed items with thin material coatings using chemical reactions – usually by galvanization process. It is an automated production system, which use hoists as transportation medium. A production line consists of series of workstations capable of performing different elementary chemical reactions or material processing operations. In order to achieve a proper effect, a rough input product must be subjected to several reactions in specific sequence.

A production line consists of:

- Baths (tanks), a production line workstations, which perform a certain stage of processing. Tanks are capable of performing a specific chemical or material processing. Some baths can perform more than one task – they are called multifunctional baths. Some stages processing times, are significantly longer, than
other. For those stages, resource is multiplied in order to avoid a bottleneck. Such multiplied instances of baths, of the same type are called bath group. Baths are arranged in line, and they create an axis of a hoist movement.

- Loading and unloading stations. Some workstations only introduce new products to line or are the place where finished products go.
- Hoists, controllable automatons, capable of transferring products between workstations. Hoists move only in production line axis. Hoists cannot pass each other. The hoist can pick up and put down products to workstation. Hoists cannot pass products between each other.
  
  Because of chemical nature, processing starts when a hoist plunges an item to a bath as a consequence of putting the carried item down. Processing stops, when the hoist picks the item out of bath.

  According to those rules cyclical and dynamic production is taken into consideration. In CHSP, it is assumed that new items are available in input buffers every constant amount of time. Usually it is considered that only one type of product is introduced each cycle. For these conditions, a schedule called cyclogram can be created. Cyclogram can be repeated at will, each repetition is producing a number of items. Cyclic scheduling is the most frequently considered organization of production in surface treatment systems, i.e. Brauner 2001 or one of the first papers describing hoist scheduling Phillips 1976.

  Usage of cyclograms has a number of advantages. The production can be easily controlled, due to a constant number of steps, which create a cyclogram. Performance of production is known in advance and constant. In spite of those positive aspects, there are some boundaries of this type of production. It is difficult to create an efficient cyclogram supporting production of multiple item types. In case of any small change in items order, a costly line reset is required.

  When flexibility of production is a priority, dynamic scheduling is used.

  Figure 1 presents an overview of typical electroplating lines. There are two hoists, three products waiting to be processed, and four items in-process. The hoist on
the left is picking the item from the second workstation in line. There are 8
workstations in total.

FIGURE HERE: Figure 1 Electroplating Line

DYNAMIC HOIST SCHEDULING PROBLEM

Electroplating production lines, alternatively called surface treatment systems
(referring to Subaï 2006), are used in production of many types of items. That requires
flexibility both in the methods of scheduling and in available workstations. In some
cases, only a subset of workstations is used to produce one item. Changing of the
cyclogram requires the production line reset. All items must be unloaded and hoists
need to go to their initial positions. This is a resource consuming operation. When
many types of products are manufactured in short series, it may happen, that the line
reset is required each couple of items produced. Performance of manufacturing is
impaired due to a long time of line reset.

Some authors Yan 2009, Chu 2009 describe also similar problem, where
surface treatment system has fixed times of processing in workstation. The Yan 2009
paper involves multiple item types processing, although the system does not work in
real-time.

Dynamic production often comes into usage instead of cyclic work. A
production line controller detects changes in orders when products are designated to
production and updates the schedule on the fly. Products are queued for a production.
They are introduced to the production line in sequence of the queue. The production
queue can contain items of various types. It can be modified at any time by adding
new items to it. We assume here that frequency and type of items cannot be foreseen
and no claims can be formulated about changing of the queue. Having in mind that
scheduling is done during production, a scheduler has to create routing of hoists,
which will cause products finish their production sequence, for both items in-process and in the queue, while maintaining feasibility of schedule.

The scheduler usually does not create a completely new schedule. The scheduling is done during production. Before rescheduling finishes, line works according to the current schedule. After feasible solution is found, adaptation of schedule is done in order to accustom the production system to new products waiting in the queue. When new order is stated at time $\tau$, the line controller decides about the adaptation time $t = \tau + \Delta$, $\Delta > 0$ - time from which an adapted schedule is implemented, where

$$t = \tau + \Delta, \Delta > 0$$

- the line controller estimates the amount of time $\Delta$ which will be sufficient to calculate a new schedule. The line controller provides also information about the line state at the adaptation time. These parameters are needed for so called Local Problem (referring to Lamothe 1996). A new feasible schedule is generated.

Solving a Local Problem lasts $\sigma$ amount of a computation time. At this point the new schedule is checked against its applicability ($\sigma < \Delta$). In case the schedule is infeasible, a new adaptation time is figured out and a new Local Problem is solved. When the adaptation is successful the controller is responsible of implementing the new schedule to the production line. The described process is presented graphically in the Figure 2.

FIGURE HERE-►►►►►►►Figure 2 Outline of dynamic scheduling system

The new schedule must include completing products, which are in-process at the time of new order specification. They are processed at the line in workstations, which means that they are directly vulnerable to be spoiled. That makes the schedule infeasible and renders it non-usable.

Dynamic hoist scheduling problem can be presented as continuous solving of a Local Problem in real-time. This divides the schedule creation process to two main parts: a Local Problem and a real-time rescheduling problem.
The creation of a schedule, based on the line state and the queue of items to be produced, while respecting all constraints and maximizing the performance of manufacturing, in this context, will be called a Local Problem. The Local Problem is separated from real-time issues and is typical example of a discrete domain optimization problem with constraints.

The real-time rescheduling deals with the line state controlling, verifying feasibility of updated schedules, implementing found schedules to automatons, controlling a frequency of rescheduling and item introduction times. It is important to control the rescheduling frequency. In case rescheduling is done, basing on a very close future line state, it may use line resources better and provide the better performance. However, in case solving a local scheduling problem lasts long enough so the line state outdates, found schedule will be impossible to implement, as it will refer to the past. This is a typical problem of responding in sufficient time in real-time systems. Introduction frequency of new items also has to be decided. Designating an item to production quickly may increase performance, but it also may cause that it is difficult to find a feasible schedule since a load of the resources is higher.

This paper focuses on solving a local scheduling problem.

Formulation of the Local Scheduling Problem
The problem of finding the schedule for a specified line, a queue of items, and a line state can be described as optimization problem. The problem parameters are: the properties of the line (workstations locations, sizes, workstation counts), the hoists count, the hoists speed. Other parameters concern produced product types: the sequence of workstation for each product type, minimum and maximum times of processing in visited workstations. The parameters describing the line state in adaptation time: hoist positions, processing stages of products, which are already on
the line, a queue of to be processed items, are also considered. A result schedule can be described by decisive variables of this problem: routes of hoists, assignments of pickup operations to hoists, assignments of put down operations to workstations in groups, times of pickup and put down operations. The optimization criterion is minimizing the time of a last put down operation. Such criterion maximizes the performance of the production line for a given product order. Similarly, to other scheduling problems, DHSP has many constraints. Schedule must fulfill process requirements – processing sequence must be correct, processing times must not be shorter than specified minimum times and longer than specified maximum times. A hoist can carry only one item at the time. A workstation can process only one item at the time. Hoists cannot move outside their physical capabilities, and they cannot collide. We provide only relevant symbols that are necessary to describe the cyclogram unfolding method.

\[ N \text{ - number of product types, } n \in \mathbb{N} = \{1, ..., N\} \]

\[ Z = \{ \zeta(1), ..., \zeta(K) \} = \{ s(1), ..., s(KC) \} \cup \{ \delta(1), ..., \delta(KQ) \} \text{ - queue of items to be produced, } s(k) \in \mathbb{N} \text{ - in process products types upon rescheduling, } \delta(k) \in \mathbb{N} \text{ - products in queue. } \zeta(k) \in \mathbb{N} , KQ \text{ - number of items in queue. } KC \text{ - number of items processed on line during rescheduling. } K = KC + KQ. \]

\[ L \text{ - number of workstation groups present on line, } g_l \text{ - number of workstations of type } l, l \in \mathbb{L} = \{1, ..., L\}. \]

\[ O_n = \{ w_1, ..., w_i(n) \} \text{ - sequence of workstation group types necessary to manufacture product of type } n. w_i(n) \in \mathbb{L} \text{ - workstation group type of } i \text{-th product processing stage of type } n. i(n) \in I(n) = \{1, ..., I(n)\} \text{ where } I(n) \text{ is number of steps in of product type } n \text{ processing sequence. } H \text{ - number of hoists present on line, } b \in \mathbb{H} = \{1, ..., H\}. \]

\[ \mu_{s,j} \]
\( \eta_{k,i} \) - minimum and maximum amount of time needed for product of type \( n \) should be processed in \( l \)-th workstation group.

Parameters are derived from physical properties of a production line and a line state in moment of rescheduling. A schedule can be created basing on values of following optimization problem decisive variables: routes of hoists

\[ U(b, \lambda), \lambda \in \{1, \ldots, \Upsilon\} \] - the routes of hoists, represented as position of hoist \( b \) in production time \( \lambda \). \( \Upsilon \) - time of production cease. The variables \( \tilde{t}_{k,i,(z(k))} \), \( \tilde{L}_{k,i,(z(k))} \) - the moments of picks and puts of products in \( i \)-th step workstation have to be established as well. The activities of picking up and putting down a product from/to workstation have to be assigned to the specific hoists \( \tilde{b}_{k,i,(z(k))} \in \tilde{H} \), \( \tilde{b}_{k,i,(z(k))} \in \tilde{H} \).

Additionally, for a given step of product processing, a specific station from group has to be chosen: \( a_{k,i,(z(k))} \). Note that not all decisive variables need to be found. Pickup and putdowns of items already in process are already calculated up to the time of adaptation \( t = \tau + \Delta \). Initial hoists positions are in fact hoists positions in adaptation time \( U(b,0) = U(b,\tau) \).

An optimization criterion used to reach a maximum manufacturing performance is \( Q = \max \left( L_{k,i,(z(k))} \right) \). The optimal solution is found when the criterion reaches a minimum. The criterion is minimized when last item in sequence leaves production line as soon it is possible.

Besides other constraints, the relevant are that \( \tilde{t}_{k,i,(z(k))} < \tilde{L}_{k,i,(z(k))} \) - conditions of proper processing sequence of products and

\( \mu_{z(k),a,(z(k))} < L_{k,i,(z(k))} \) - condition of proper realization of \( i \)-th step of item \( z_\lambda \) production.
REVIEW OF EXISTING SOLUTIONS

The described problem was a topic of many publications. Nevertheless, the most of works are referring to the simplified problem, as they introduce additional constraints.

A method proposed by Hindi 2004 deals with a scheduling problem, which is equivalent to the DHSP Local Problem. The presented method tries to create an optimal schedule, by solving a scheduling problem as the CSP (Constraint Satisfaction Problem). Constraints are propagated and heuristic forward checking and backtracking is done. In spite promising results achieved, the method is hardly usable in real cases because of simplifications. Only one hoist is supported, no multifunction tanks or bath groups are considered.

Jegou 2006 proposed somewhat better solution. This heuristic deals also with deciding when items are introduced to production. The method uses an agent system, to react at production events: new product, product processing reached a minimum time etc. A schedule is built as a result of reacting on these production events.

Schedule is created only in short time extent, waiting for another event. Solution supports multiple hoists and workstation groups.

Reactive scheduling is an almost perfect solution for non-cyclic manufacturing systems. The system is very flexible and can support production state changes. Authors state that even hoist breakdown during production can be supported.

Calculation steps of scheduling are small and homogenous, which is important in real-time computation. There are also cons of this solution. It may happen that some products will be defective. Another problem is that the proposed solution does not support multifunction workstations and created schedules have low performance.

There are also a number of surface treatment systems, which are similar to DHSP, yet are treated as separate problems, since they differ in vital details. For instance, in paper Chauvet 2000, authors describe a production line where multiple
hoists work but collisions are not taken into consideration. Additionally stages of productions do not have specific minimum and maximum processing time, but are defined as intervals, with one maximum time set as infinity.

**CYCLOGRAM UNFOLD METHOD**
In described scheduling problem many of its elements have combinatory character. In spite of constraints, finding the optimal solution would require a full subspace search. In real-life cases of big scale manufacturing, where many product types are considered, this is not an option. Scheduling in real-time is making this problem even more difficult. As a consequence, heuristics are used instead of exact methods. Although heuristics do not guarantee that found solutions will be optimal, they allow creating high performance schedules, while providing lower computational complexity than exact methods.

In this section, a heuristic method is described. It minimizes calculation overhead during real-time schedule generation by using earlier prepared plans. The method emulates a behaviour of a line during cyclic production. It modifies a schedule of line reset in the way it is the shortest. For a proper working, a method of cyclogram unfolding requires a set of plans called cyclograms. Creating cyclograms is quite a challenge itself. However, since cyclograms are often used on real-world installations, there are many works present concerning cyclic production and creating cyclograms. They can be used as source of the cyclograms for cyclogram unfolding method. The described method requires at least one cyclogram for each of \( N \) types of processed products.

**Cyclogram**
A Cyclogram is a hoist route plan. Its length is called cycle-time \( T \). A Cyclogram is built in the way, that it can be repeated infinitely, and it is still feasible manufacturing
schedule. Cyclograms are widely used in surface treatment systems Che 2004, Liu 2002, Mak 2002. Cyclograms are very useful when only one item type is going to be produced for a significant amount of time, because all scheduling can be done before production starts. There are a variety of successful methods of creating cyclograms. Integer, linear programming solutions presented by Liu 2002 and Liu 2008. Branch and Bound algorithms presented by Yan 2008, Che 2004. And other heuristic methods, like presented in Mak 2002 and Kujawski 2007. Since cyclogram unfolding method is able to use any cyclograms, the best results of mentioned methods can be selected for real-time scheduling. On the other hand, not all methods support multiple hoists (Che 2007, Liu 2002, Liu 2008 only two hoists, Yan 2008), arbitrary production sequence (i.e. Che 2004, Leung 2003, Yan 2008), workstation groups (i.e. Leung 2003, Liu 2002), other features like multi-function workstations, multiple columns, etc....

We will only consider cyclograms where one product is finished and leaves the line each cycle. The line performance is directly connected with cycle length in such case. In literature there can be found higher grade cyclograms (Che 2005, Che 2009), which produce more products each cycle, but they are less valuable in the current context and are not going be considered from now on. Mentioned papers Che 2005, Che 2009 are addressed to the slightly modified problem, with “no-wait” rule. As soon as a part completes processing on a machine, it must be immediately removed from that machine and transferred to the next machine.

Cyclograms can support multiple hoists, multifunction tanks and workstation groups. All those features are also available in cyclogram unfold method, depending on the used cyclograms.

FIGURE HERE->>>>>>>>>Figure 3 Cyclogram of two hoists with one group of baths (6, 7)
Cyclogram presented in the Figure 3 is a graphical representation of schedule. Horizontal axis represents the time flowing from left to right. Vertical axis represents the physical position on line in hoist movement axis. Each workstation is represented as the number. This particular diagram represents two hoists routes by lines. Left arrow represents putting the item down, right arrow picking up. Horizontal bar aligned to workstation position represent item processing within the workstation. Workstation 6,7 form a group, hence hoist pick and put operations are connected. Note that hoists do not collide because they do not get close enough in any time moment. Such charts are frequently used by electroplating companies because they give an overview of schedule and allow easy verification of feasibility.

Cyclic scheduling can be formulated as an optimization problem similarly as dynamic scheduling.

Some of the notion is common in both problems and will not be repeated.

Main features of cyclogram is capacity and cycle time. Cycle time is a length of cyclogram in time domain. Capacity is a number of cycles needed to be done in order to fully load a production line and produce first item. After a production line is loaded each next cycle produces a new item. There can be many feasible cyclograms for one item type. Let us assume that there are a number cyclograms for each item type \( n \in \{1,\ldots,N\} \) referred as \( V_{1},\ldots,V_{n}, \ v_{n} \in \tilde{V}(n)=\{1,\ldots,V_{n}\} \cdot V_{n} \) - number of cyclograms prepared for item type \( n \cdot T(v_{n}) \) – cycle-time of cyclogram \( v_{n} \), \( G(v_{n}) \) – capacity of cyclogram \( v_{n} \). Cyclogram can be defined similarly as schedules in DHSP – routes, times of operations and assignment of operations to hoists. \( U(v_{n},h,\Lambda) \), \( \Lambda \in \{0,\ldots,T(v_{n})\} \) – routes of hoists, represented as position of hoist \( h \) in production time \( \Lambda \) for cyclogram \( v_{n} \). \( \tilde{I}(v_{n},i(n)) \), \( \ell(v_{n},i(n)) \) – moments of pick and put of \( i \)-th stage of product
n-th in cyclogram $v_s$, $\vec{h}(v_s,i(n)), h(v_s,i(n))$ – indexes of hoists which perform $i$-th pick and put of product $n$-th in cyclogram $v_s$. $a(v_s,i(n)), x$ - number of workstation in group $w_{(a)}$ in cycle $x$.

Although a line can consist of groups of workstations, there is no need of assigning putdowns and pickups to specific workstation for each processing stage. In cyclic work, it is assumed that workstation group is used as FIFO queue. It means that

$$a(v_s,i(n)), x = x \mod g_i + 1 \text{ where } l = w_{(a)}.$$  

An optimization criterion used to reach a maximum manufacturing performance is $Q = T(v_s)$, providing that number of items produced is not smaller than $G(v_s)$. Optimal solution is found when criterion reaches a minimum.

Hoist speed constraints, no collision policy, one product per hoist and workstation requirements are still in place. A Cyclogram must also bend to pickup and putdown times relation constraint:

$$\mu_{s,j} \leq (t_{s,i(v)} - L_{s,i(v)} + T) \mod T + T \cdot (g_{a(i(v))} - 1) \leq \eta_{s,j}, \text{ where } g_{a(i(v))} \text{ is a number of workstations in workstation group performing } i \text{-th step of product } n \text{ and } T \text{ is a cycle-time of cyclogram.}$$

The Cyclogram Unfolding
Let us mark $C(x,v_s)$ as cyclogram unfold operation, for $x$ items of type $n$ using cyclogram $v_s$. As a result of $C(x,v_s)$ we get a schedule based on cyclogram $v_s$, which produces $x$ items of type $n$. Cyclogram operations are repeated certain number of times until $x$ items leave the line. Length of this schedule: $\Upsilon = |C(x,v_s)| = (G(v_s) + x) \cdot T(v_s) - T_{load}[v_s] + T_{unload}[v_s]$. $C(x,v_s)$ creates solution for $Z = \{z_1, \ldots, z_x\}$, where $z_4 = n$ by assignment: $U(h,t) = U(v_s,h,t \mod T(v_s))$, $t \in [0, \ldots, |C(x,v_s)|]$. 

http://mc.manuscriptcentral.com/tprs Email: ijpr@lboro.ac.uk
\[ \tilde{t}_{k,i(a)} = t(v_s, i(n)) + (k + \theta(i(n))) \cdot T(v_s), \quad \theta(i(n)) = \sum_{a=1}^{n_{\mathcal{V}} - i(n)} \begin{cases} 1 - \tilde{t}(v_s,a) < t(v_s,a) \\ 0 - \tilde{t}(v_s,a) > t(v_s,a) \end{cases}, \]

\[ \tilde{t}_{k,i(a)} = t(v_s, i(n)) + (k + \theta(i(n))) \cdot T(v_s), \quad a_{k,i(a)} = a(v_s, i(n), \theta(i(n))) \] where \( \theta \) is counter of passing product between two cycles. \( \tilde{h}_{k,i(\eta)} = h(v_s, i(n)), \quad \tilde{h}_{k,i(\eta)} = h(v_s, i(n)) \).

**FIGURE HERE->>>>>>>Figure 4 Cyclogram from Figure 3 unfold, 5 products, 9 cycles**

The Figure 4 presents the result of the operation of cyclogram unfolding for a cyclogram \( \mathcal{V} \) with capacity \( G(\mathcal{V}) = 5 \). The figure presents production schedule of five items created by unfolding cyclogram \( \mathcal{V} \) from the Figure 3.

**Segment operations**

In the Local Problem a product type can partition a queue. Segmenting operation \( \Xi(Z) \), divides products from orders to segments of items of the same type, such that:

\[ \Xi\left(\{\delta(1), \ldots, \delta(KQ)\}\right) = \{s(1,1), \ldots, s(1,S_i)\} \cup \ldots \cup \{s(SEG,1), \ldots, s(SEG,SEG)\} \], where SEG is number of segments, \( S_i \) is a number of items of \( i \)-th segment and \( \sum_{i=1,\ldots,SEG} S_i = KQ \),

\[ s_{i,j} = s_{i,m} \forall j, m = 1, \ldots, S_i, i = 1, \ldots, SEG \]. The segmenting operation divides new order to subsequences of items of the same type. E.g., for queue \( Z = \{1,1,3,1,2,2,3,3\} \), \( \Xi(Z) \) means that following segments are created: \( \{1,1\}, \{3\}, \{1\}, \{2,2\}, \{3,3\} \). That means that order of 8 products of 3 types is divided to 5 segments. The segmenting operation keeps the original sequence of products in queue.

For each segment created by \( \Xi(Z) \) we can propose a route using cyclogram unfolding operation \( C(x, \mathcal{V}) \). Schedules generated for segments are not enough to create a valid schedule for whole queue \( Z \), but they can be adapted together.

In order to fit routes generated by cyclogram unfolding to the final schedule, operation of time shift is used. As well as rerouting of hoists operation. Let us use
For Peer Review Only

Ψ \( C(x, v_a), \Delta \) as shift in time \( \Delta \) of cyclogram unfolding \( C(x, v_a) \). The \( \Psi \) operation causes following changes in values of decisive variables: \( \tilde{L}_{k,i}(\tilde{\xi}(k)) \equiv \tilde{L}_{k,i}(\tilde{\xi}(k)) + \Delta \),

\[
L_{k,i}(\tilde{\xi}(k)) \equiv L_{k,i}(\tilde{\xi}(k)) + \Delta ,
\]

\[
U(h,t) \equiv \begin{cases} U(h,t-\Delta), t-\tau > 0 \\ U(h,0), t-\tau \leq 0 \end{cases} .
\]

In order to find a valid schedule for a given queue \( Z = \{z_1, \ldots, z_K\} \) by using segments, a series of time shifts \( \Delta \) between all segments in queue \( Z \) has to be determined, where \( s \in \{1, \ldots, SEG-1\} \). Besides time shifts a hoist routes \( U(h,t) \) must also be determined, because the \( \Psi \) operation may break collision constraints.

Determining of time shifts and routes can be performed sequentially by determining respectively \( \Delta_1, \Delta_2, \) etc. It is possible because of constraint

\( \Delta_1 < \Delta_2 < \ldots < \Delta_{SEG-1} \). Scheduling item queues composed from many segments may be solved as many updates of the queue segment by segment. It does not interfere with finding routes, as they need to be defined after all items from segments are completed.

In this way local scheduling problem is divided to several independent problems, with decisive variables reduced to \( \Delta \) and hoist route matrix to be determined. Decisive variables of the original problem \( \tilde{L}_{k,i}(\tilde{\xi}(k)), L_{k,i}(\tilde{\xi}(k)) \), \( \tilde{b}_{k,i}(\tilde{\xi}(k)), b_{k,i}(\tilde{\xi}(k)) \) and \( a_{k,i}(\tilde{\xi}(k)) \) are already determined by cyclogram unfolding procedure, or are bound to value of \( \Delta \). Solving such simplified task may not give optimal solution, tough solving itself is much easier and less complex procedure.

Referring to original problem one additional constraint is present:

\( \tilde{L}_{s,i} < \Delta_{s+1} \leq \tilde{L}_{s,i} + \varepsilon, a = s, b = \tilde{b} \) - a time shift must be at least bigger, than time of introduction of last item from last segment and it can be only smaller, than time of finishing this product. When \( \Delta_{s+1} = \tilde{L}_{s,i} + \varepsilon, a = s, b = \tilde{b} \) the segments are shifted.
enough so they do not influence each other. No line resources are used after time
\[ \Delta_{t+1} = \Delta_{t, t+1} + \varepsilon, a = \varepsilon_{t, t+1}. \] It is similar situation as if new production was started when
line is unloaded. \( \varepsilon \) - is minimal time needed for hoists to take their start up positions.
\( \varepsilon \) is constant value which can be calculated basing on production line parameters.

When dealing with segments, some of the constraints from the original
problem are always satisfied. For instance the time of stage processing and that items
are processed in valid sequence. Some of the other constraints are also always
satisfied, in time intervals outside of segment overlapping. This makes scheduling
much easier.

The Cyclogram Unfolding Method Algorithm
The cyclogram unfolding method is a heuristic method, which is able to create
schedule for given DHSP problem. It bases on segmenting operation \( \Xi \), shifting
operation \( \Psi \) and routing method (point 4b) proposed in Kujawski 2007

Cyclogram Unfolding
1) Calculate \( \Xi \) for queue \( Z \)
2) for \( x = 1 \) to SEG in \( \{s(1,1), \ldots, s(1,S)\} \cup \ldots \cup \{s(SEG,1), \ldots, s(SEG,SSEG)\} \)
   2a) Select arbitrary \( v_{s1} \) from \( \{1, \ldots, V_s\} \)
   2b) Calculate \( S(x) = C(S, v_{s1}) \).
3) Result = \( S(1) \)
4) for \( x = 2 \) to SEG
   4a) Create \( \hat{\Delta} \) - set of \( \Delta \) to check by constraints propagation
   4b) Find smallest \( \Delta \) from \( \hat{\Delta} \) that there exists
      collision-less routing for Result \( \cup \Psi(S(x), \Delta) \).
   4c) Result = Result \( \cup \Psi(S(x), \Delta) \)

Results provided in the next sections are generated assuming that in point 2a)
we select the cyclogram with the shortest cycle-time. Point 2a) is the next field of
research. In Kujawski 2009 we show that there is no optimal cyclogram from a given
set that minimizes the overall criterion.
Creation $\Delta$ in point 4a) of the algorithm is done by propagating two of the problem constraints. At the beginning $\Delta_0$ is some number, constrained from above and below by $\tilde{t}_{x,0} < \Delta_{x+1} \leq \tilde{t}_{x,j(x)} + \varepsilon$, $a = j_{x,j(x)}$. The constraints that a hoist can carry only one item at time and that a workstation can process one item at time are used to limit number of individual time shifts to check.

One of the advantages of the cyclogram unfolding method is that exact item processing times for processing stages are known in advance. They are always equal to the time of the item processing in a cyclogram that we use in step 2a). The time of the processing of $i$-th stage of each item from $n$-th segment is equal to $\tilde{t}(v_s,i(n)) - \tilde{t}(v_s,i(n))$. If we always select the same cyclogram for given item type, the exact processing time is known before the production starts.

**The real-time scheduling**

The cyclogram unfolding method can be described as largely flexible method in context of real-time scheduling of upcoming orders during production. Flexibility is provided by:

- Work on segments allows splitting large orders to a separate, smaller local problems. In this way the created schedule can be appended many times. Schedule update for single segment is faster than for whole new order. Each found time shift $\Delta_x$, causes update of the production schedule and pushes out-of-date border of real-time system further into the future. It gives more time for finding feasible schedule in general.

- When out-of-date border is close and our scheduler cannot find a proper time shift, method can quickly return less efficient but feasible schedule. A time shift can be set to $\Delta_{x+1} = \tilde{t}_{x,j(x)} + \varepsilon$, $a = j_{x,j(x)}$, which we know that is the definitely feasible shift.

- The most of time-consuming decisions were eliminated from on-line scheduling. The original scheduling problem included decisions about assignment of pickups to hoists (complexity $O(n^2)$) and sequencing of processing times (complexity $O(n!)$). Thanks to the cyclogram unfolding the most of these decisions are already made during creation of cyclograms. They do not influence complexity of scheduling during adaptation.
Experimental results
In order to compare work results of the proposed hoist scheduling method with results of methods presented in literature, a series of tests on benchmark problems were analyzed. We present comparison of results from Jegou 2006, Paul 2007 and Lamothe 1996.

Table 1. Comparison to literature benchmark problems
In Jegou 2006 a local scheduling problem of 3 types of products and 2 hoists is given as the example. A goal was to create a schedule for 8 hours of work and see how many products will be finished in this time.

In Paul 2007 benchmark deals with 4 types of products on single hoist line. A mean of 10 tests with random sequence of 40 products with given quantities of each product is presented as the result. Presented results are for default parameters presented in Paul 2007.

Lamothe 1996 presents a classic benchmark problem known in HSP, introduced by Phillips 1976. Only one type of product is processed. Lamothe 1996 uses this process definition for scheduling with two hoists and queue of items. Since only one type of product is scheduled and new item is always accessible in input buffer, although no cyclic assumption, authors claim that schedule should have cyclic nature in this case.

The Case Study
The production line in Wrocław, Poland is a typical small electroplating line. It is used in production of metal furniture elements. The line consists of 16 workstation groups composed into one column. One group has 3 workstations, and the other groups are in fact, singular workstations. Two hoists are available. Hoists maximum speed is 0.7 m/s. Hoist collision zone is 2 meters - two hoists centers cannot be closer than 2 meters or the collision occurs.
We will consider two item types that are frequently ordered together. Let us mark the technology process of technical chroming as a "type A" and process of nickeling as a "type B". The Table 2 and the Table 3 define the processes sequence and quality constraints for process, A and B respectfully. All pick up and put down times are 8 seconds.

Table 2: The process A definition
The step number 10, "Chroming" lasts extensively longer than other steps. For that reason, the chroming bath is multiplied to three workstations.

Table 3. The process B definition
To calculate the time required to move a hoist from the workstation, i to j the distance between workstation centers is divided by the maximum hoist speed and rounded up to whole seconds. In order to transport an item from a group hoist need to go to the group center, pick the item up (8 seconds), move to the destination bath and put the item down (8 seconds). In Table 4 positions of workstations are defined to calculate the time of hoist movement during item transfer.

Table 4. The Workstation Centers Positions
We performed four tests on the defined production line. The first test was for several queues and assumption that we create an initial schedule, so we solve some Local Problem. The second test is similarly the crating initial schedule, but we consider the random sequence from 40 items in two different quantity ratios: 16 products A plus 24 products B; 18 products A plus 22 products B. The third test
checks how the system works for same quantity ratios, but orders (next segment) are specified in real-time, 100 seconds before the last item of the current queue is loaded to the line. In the fourth test orders are specified in the time when the last item of the current queue is loaded to the line. All calculations are performed on Intel Q9300 2.5GHz processor. Results for random sequences are calculated from 100 instances.

Table 5. Queues analyzed in first test
In order to show how much time can be gained using calculated schedule, we compare their length to a base solution length. To calculate the base solution length, we sum up all the products minimum times, pick up times, putdown times, and transition times required to produce items from the given queue. The utilization ratio is a length of base solution divided by a length of schedule.

Table 6 Ratio 16:24; The Scenario Selection

Table 7 Ratio 18:22; The Scenario Selection
In all analyzed cases, during real-time tests, there was no need establish new adaptation time because all calculations managed to be completed before outdating.

Conclusion
The cyclogram unfold method generates high performance results both for benchmarks and Wrocław production line. It must be pointed out that, quality of unfolded cyclograms used during scheduling have big influence on the result. This can be observed in Lamothe 1996 case. Only one type of item is produced and all items are available from scratch. In this case cyclogram unfold method algorithm is simplified only to one segment. Only one segment is used without the need of finding
any offsets. One segment is directly created by unfolding cyclogram. This leads to a conclusion, that performance of a result is directly connected to cycle time of used cyclogram. Used cyclogram had cycle time sufficiently shorter to gain advantage over the Lamothe 1996 method. All cyclograms used in benchmarking were generated by the method presented in paper Kujawski 2007.

An efficiency of production dynamically driven by orders may be high enough to introduce such production organization to electroplating lines. The main economic consequence is that dynamic scheduling has low requirements conforming to time extent of orders. This eases middle term planning, as it constrains orders minimally. Additionally, experiments show that dynamic hoist scheduling is superior to cyclic scheduling because schedules manage to achieve higher level of resource utilization thanks to parallel multiple item type processing. Cyclic scheduling in the most cases requires sequential item processing of different types, while dynamic scheduling takes advantage of different item types produced together on mutually exclusive resources. Nevertheless, further research is needed in order to identify such superiority cases.

References
Che, A., Chu, C., 2007, Cyclic hoist scheduling in large real-life electroplating lines, OR Spectrum, 29 (3), 445–470
Geismar, H. N., Pinedo, M., Sriskandarajah, C., 2008, Robotic cells with parallel
machines and multiple dual gripper robots: a comparative overview, *IIE
Transactions*, 40 (12), 1211–1227.

Hindi, K., Fleszar, K., 2004, A constraint propagation heuristic for the single-hoist,
multiple-products scheduling problem, *Computers & Industrial Engineering*

system for solving the reactive hoist scheduling problem, *Expert Systems with
Applications*, 30, 156–167.

Kujawski, K., 2007 Zagadnie nie optymalizacji działania wybranych linii
produkcjnych, Dissertation, Wrocław University of Technology, Poland.

Kujawski, K., Świątek, J., 2009 Using Cyclogram Unfolding in Dynamic Hoist

Lamothe, J., Thierry, C., Delmas, J., 1996, A multihoist model for the real time hoist
scheduling problem, *Proceedings of the Computational Engineering in
Systems Application Multiconference (CESA’96)-Symposium on Discrete

production lines with multiple hoists and general processing sequences, *IEEE

Liu, J., Zhou, Z., 2008, A heuristic algorithm for the two-hoist cyclic scheduling
problem with overlapping hoist coverage ranges, *IIE Transactions*, 40, 782–
794.

Liu, J., Jiang, Y., Zhou, Z., 2002, Cyclic scheduling of a single hoist in extended
electroplating lines: a comprehensive integer programming solution, *IIE
Transactions*, 34, 905–914.

Mak, R.T., Gupta, S.M., Lam, K., 2002, Modeling of material handling hoist
operations in a PCB manufacturing facility, *Journal of Electronics

Paul, H., Bierwirth, C., Kopfer, H., 2007, A heuristic scheduling procedure for multi-
item hoist production lines, *International Journal of Production Economics*

Phillips, L.W., Unger P.S., 1976, Mathematical programming solution of a hoist

Subaï, C., Baptiste, P., Niel, E., 2006, Scheduling issues for environmentally
responsible manufacturing: The case of hoist scheduling in an electroplating


Yan P., Chu C., Che A., Yang N., 2008, Algorithm for optimal cyclic scheduling in a
robotic cell with flexible processing times, *IEEE-Proceedings of International
Conference on Industrial Engineering and Engineering Management,

Yan, P., Chu, C., Yang, N., Che, A., 2009, A branch and bound algorithm for optimal
cyclic scheduling in a robotic cell with processing time windows,
*International Journal of Production Research*,
DOI: 10.1080/00207540903225205

Figure 1 Electroplating Line

Figure 2 Outline of dynamic scheduling system
Figure 3 Cyclogram of two hoists with one group of baths (6, 7)

Figure 4 Cyclogram from Figure 3 unfold, 5 products, 9 cycles
Table 1. Comparison to literature benchmark problems

<table>
<thead>
<tr>
<th>Type of benchmark</th>
<th>Result from literature</th>
<th>Results of Cyclogram Unfolding</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jegou 2006</td>
<td>62 products in 8:00:00 hours</td>
<td>108 products in 8:00:00 hours</td>
</tr>
<tr>
<td></td>
<td></td>
<td>62 products in 4:44:00 hours</td>
</tr>
<tr>
<td>Lamothe 1996</td>
<td>00:04:11 hours</td>
<td>00:03:29 hours</td>
</tr>
<tr>
<td>(sequence 1,2,3,…,13)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lamothe 1996</td>
<td>00:04:55 hours</td>
<td>00:04:02 hours</td>
</tr>
<tr>
<td>(sequence 1,5,6,…,1)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 2: The process A definition

<table>
<thead>
<tr>
<th>No. Step</th>
<th>Step name</th>
<th>Min. time</th>
<th>Max. time</th>
<th>Group No.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Loading station</td>
<td>-</td>
<td>-</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>Chem. degreasing</td>
<td>300</td>
<td>420</td>
<td>10</td>
</tr>
<tr>
<td>3</td>
<td>Dripping</td>
<td>30</td>
<td>90</td>
<td>11</td>
</tr>
<tr>
<td>4</td>
<td>Electrochem. degr.</td>
<td>60</td>
<td>180</td>
<td>12</td>
</tr>
<tr>
<td>5</td>
<td>Warm rinse</td>
<td>30</td>
<td>90</td>
<td>13</td>
</tr>
<tr>
<td>6</td>
<td>Rinse I</td>
<td>20</td>
<td>80</td>
<td>14</td>
</tr>
<tr>
<td>7</td>
<td>Cascade rinse I</td>
<td>20</td>
<td>80</td>
<td>15</td>
</tr>
<tr>
<td>8</td>
<td>Cascade rinse II</td>
<td>20</td>
<td>80</td>
<td>16</td>
</tr>
<tr>
<td>9</td>
<td>Anode etching</td>
<td>150</td>
<td>210</td>
<td>9</td>
</tr>
<tr>
<td>10</td>
<td>Chroming</td>
<td>1200</td>
<td>1300</td>
<td>8(3)</td>
</tr>
<tr>
<td>11</td>
<td>Salvaging rinse</td>
<td>20</td>
<td>80</td>
<td>7</td>
</tr>
<tr>
<td>12</td>
<td>Rinse II</td>
<td>20</td>
<td>80</td>
<td>6</td>
</tr>
<tr>
<td>13</td>
<td>Rinse with chrome reduction</td>
<td>20</td>
<td>80</td>
<td>5</td>
</tr>
<tr>
<td>14</td>
<td>Rinse III</td>
<td>20</td>
<td>80</td>
<td>4</td>
</tr>
<tr>
<td>15</td>
<td>Timed rinse.</td>
<td>20</td>
<td>80</td>
<td>3</td>
</tr>
<tr>
<td>16</td>
<td>Blow in bath</td>
<td>30</td>
<td>90</td>
<td>2</td>
</tr>
<tr>
<td>17</td>
<td>Unloading station</td>
<td>-</td>
<td>-</td>
<td>1</td>
</tr>
</tbody>
</table>
Table 3. The process B definition

<table>
<thead>
<tr>
<th>No.</th>
<th>Step Name</th>
<th>Min. time</th>
<th>Max. time</th>
<th>Group No.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Loading station</td>
<td>-</td>
<td>-</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>Timed rinse</td>
<td>300</td>
<td>400</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>Salvaging rinse I</td>
<td>30</td>
<td>90</td>
<td>5</td>
</tr>
<tr>
<td>4</td>
<td>Salvaging rinse II</td>
<td>30</td>
<td>90</td>
<td>7</td>
</tr>
<tr>
<td>5</td>
<td>Degreasing I</td>
<td>60</td>
<td>180</td>
<td>10</td>
</tr>
<tr>
<td>6</td>
<td>Electrochem. degr.</td>
<td>20</td>
<td>80</td>
<td>12</td>
</tr>
<tr>
<td>7</td>
<td>Rinse I</td>
<td>20</td>
<td>80</td>
<td>14</td>
</tr>
<tr>
<td>8</td>
<td>Cascade rinse I</td>
<td>20</td>
<td>80</td>
<td>16</td>
</tr>
<tr>
<td>9</td>
<td>Cascade rinse II</td>
<td>20</td>
<td>80</td>
<td>15</td>
</tr>
<tr>
<td>10</td>
<td>Warm rinse</td>
<td>150</td>
<td>210</td>
<td>13</td>
</tr>
<tr>
<td>11</td>
<td>Nickeling</td>
<td>200</td>
<td>400</td>
<td>11</td>
</tr>
<tr>
<td>12</td>
<td>Degreasing II</td>
<td>20</td>
<td>80</td>
<td>10</td>
</tr>
<tr>
<td>13</td>
<td>Rinse II</td>
<td>20</td>
<td>80</td>
<td>6</td>
</tr>
<tr>
<td>14</td>
<td>Rinse III</td>
<td>20</td>
<td>80</td>
<td>4</td>
</tr>
<tr>
<td>15</td>
<td>Blow in bath</td>
<td>30</td>
<td>90</td>
<td>2</td>
</tr>
<tr>
<td>16</td>
<td>Unloading station</td>
<td>-</td>
<td>-</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 4. The Workstation Centers Positions

<table>
<thead>
<tr>
<th>Group No.</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Position (m)</td>
<td>0.3</td>
<td>0.94</td>
<td>1.56</td>
<td>2.18</td>
<td>2.8</td>
<td>3.43</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Group No.</th>
<th>7</th>
<th>8(1)</th>
<th>8(2)</th>
<th>8(3)</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Position (m)</td>
<td>4.08</td>
<td>4.85</td>
<td>5.75</td>
<td>6.65</td>
<td>7.46</td>
<td>8.14</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Group No.</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>15</th>
<th>16</th>
</tr>
</thead>
<tbody>
<tr>
<td>Position (m)</td>
<td>8.76</td>
<td>9.42</td>
<td>10.08</td>
<td>10.7</td>
<td>11.27</td>
<td>11.79</td>
</tr>
</tbody>
</table>
Table 5. Queues analyzed in first test

<table>
<thead>
<tr>
<th>Queue</th>
<th>Items sequence</th>
<th>Schedule length</th>
<th>Utilization ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>Queue 1</td>
<td>AABBAABB</td>
<td>2:41:36</td>
<td>1.81</td>
</tr>
<tr>
<td>Queue 2</td>
<td>7xA;5xB;5xA</td>
<td>3:28:48</td>
<td>2.95</td>
</tr>
<tr>
<td>Queue 3</td>
<td>20xA;15xB</td>
<td>4:34:51</td>
<td>4.64</td>
</tr>
<tr>
<td>Queue 4</td>
<td>5xA;5xB;5xA;5xB;5xA</td>
<td>5:11:12</td>
<td>2.93</td>
</tr>
<tr>
<td>Queue 5</td>
<td>12xB;4xA;6xB;2xA</td>
<td>4:02:48</td>
<td>3.65</td>
</tr>
<tr>
<td>Queue 6</td>
<td>8xB;8xA</td>
<td>2:33:24</td>
<td>3.81</td>
</tr>
</tbody>
</table>

Table 6 Ratio 16:24; The Scenario Selection

<table>
<thead>
<tr>
<th>Test Type</th>
<th>Schedule Length</th>
<th>Utilization Ratio</th>
<th>Scheduling Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test 2</td>
<td>10:52:58±0:39:38</td>
<td>1.81±0.20</td>
<td>00:14:19±0:05:36</td>
</tr>
<tr>
<td>Test 3</td>
<td>13:22:23±0:18:04</td>
<td>1.47±0.03</td>
<td>0:00:27 per segment</td>
</tr>
<tr>
<td>Test 4</td>
<td>13:19:54±0:16:06</td>
<td>1.48±0.03</td>
<td>0:00:27 per segment</td>
</tr>
</tbody>
</table>

Table 7 Ratio 18:22; The Scenario Selection

<table>
<thead>
<tr>
<th>Test Type</th>
<th>Avg. Schedule Length</th>
<th>Utilization Ratio</th>
<th>Scheduling Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test 2</td>
<td>10:49:09±0:36:00</td>
<td>1.76±0.18</td>
<td>00:16:47±00:10:39</td>
</tr>
<tr>
<td>Test 3</td>
<td>13:00:54±0:12:42</td>
<td>1.46±0.03</td>
<td>0:00:26 per segment</td>
</tr>
<tr>
<td>Test 4</td>
<td>12:59:30±0:14:19</td>
<td>1.46±0.03</td>
<td>0:00:26 per segment</td>
</tr>
</tbody>
</table>