Periodic assignment and graph colouring

EINDHOVEN UNIVERSITY OF TECHNOLOGY

Department of Mathematics and Computing Science

COSOR-Memorandum 91-14

Periodic assignment and graph colouring

by

J. Korst, E. Aarts, J.K. Lenstra, J. Wessels

ISSN: 0926-4493

June, 1991

Periodic Assignment and Graph Colouring

Jan Korst[1], Emile Aarts[1,2], Jan Karel Lenstra[2,3] and Jaap Wessels[2]

1. Philips Research Laboratories, P.O. Box 80.000, 5600 JA Eindhoven, the Netherlands
2. Eindhoven University of Technology, P.O. Box 513, 5600 MB Eindhoven, the Netherlands
3. CWI, P.O. Box 4079, 1009 AB Amsterdam, the Netherlands

Abstract

We analyse the problem of executing periodic operations on a minimum number of identical processors under different constraints. The analysis is based on a reformulation of the problem in terms of graph colouring. It is shown that different constraints result in colouring problems that are defined on different classes of graphs, viz. interval graphs, circular-arc graphs and periodic-interval graphs. We discuss the colouring of such graphs in detail.

Keywords: periodic assignment, graph colouring, interval graphs, circular-arc graphs, periodic-interval graphs

1 Introduction

In this paper we consider the periodic assignment problem, i.e., the problem of assigning periodic operations to processors. Operations are called periodic if they have to be repeatedly executed at a constant rate over an infinite-time horizon. Here, we assume that the executions of periodic operations have fixed start times and that they have to be assigned to a minimum number of processors. The more general problem of finding start times that minimize the number of processors is discussed in [Korst, Aarts, Lenstra & Wessels, 1991]. The periodic assignment problem naturally arises in such diverse areas as real-time processing, vehicle scheduling, and compiler design [Liu & Layland, 1973; Orlin, 1982; Park & Yun, 1985].

We analyse the periodic assignment problem under different constraints, resulting in a graph colouring formulation of the problem for three different classes of graphs. The problem of colouring the vertices of a graph with a minimum number of colours such that adjacent vertices receive different colours, is known to be NP-hard for arbitrary graphs. Furthermore, no efficient approximation algorithm is known that colours arbitrary graphs within a constant factor of the optimum. We show that the graphs in some of the classes of graphs related to periodic assignment are less difficult to colour.

The organization of the paper is as follows. In Section 2 we introduce some basic concepts and notation, discuss periodic assignment and show its relation to colouring graphs from three classes of graphs, viz. interval graphs, circular-arc graphs, and periodic-interval graphs. In Section 3 we consider the computational complexity of colouring these graphs and discuss appropriate graph colouring algorithms.

2 Periodic Assignment

A periodic operation is an operation that is repeately executed at a constant rate over an infinite-time horizon. Its executions are considered to be nonpreemptive. Hence, a periodic operation can be viewed as an infinite sequence of executions of identical length that are equally spaced in time. A periodic operation o is characterized by an *execution time* $e(o) \in \mathbb{N}$, denoting the length of each execution, and a *period* $p(o) \in \mathbb{N}$, denoting the time between the start times of two successive executions. We assume that $e(o) \leq p(o)$ for each periodic operation o.

The executions of a periodic operation o are all uniquely determined in time[1] by a *reference time* $r(o) \in \{1,\ldots,p(o)\}$ that specifies the start time of the execution of o that starts in the interval $[1,p(o)]$. Note that $r(o)$ is well defined, since exactly one execution is started in $[1,p(o)]$. The executions of operation o are started at times $r(o)+k\ p(o)$, $k \in \mathbb{Z}$. For a given set of periodic operations $O = \{o_1,\ldots,o_n\}$, a schedule $S = (r(o_1),\ldots,r(o_n))$ determines the start times of all executions. A schedule S is periodic with period $P = \text{lcm}(p(o_1),\ldots,p(o_n))$, which means that P is the smallest positive number such that for each time $t \in \mathbb{Z}$ and each operation $o \in O$ we have

operation o is executed at t if and only if it is executed at $t + P$.

The periodic assignment problem is now defined as follows. Given a set of periodic operations O and a corresponding schedule S, assign the executions of the operations in O to a minimum number of identical processors, assuming that a processor can execute only one operation at a time.

Given a periodic schedule S with period P, we define the *thickness function* $T_S : [1,P] \rightarrow \mathbb{N}$ as the function that assigns to each time $t \in [1,P]$ the number of operations that are executed at that time. Since a processor can execute only one operation at a time, the maximum thickness $T_S^{\max} = \max_t T_S(t)$ gives a lower bound on the number of processors that is required to execute a given schedule S.

With respect to the assignment of executions to processors we consider two different cases, namely

- *unconstrained periodic assignment*, where different executions of an operation may be assigned to different processors, and
- *constrained periodic assignment*, where all executions of an operation have to be assigned to the same processor.

An assignment is called *periodic with period P'* if P' is the smallest positive integer such that for each time unit $t \in \mathbb{Z}$, each $o \in O$, and each processor m we have

processor m executes operation o at t if and only if m executes o at $t + P'$.

If for a given periodic schedule S with period P an assignment is periodic with period P', then necessarily $P|P'$. For the constrained periodic assignment problem, an assignment is necessarily periodic with a period equal to $\text{lcm}(p(o_1),\ldots,p(o_n))$.

[1] In this paper time is measured in time units, i.e., time periods of equal length. If an operation o with execution time $e(o)$ starts at time t, then it is started at the beginning of time unit t and is completed at the end of time unit $t+e(o)-1$. Similarly, a time interval $[t_1,t_2]$ denotes a set of consecutive time units, given by $\{t_1,t_1+1,\ldots,t_2\}$.

In the following subsections we consider in more detail unconstrained and constrained periodic assignment.

2.1 Unconstrained Periodic Assignment

Before discussing unconstrained periodic assignment, let us first consider as a simple example the assignment problem for a finite set of executions. The assignment problem then amounts to assigning the finite set of executions, with given start and execution times, to a minimum number of processors. This problem can directly be formulated as the problem of colouring the vertices of a graph with a minimum number of colours, by associating with each execution a vertex in the corresponding interval graph. An example of a set of executions, i.e., a set of time intervals, and the corresponding interval graph is given in Figure 1.

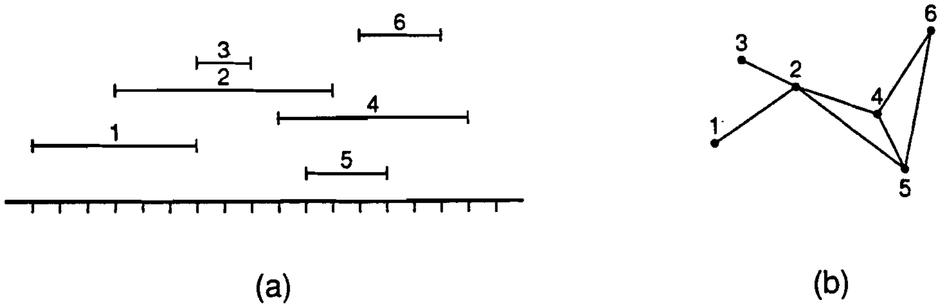

Figure 1: (a) a set of execution intervals and (b) the associated interval graph. The vertices are adjacent if and only if the associated intervals overlap

Definition (interval graph). *A graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is an interval graph if we can associate with each vertex $v_i \in \mathcal{V}$ an interval $[l_i, r_i]$, with $l_i, r_i \in \mathbb{Z}$ and $l_i \leq r_i$, such that $\{v_i, v_j\} \in \mathcal{E}$ if and only if the corresponding intervals $[l_i, r_i]$ and $[l_j, r_j]$ overlap.*

The set of all interval graphs is denoted by $\mathcal{S}_{\text{IG}}$. Colouring interval graphs is discussed in Section 3.1. Here, we restrict ourselves to showing that $T^{\max}$ processors suffice for assigning the set of executions, where the maximum thickness $T^{\max}$ is defined in a similar way as for periodic schedules. To that end, we use the *left edge algorithm* [Hashimoto & Stevens, 1971] and show that this algorithm uses $T^{\max}$ processors to assign the executions. The left edge algorithm first sorts the executions in order of nondecreasing start times and then assigns the executions in this order to the first available processor, i.e., the processor with the smallest index that is idle at the start time of the execution. Now it is easy to show by contradiction that the left edge algorithm uses exactly $T^{\max}$ processors. Suppose that the left edge algorithm uses $T^{\max}+1$ processors. Then at some point in time an execution is assigned to the $(T^{\max}+1)$th processor. But this implies that $T^{\max}$ other executions are being carried out at that time, which contradicts the assumption that $T^{\max}$ gives the maximum thickness. Consequently, the left edge algorithm assigns a finite set of executions to exactly $T^{\max}$ processors.

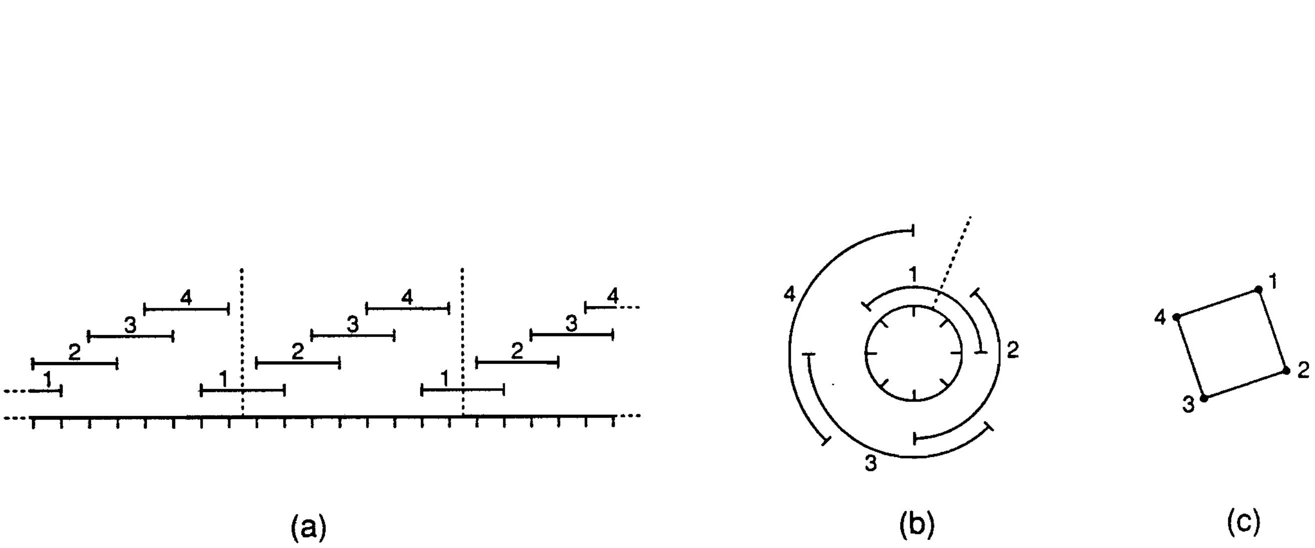

Figure 2: (a) the executions of a set of operations with identical periods, (b) the associated set of circular arcs, and (c) the associated circular-arc graph. Note that the graph is not an interval graph.

Using this result for a finite set of executions, we formulate the following theorem for unconstrained periodic assignment.

Theorem 1 *For a given set of periodic operations $O = \{o_1, \dots, o_n\}$ and a corresponding schedule S, an unconstrained assignment of the executions of O exists that uses only $T_S^{\max}$ processors.*

Proof Let the left edge algorithm be used to assign the executions, starting at time 0. Clearly, from the above result for a finite set of executions we deduce that the left edge algorithm uses $T_S^{\max}$ processors. It remains to be shown that the assignment obtained by the left edge algorithm becomes periodic. The schedule S is periodic with period $P = \mathrm{lcm}(p(o_1), \dots, p(o_n))$. Now consider the time intervals $[1 + lP, (l+1)P]$, $l = 0, 1, \dots$ In each of these intervals the left edge algorithm assigns a finite number of executions to a finite number of processors. Hence, only a finite number of different assignments exist for such intervals. Consequently, the assignment obtained by the left edge algorithm necessarily becomes periodic, using only $T_S^{\max}$ processors. ■

Note that the number of executions for which the left edge algorithm has to specify a processor needs not be polynomial in the number of processors.

2.2 Constrained Periodic Assignment

If all executions of a periodic operation have to be assigned to the same processor, then an assignment is fully determined if for each periodic operation the processor on which it is repeatedly executed is specified. A periodic operation o_i with period $p(o_i)$, execution time $e(o_i)$ and reference time $r(o_i)$, requires an infinite set of time intervals during which it has to be executed. This set is given by $\{[r(o_i)+lp(o_i), r(o_i)+lp(o_i)+e(o_i)-1] \mid l \in \mathbb{Z}\}$. Such an infinite set of intervals is called a *periodic interval* and is denoted by the 3-tuple $(p(o_i), e(o_i), r(o_i))$, with $0 < e(o_i), r(o_i) \leq p(o_i)$.

Let us first consider the special case, where $p(o_i) = p$ for all $o_i \in O$. Clearly, in this case, an assignment is periodic with period p. The periodic assignment problem can then be reformulated as the problem of colouring a circular-arc graph with a minimum number of colours. Figure 2 gives an example of a set of periodic operations and an associated circular-arc graph.

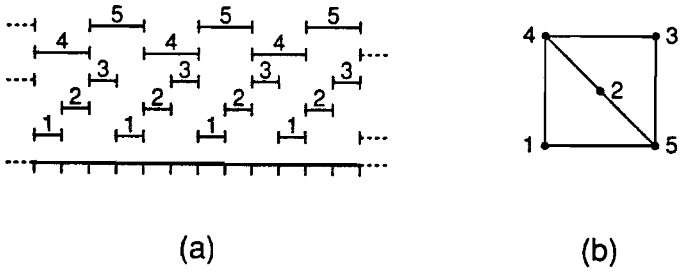

Figure 3: (a) the executions of a set of periodic operations and (b) the associated periodic-interval graph. Note that the graph is not a circular-arc graph.

Definition (circular-arc graph). *A graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is a circular-arc graph if we can associate it with a circle that is divided into a number of segments, numbered clockwise as $1, \ldots, n$, in such a way that each vertex $v_i \in \mathcal{V}$ can be associated with a circular arc $A_i = [l_i, r_i]$, with $l_i, r_i \in \{1, \ldots, n\}$, i.e., an arc on the circle that stretches clockwise from segment l_i to segment r_i, containing both l_i and r_i, and such that $\{v_i, v_j\} \in \mathcal{E}$ if and only if the corresponding arcs $[l_i, r_i]$ and $[l_j, r_j]$ overlap.*

The set of all circular-arc graphs is denoted by $\mathcal{S}_{\text{CAG}}$. In Section 3.2 we extensively examine the problem of colouring circular-arc graphs.

If the operations in O can have arbitrary integral periods, then we can reformulate the periodic assignment problem as the problem of colouring a periodic-interval graph with a minimum number of colours. Figure 3 gives an example of a periodic-interval graph.

Definition (periodic-interval graph). *A graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is a periodic-interval graph if we can associate with each vertex $v_i \in \mathcal{V}$ a periodic interval (p_i, e_i, r_i), with $p_i, e_i, r_i \in \mathbb{N}$ and $0 < e_i, r_i \leq p_i$, such that $\{v_i, v_j\} \in \mathcal{E}$ if and only if the corresponding periodic intervals (p_i, e_i, r_i) and (p_j, e_j, r_j) overlap, i.e., if and only if there exist integers l, m for which*

$$[r_i + lp_i, r_i + lp_i + e_i - 1] \cap [r_j + mp_j, r_j + mp_j + e_j - 1] \neq \emptyset.$$

The set of all periodic-interval graphs is denoted by $\mathcal{S}_{\text{PIG}}$. The following theorem gives a necessary and sufficient condition for the overlap of two periodic intervals.

Theorem 2 [Korst, Aarts, Lenstra & Wessels, 1991] *Two periodic intervals (p_i, e_i, r_i) and (p_j, e_j, r_j) do* not *overlap if and only if*

$$e_i \leq (r_j - r_i) \bmod g_{ij} \leq g_{ij} - e_j,$$

where $g_{ij} = \gcd(p_i, p_j)$.

We end this section with some remarks. From the definitions of interval, circular-arc and periodic-interval graphs it is obvious that

$$\mathcal{S}_{\text{IG}} \subset \mathcal{S}_{\text{CAG}} \subset \mathcal{S}_{\text{PIG}}.$$

Furthermore, the examples given in Figures 2 and 3 show that the inclusions are strict.

Finally,we observe that these classes of graphs can all be considered as intersection graphs, i.e., for each of these graphs we can associate objects with the vertices such that vertices are adjacent if and only if the associated objects intersect or overlap. Intersection graphs can thus be represented in two different ways, either as a graph (i.e., as sets of vertices and edges) or as a collection of associated objects (intervals, circular arcs, periodic intervals). The latter representation is called the *intersection model*. In the following sections we will use both representations interchangeably, since both representations apply to periodic assignment. Furthermore, by using Theorem 2, it follows directly that a graph representation can be constructed from an intersection model in polynomial time. This holds for periodic-interval graphs and hence also for interval and circular-arc graphs. With respect to the inverse transformation, often called the recognition problem, we make the following remarks. Early results on characterizing interval graphs are given by Lekkerkerker & Boland [1962], Gilmore & Hoffman [1964], and Fulkerson & Gross [1965]. Based on these characterizations $O(n^3)$ recognition algorithms can be constructed, with n the number of vertices. An $O(n+m)$ recognition algorithm is given by Booth & Lueker [1976], with n the number of vertices and m the number of edges. A simpler $O(n+m)$ algorithm is given by Korte & Möhring [1989]. Tucker [1980] proved that also circular-arc graphs can be recognized in polynomial time. Recognizing periodic-interval graphs is discussed in Section 3.3.

3 Graph Colouring

In this section we discuss colouring interval graphs, circular-arc graphs and periodic-interval graphs. Let us first summarize some results known for colouring arbitrary graphs. Graph colouring is defined as the problem of colouring the vertices of a graph with a minimum number of colours, such that adjacent vertices receive different colours [Bondy & Murty, 1976]. The minimum number of colours necessary for colouring a graph G is called the chromatic number of G, which is denoted by $\chi(G)$. Graph colouring has been shown to be NP-hard [Karp, 1972], which implies that it is unlikely that there exists a polynomial-time algorithm that colours every graph with $\chi(G)$ colours. Furthermore, Garey and Johnson [1976] showed that if a polynomial-time algorithm exists that colours any graph G with at most $a\chi(G)+b$ colours, with $a < 2$, then there also exists a polynomial-time algorithm that colours each graph G with $\chi(G)$ colours. Consequently, unless $\mathcal{P} = \mathcal{NP}$, no polynomial-time approximation algorithm exists that is guaranteed to use $a\chi(G)+b$ or less colours, with $a < 2$. Furthermore, no polynomial-time approximation algorithm is known that guarantees to colour each graph G with at most $a\chi(G)+b$ colours, for any fixed a and b. And there is evidence that such an algorithm does not exist [Linial & Vazirani, 1989]. The best known performance ratio for a polynomial-time approximation algorithm is $O(n(\log\log n)^3/(\log n)^3)$, where n denotes the number of vertices [Berger & Rompel, 1990]. Hence, graph colouring is not only difficult to solve to optimality, it also seems equally hard to solve to proximity within a constant factor of the optimum.

3.1 Colouring Interval Graphs

As we already showed in Section 2.1, interval graphs can be optimally coloured in $O(n\log n)$ time by the left edge algorithm of Hashimoto & Stevens [1971]. We showed that this algorithm uses $T^{\max}$ colours to colour the vertices of an interval graph.

Given a set of intervals $\{[l_i, r_i] \mid l_i \le r_i, i = 1, \ldots, n\}$ and a set of colours $\{c_1, \ldots, c_n\}$, the algorithm can be restated as follows.

Left Edge Algorithm

1. Sort the intervals in order of nondecreasing left end point.
2. Colour the intervals in this order by assigning to each interval $[l_i, r_i]$ the colour with the smallest index that has not yet been assigned to an interval overlapping $[l_i, r_i]$.

Gupta, Lee & Leung [1979] show that obtaining a minimum number of colours for interval graphs requires $(n \log n)$ time, indicating that the time complexity of the left edge algorithm is optimal.

3.2 Colouring Circular-Arc Graphs

Garey, Johnson, Miller & Papadimitriou [1980] showed that colouring circular-arc graphs is NP-hard. Furthermore, they showed that k-colourability, i.e., the problem of determining whether a circular-arc graph can be coloured with k or less colours, can be solved in $O(n\ k!\ k \log k)$ time. Thus, for fixed k this problem can be solved in polynomial time.

A circular-arc graph is said to be *proper* if none of the corresponding arcs is completely contained in another arc. Proper circular-arc graphs can be coloured with a minimum number of colours in polynomial time. Orlin, Bonuccelli & Bovet [1981] gave an $O(n^2 \log n)$ algorithm which is based on the following observation. For proper circular-arc graphs, k-colourability can be transformed into a shortest path problem which can be solved in $O(n^2)$ time. Combining this with a binary search procedure results in an $O(n^2 \log n)$ algorithm. Successive improvements of this result are presented by Teng & Tucker [1985] and Shih & Hsu [1989], having $O(n^{1.5} \log n)$ and $O(n^{1.5})$ time complexities, respectively.

To the best of our knowledge, Tucker [1975] is the only author that considered approximation algorithms for colouring circular-arc graphs. Here, we consider two approximation algorithms, viz.

(*i*) a generally applicable graph colouring algorithm that was first proposed by Welsh & Powell [1967], and

(*ii*) an extension of an algorithm that was proposed by Tucker, which we call *Sort&Match*.

In the following subsections we formulate both algorithms and examine their worst-case behaviour. To discuss the worst-case behaviour of an approximation algorithm A we introduce the following notation. For a colouring algorithm A, let $A(\mathcal{G})$ be the maximum number of colours A might use when applied to graph $\mathcal{G}$. Then, the performance ratio $R_A(\mathcal{G}) = A(\mathcal{G})/\chi(\mathcal{G})$ gives an upper bound on the relative deviation from the optimum for $\mathcal{G}$.

3.2.1 Algorithm of Welsh & Powell

Let a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and a set of colours $\{c_1, \ldots, c_n\}$ be given. Then the colouring algorithm of Welsh & Powell [1967] can be described as follows:

Algorithm of Welsh & Powell *(W&P)*

1. Sort the vertices in $\mathcal{V}$ in order of nonincreasing degree. The degree $d(v_i)$ of a vertex $v_i \in \mathcal{V}$ gives the number of vertices to which v_i is adjacent.
2. Colour the vertices in this order by assigning to each vertex v_i the colour with the smallest index that has not yet been assigned to a vertex that is adjacent to v_i.

For arbitrary graphs, *W&P* can give results that are arbitrarily far from optimal, i.e., $R_{W\&P}(\mathcal{G})$ has no finite upper bound. Moreover, graphs exist for which the performance ratio $R_{W\&P}(\mathcal{G})$ grows linearly with $|\mathcal{V}|$. This can be seen from the following subset of instances. Let $\mathcal{G}_m = (\mathcal{V}_m, \mathcal{E}_m)$ with $\mathcal{V}_m = \{a_i, b_i \mid 1 \leq i \leq m\}$ and $\mathcal{E}_m = \{\ \{a_i, b_i\} \mid i \neq j\}$. Since all vertices have equal degree, they can be ordered arbitrarily in the first step. If the order of the vertices is $a_1, b_1, a_2, b_2, \ldots, a_m, b_m$, then $W\&P(\mathcal{G}_m) = m$, while $\chi(\mathcal{G}) = 2$. Fortunately, $\mathcal{G}_m$ is not a circular-arc graph, if $m > 3$.

Colouring circular-arc graphs with *W&P* requires less than twice the minimum number of colours, as is shown by the following theorem.

Theorem 3 *For any circular-arc graph $\mathcal{G}$, $R_{W\&P}(\mathcal{G}) < 2$.*

Proof The proof is by contradiction. For convenience we use 'vertices' and 'circular arcs' interchangeably in this proof. Suppose that for some circular-arc graph $\mathcal{G}$ the algorithm of Welsh & Powell requires m colours, with $m \geq 2\chi(\mathcal{G})$. Clearly, in that case some arc a_i receives colour c_m and must consequently be adjacent to at least $m - 1$ other arcs, which receive a colour from $\{c_1, \ldots, c_{m-1}\}$ prior to arc a_i. Let this subset of neighbours of a_i be denoted by $N(a_i)$. Clearly, $d(a_j) \geq d(a_i)$ for each $a_j \in N(a_i)$. Now we consider two cases, viz.

(1) *None of the arcs in* N(a$_i$) *are completely contained in* a$_i$. Then, each of the arcs in $N(a_i)$ covers at least one of the end points of a_i. Hence, one of the end points is covered by at least $\lceil \frac{m-1}{2} \rceil$ arcs. Since $\lceil \frac{m-1}{2} \rceil \geq \chi(\mathcal{G})$, this results in a thickness of at least $\chi(\mathcal{G})+1$. However, this contradicts the fact that $\chi(\mathcal{G})$ is greater than or equal to the maximum thickness.

(2) *One or more arcs in* N(a$_i$) *are completely contained in* a$_i$. Then, there is at least one of these arcs, say arc a_j, such that none of the other arcs in $N(a_i)$ are completely contained in a_j. Since a_j is completely contained in a_i and $d(a_j) \geq d(a_i)$, we conclude that $d(a_j) = d(a_i)$. Consequently, this implies that all arcs that overlap with a_i also overlap with a_j, and vice versa. Hence, one of the end points of a_j is covered by at least $\lceil \frac{m-1}{2} \rceil$ arcs. Again, this leads to a contradiction with the fact that $\chi(\mathcal{G})$ is greater than or equal to the maximum thickness.

Hence, for both cases we have derived a contradiction, which completes the proof of the theorem. ■

We next show that the worst-case performance bound given by Theorem 3 is tight. To that end, we first give the following lemma.

Lemma 1 *For all $m \in \mathbb{N}$, $\gcd(m^2, 2m-1) = 1$.*

Proof Let $a = \gcd(m^2, 2m-1)$. Suppose $a > 1$. Now, if $a | m^2$ and $a | (2m-1)$, then also for any prime factor π of a, $\pi | m^2$ and $\pi | (2m-1)$. But, for any prime number π, if $\pi | m^2$ then $\pi | m$ and if $\pi | m$ then $\pi \nmid (2m-1)$. Consequently, a cannot be greater than 1. ■

Using Lemma 1 we can now prove the following theorem.

Theorem 4 *For any $\varepsilon > 0$, a circular-arc graph $\mathcal{G}$ exists such that $R_{W\&P}(\mathcal{G}) > 2 - \varepsilon$.*

Proof This follows directly from the set of instances defined below. Let $\mathcal{S}_m$ be a set of m^2 arcs on a circle with circumference $2m^2$, with m odd and $m \geq 3$. The arcs are defined by

$$((4lm - 2l) \bmod 2m^2, (4lm - 2l + 2m - 1) \bmod 2m^2), \quad l = 0, 1, \ldots, m^2 - 1.$$

All of these arcs are different, since $\gcd(m^2, 2m-1) = 1$, as shown in Lemma 1. All arcs overlap with $2m - 2$ other arcs. Consequently, all vertices in the corresponding circular-arc graph have the same degree and the arcs are thus coloured in an arbitrary order by *W&P*. If the arcs are coloured in the order as given above, then *W&P* requires $2m - 1$ colours. With each new colour *W&P* colours at most $(m+1)/2$ arcs. However, an optimal colouring requires only m colours. Hence, choosing $m > 1/\varepsilon$ results in a graph that has the required property. This completes the proof of the theorem.. ■

3.2.2 Sort&Match Algorithm

Elaborating on the work of Tucker [1975], we present a two-step approximation algorithm for colouring circular-arc graphs, called *Sort&Match*. For reasons of simplicity the algorithm is formulated in terms of colouring circular arcs instead of vertices.

Sort&Match *(S&M)*

1. Determine on the circle a point t with minimum thickness. Partition the set of arcs into two subsets $\mathcal{A}$ and $\mathcal{B}$, where $\mathcal{A}$ is the set of arcs that cover point t. Hence, $|\mathcal{A}| = T_{\mathcal{S}}^{\min}$. Now, the arcs in $\mathcal{B}$ define an interval graph. Consequently, the arcs in $\mathcal{B}$ can be coloured, using the left edge algorithm, with $T_{\mathcal{S}}^{\max}$ colours.
2. Determine a maximum subset $\mathcal{A}' \subseteq \mathcal{A}$, whose arcs can be coloured with a colour that has already been used in step 1. This problem can be formulated as a maximum-cardinality matching problem in a bipartite graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, with $\mathcal{V} = \mathcal{V}_1 \cup \mathcal{V}_2$, Each vertex $v \in \mathcal{V}_1$ is associated with an arc in $\mathcal{A}$ and each vertex $u \in \mathcal{V}_2$ is associated with a colour that is used in step 1. An edge $\{v, u\}$ is in $\mathcal{E}$ if the arc associated with v can be given the colour associated with u. This matching problem can be solved efficiently using an augmenting path algorithm [Edmonds, 1965; Hopcroft & Karp, 1973]. Finally, each remaining arc in $\mathcal{A} - \mathcal{A}'$ is given a different free colour.

The following theorem states the worst-case performance of *S&M*.

Theorem 5 *For any circular-arc graph $\mathcal{G}$, $R_{S\&M}(\mathcal{G}) \leq 2$.*

Proof Since the arcs in subset $\mathcal{B}$ are coloured with $T_S^{\max}$ colours and the arcs in subset $\mathcal{A}$ are coloured using at most $T_S^{\min}$ colours, we obtain that $S\&M(\mathcal{G}) \leq T_S^{\max} + T_S^{\min} \leq 2T_S^{\max}$. Combining this result with the fact that $\chi(\mathcal{G}) \geq T_S^{\max}$, we obtain the theorem. ■

Tucker [1975] only considers the first step of the algorithm presented above, but essentially proves the same worst-case performance bound. We can again prove that this bound is tight.

Theorem 6 *For any $\varepsilon > 0$, a circular-arc graph $\mathcal{G}$ exists such that $R_{S\&M}(\mathcal{G}) > 2 - \varepsilon$.*

Proof This is directly derived from the following subset of instances. Let S_m be a set of $3m - 3$ arcs on a circle with circumference $6m$, $m \geq 4$, defined by

$(2l, 2m + 2l - 1)$, $\quad l = 0, \ldots, m - 1$,
$(2m + 2l + 2, 4m + 2l + 1)$, $\quad l = 0, \ldots, m - 1$, and
$(4m + 2l + 2, 2l + 1)$, $\quad l = 0, \ldots, m - 4$.

Applying $S\&M$ to S_m results in a colouring with $2m - 3$ colours, while the minimum number of colours is m. Hence, by choosing $m > 3/\varepsilon$, we obtain a graph with the required property. This completes the proof of the theorem. ■

Note that applying $W\&P$ to the above set of arcs may also result in a colouring with $2m - 3$ colours. Hence, choosing the best result of both $S\&M$ and $W\&P$ does not improve the worst case performance ratio of 2.

3.2.3 Experimental Results

In this subsection we present some experimental results that give an indication of the performance of $S\&M$, $W\&P$ and $\min(S\&M, W\&P)$. The results are obtained by applying the algorithms to randomly generated instances. Each instance contains 100 arcs on a circle with a circumference equal to 1. The left end point of an arc is chosen uniformly from the interval $[0, 1)$ and the length of an arc is chosen uniformly from the interval $[minlength, maxlength)$, with $0 \leq minlength \leq maxlength \leq 1$. For different choices of *minlength* and *maxlength*, Table 1 gives the mean error and corresponding mean deviation for $S\&M$, $W\&P$ and $\min(S\&M, W\&P)$. Since the minimum number of colours is unknown, errors are calculated from the cardinality of a maximum clique. The cardinality of a maximum clique clearly gives a lower bound on the minimum number of colours. Although determining the cardinality of a maximum clique is NP-hard for arbitrary graphs, it can be obtained in polynomial time for circular-arc graphs, by iteratively constructing a maximum matching in bipartite graphs [Gavril, 1974]. Consequently, the errors from the minimum number of colours are no more than the given errors. Comparing both algorithms, we see that on average $S\&M$ outperforms $W\&P$ if the arc lengths are small. However, for larger arc lengths $W\&P$ produces better average results than $S\&M$. This motivates the interest in the best result of both algorithms. From Table 1 we observe that the mean error of $\min(S\&M, W\&P)$ remains almost always within 5% of the optimum.

An important exception is given by instances with arc lengths chosen from $[0.3, 0.4)$. Both $S\&M$ and $W\&P$ seem to perform less well for these instances – they give average errors of 27.0% and 16.6%, respectively. These large errors may have been caused by the fact that for

		max. arc length									
		0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
min. arc length	0.0	0.0 0.0	0.6 2.6	1.2 3.2	2.1 3.7	4.3 5.5	6.1 5.7	9.3 4.9	10.3 4.8	8.0 4.0	5.9 3.0
		2.5 4.3	2.8 3.7	2.7 2.9	3.5 2.5	3.5 2.7	3.6 2.5	2.7 2.0	1.9 1.6	1.3 1.4	0.7 1.0
		0.0 0.0	0.3 1.4	0.6 1.7	1.0 1.9	2.0 2.0	2.8 2.6	2.5 1.9	1.8 1.6	1.2 1.3	0.6 1.0
	0.1		0.8 2.5	2.5 5.3	4.2 5.6	4.9 5.6	7.3 5.9	10.1 6.0	9.6 4.3	6.4 3.5	5.1 3.2
			4.9 4.0	6.3 3.7	5.6 3.4	5.6 3.3	4.2 2.5	3.0 2.1	1.4 1.6	0.8 1.1	0.6 0.8
			0.6 1.8	1.7 3.3	2.9 4.0	3.4 3.9	3.4 2.7	2.9 2.2	1.4 1.6	0.8 1.1	0.5 0.8
	0.2			6.0 7.7	4.7 6.7	8.3 7.4	7.3 5.5	8.7 5.0	6.5 3.6	4.4 2.8	3.5 2.4
				5.4 3.9	7.7 4.9	6.8 3.8	3.6 2.2	1.4 1.4	0.8 1.1	0.5 0.8	0.4 0.7
				3.6 4.1	3.8 5.0	5.4 4.7	3.2 2.4	1.2 1.3	0.8 1.1	0.5 0.8	0.4 0.7
	0.3				27.0 13.6	10.2 8.5	5.6 4.3	4.0 3.0	2.4 1.8	1.8 1.6	1.4 1.2
					16.6 7.4	5.6 3.4	1.3 1.4	0.5 0.9	0.2 0.4	0.1 0.4	0.1 0.4
					16.3 7.8	4.5 4.0	1.2 1.4	0.5 0.9	0.1 0.4	0.1 0.4	0.1 0.4
	0.4					1.1 2.1	1.2 1.3	0.4 0.7	0.2 0.5	0.2 0.5	0.0 0.4
						1.8 1.8	0.2 0.5	0.0 0.2	0.0 0.2	0.0 0.1	0.0 0.0
						0.6 1.1	0.1 0.4	0.0 0.1	0.0 0.1	0.0 0.0	0.0 0.0

Table 1: Results obtained by applying *S&M*, *W&P* and min(*S&M*,*W&P*) to randomly generated circular-arc graphs. Each entry in the table gives the means error and mean deviation for *S&M*, *W&P* and min(*S&M*,*W&P*), respectively. for both algorithms. The results of each entry are obtained by applying the algorithms to 100 instances. Each instance contains 100 circular arcs, for which left end points are chosen uniformly from $[0,1)$ and arc lengths uniformly from [min. arc length, max. arc length).

these instances the cardinality of a maximum clique is a bad approximation of the chromatic number. Experimental results, however, contradict this assumption. We have optimally coloured 50 instances from this class each containing 30 arcs. For these instances the chromatic number deviated on average only 1% from the cardinality of the maximum clique.

In addition to the information in Table 1, we mention that, except if arc lengths are chosen from $[0.3, 0.4)$, the observed maximum error for *S&M*, *W&P* and min(*S&M*,*W&P*) is 33.3%, 21.2% and 18.4%, respectively. If the arc lengths are chosen from $[0.3, 0.4)$, then the observed maximum error for *S&M*, *W&P* and min(*S&M*,*W&P*) is 56.4%, 32.5% and 32.5%, respectively.

From the experimental results presented in this subsection, we conclude that the average-case performance of *S&M* and *W&P* is usually much better than the worst-case bounds given in the previous subsections. Furthermore, we conclude that both algorithms perform less well if the lengths of the circular arcs are approximately one third of the circumference of the circle.

3.3 Colouring Periodic-Interval Graphs

Colouring periodic-interval graphs is NP-hard. This follows immediately from the fact that colouring circular-arc graphs is NP-hard and that each circular-arc graph is a periodic-interval graph. The next theorem gives a somewhat surprising result.

Theorem 7 *Each graph is a periodic-interval graph.*

Proof Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be an arbitrary graph. We show that we can associate with each $v_i \in \mathcal{V}$ a periodic interval (p_i, e_i, r_i), such that, for each pair of vertices $v_i, v_j \in \mathcal{V}$, $\{v_i, v_j\} \in \mathcal{E}$ if and only if the associated periodic intervals (p_i, e_i, r_i) and (p_j, e_j, r_j) overlap.

First, construct the complementary graph $\mathcal{G}^c = (\mathcal{V}, \mathcal{E}^c)$, for which $\{v_i, v_j\} \in \mathcal{E}^c$ if $\{v_i, v_j\} \notin \mathcal{E}$, and vice versa. Next, construct all maximal cliques of $\mathcal{G}^c$, denoted by $C_1, C_2, \ldots, C_m$. A maximal clique is a complete subgraph that cannot be enlarged by adding one more vertex. Furthermore, associate with each maximal clique C_i a unique prime number π_i, with $\pi_i \geq n$. We can now associate a periodic interval (p_i, e_i, r_i) with each $v_i \in \mathcal{V}$ according to: $r_i = i$, $e_i = 1$, and $p_i = \prod_{j \in I_i} \pi_j$, where $I_i = \{j \mid v_i \in C_j\}$. Note that $\gcd(p_i, p_j) = 1$ for two periodic intervals (p_i, e_i, r_i) and (p_j, e_j, r_j), for with $\{v_i, v_j\} \in \mathcal{E}$. Consequently, using Theorem 2, they necessarily overlap. Furthermore, for each independent set in $\mathcal{G}$, the greatest common divisor of the periods of the associated periodic intervals is at least n. The associated periodic intervals do not overlap therefore. This completes the proof of the theorem. ■

In Section 1 we stated that no polynomial-time approximation algorithm is known that colours arbitrary graphs within a constant factor of the optimum and that there is evidence that no such algorithm exists. Note that Theorem 7 does not imply that the same holds for periodic-interval graphs, since the transformation from an arbitrary graph to a set of periodic intervals, as presented in the proof of the theorem, is not polynomial. Hence, it remains an open problem whether a polynomial-time algorithm exists that colours periodic intervals within a constant factor of the optimum.

We end this paper with the observation that it also remains open whether or not a polynomial-time algorithm exists for the transformation of a graph representation to an intersection model.

Bibliography

Berger, B. and J. Rompel [1990], A better performance guarantee for approximate graph colouring, *Algorithmica* 5, 459-466.

Bondy, J.A. and U.S.R. Murty [1976], *Graph Theory with Application*, MacMillan Press.

Booth, S. and S. Lueker [1976], Testing for the consecutive ones property, interval graphs, and graph planarity using PQ-tree algorithms, *Journal of Computer and System Sciences* 13, 335-379.

Edmonds, J. [1965], Paths, trees and flowers, *Canadian Journal of Mathematics* 17, 449-467.

Fulkerson, D.R. and O.A. Gross [1965], Incidence matrices and interval graphs, *Pacific Journal of Mathematics* 15, 835-855.

Garey, M.R. and D.S. Johnson [1979], *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W.H. Freeman and Co., San Francisco.

Garey, M.R., D.S. Johnson, G.L. Miller, and C.H. Papadimitriou [1980], The complexity of coloring circular arcs and chords, *SIAM Journal on Algebraic and Discrete Methods* 1, 216-227.

Gavril, F. [1974], Algorithms on circular-arc graphs, *Networks* 4, 357-369.

Gilmore, P.C. and A.J. Hoffman [1964], A characterization of comparability graphs and of interval graphs, *Canadian Journal of Mathematics* 16, 539-548.

Gupta, U.I., D.T. Lee, and J.Y.-T. Leung [1979], An optimal solution for the channel-assignment problem, *IEEE Transaction on Computers* 28, 807-810.

Hashimoto, A. and J. Stevens [1971], Wire routing by optimizing channel assignment with large apertures, *Proceedings of the 8th Design Automation Conference*, 155-169.

Hopcroft, J.E. and R.M. Karp [1973], An $n^{5/2}$ algorithm for maximum matchings in bipartite graphs, *SIAM Journal on Computing* 2, 225-231.

Karp, R.M. [1972], Reducibility among combinatorial problems, in: R.E. Miller and J.W. Thatcher (Eds.) *Complexity of Computer Computations*, Plenum Press, New York, 85-103.

Korst, J.H.M., E.H.L. Aarts, J.K. Lenstra, and J. Wessels [1991], Periodic multiprocessor scheduling, accepted for publication in proceedings of the *Conference on Parallel Architectures and Languages Europe, PARLE '91*, Eindhoven, June 10-13, 1991.

Korte, N. and R.H. Möhring [1989], An incremental linear-time algorithm for recognizing interval graphs *SIAM Journal on Computing* 18, 68-81.

Lekkerkerker, C.G. and J.Ch. Boland [1962], Representation of a finite graph by a set of intervals on the real line, *Fundamenta Mathematicae* 51, 45-64.

Linial, L., and U. Vazirani [1989], Graph products and chromatic numbers, *Proceedings of the 30th Annual IEEE Symposium on Foundations of Computer Science*, 124-128.

Liu, C.L., and J.W. Layland [1973], Scheduling algorithms for multiprogramming in a hard real-time environment, *Journal of the ACM* 20, 46-61.

Orlin, J.B. [1982], Minimizing the number of vehicles to meet a fixed periodic schedule: an application of periodic posets, *Operations Research* 30, 760-776.

Orlin, J.B., M. Bonuccelli, and D. Bovet [1981], An $O(n^2)$ algorithm for coloring proper circular arc graphs, *SIAM Journal on Algebraic and Discrete Methods* 2, 88-93.

Park, K.S., and D.K. Yun [1985], Optimal scheduling ofperiodic activities, *Operations Research* 33, 690-695.

Shih, W.-K., and W.-L. Hsu [1989], An $O(n^{1.5})$ algorithm to color proper circular arcs, *Discrete Applied Mathematics* 25, 321-323.

Teng, A., and A. Tucker [1985], An $O(qn)$ algorithm to q-color a proper family of circular arcs, *Discrete Mathematics* 55, 233-243.

Tucker, A. [1975], Coloring a family of circular arcs, *SIAM Journal on Applied Mathematics* 29, 493-552.

Tucker, A. [1980], An efficient test for circular-arc graphs, *SIAM Journal on Computing* 9, 1-24.

Welsh, D.J.A., and M.B. Powell [1967], An upper bound on the chromatic number of a graph and its application to timetabling problems, *The Computer Journal* 10, 85-87.

EINDHOVEN UNIVERSITY OF TECHNOLOGY
Department of Mathematics and Computing Science
PROBABILITY THEORY, STATISTICS, OPERATIONS RESEARCH
AND SYSTEMS THEORY
P.O. Box 513
5600 MB Eindhoven, The Netherlands

Secretariate: Dommelbuilding 0.03
Telephone : 040-473130

-List of COSOR-memoranda - 1991

Number	Month	Author	Title
91-01	January	M.W.I. van Kraaij W.Z. Venema J. Wessels	The construction of a strategy for manpower planning problems.
91-02	January	M.W.I. van Kraaij W.Z. Venema J. Wessels	Support for problem formulation and evaluation in manpower planning problems.
91-03	January	M.W.P. Savelsbergh	The vehicle routing problem with time windows: minimizing route duration.
91-04	January	M.W.I. van Kraaij	Some considerations concerning the problem interpreter of the new manpower planning system formasy.
91-05	February	G.L. Nemhauser M.W.P. Savelsbergh	A cutting plane algorithm for the single machine scheduling problem with release times.
91-06	March	R.J.G. Wilms	Properties of Fourier-Stieltjes sequences of distribution with support in [0,1).
91-07	March	F. Coolen R. Dekker A. Smit	Analysis of a two-phase inspection model with competing risks.
91-08	April	P.J. Zwietering E.H.L. Aarts J. Wessels	The Design and Complexity of Exact Multi-Layered Perceptrons.
91-09	May	P.J. Zwietering E.H.L. Aarts J. Wessels	The Classification Capabilities of Exact Two-Layered Peceptrons.
91-10	May	P.J. Zwietering E.H.L. Aarts J. Wessels	Sorting With A Neural Net.
91-11	May	F. Coolen	On some misconceptions about subjective probability and Bayesian inference.

COSOR-MEMORANDA (2)

91-12	May	P. van der Laan	Two-stage selection procedures with attention to screening.
91-13	May	I.J.B.F. Adan G.J. van Houtum J. Wessels W.H.M. Zijm	A compensation procedure for multiprogramming queues.
91-14	June	J. Korst E. Aarts J.K. Lenstra J. Wessels	Periodic assignment and graph colouring.