Generalizations of assignments of tasks with interval times

Klaus Jansen*

Abstract

In this paper assignments problem of tasks $t \in T$ to processors with given time intervals $[s(t), e(t)]$ and given partitioning into classes $T = T_1 \cup \ldots \cup T_m$ are analysed. The incompatibility of the tasks is given by the overlapping of the intervals and by the class membership. These both conditions are connected either by a conjunction or by a disjunction. For the second condition the cases are considered that the tasks must be belong to the same class or that the tasks must be belong to different classes. The corresponding incompatibility graphs are generalizations of interval graphs and the assignment problem is equal to the coloring problem.

1 Introduction

We begin with a description of the problems which are analysed in this paper. An interval graph G is a graph where the vertices can be represented by intervals on the real line with an edge between two vertices if their corresponding intervals intersect. Let T be a set of tasks and let $T = T_1 \cup \ldots \cup T_m$ be a partition of this set into task classes. Given a set of discrete times $Z_n = \{1, \ldots, n\}$ and discrete time intervals $Int_n = \{[x, y] | x, y \in Z_n, x \leq y\}$ we have for each task $t \in T$ an interval $[s(t), e(t)] \in Int_n$ in which the tasks are executed. We denote with $s(t)$ and $e(t)$ the start time and end time of the execution of task t. The problem we consider is to find an assignment of the tasks to processors where only compatible tasks are assigned to the same processor and where a minimum number of processors is used. If the compatibility relation depends only on the non-overlapping of the corresponding intervals, we get the coloring problem of an interval graph $G_{int} = (T, \{\{t, t'\} | [s(t), e(t)] \cap [s(t'), e(t')] \neq \emptyset\})$ which can be solved in linear time [4].

We consider generalizations of the compatibility relation which depend on the partition of the tasks in different classes. In addition to the non-overlapping of

*Fachbereich IV, Mathematik und Informatik, Universität Trier, Postfach 3825, W-5500 Trier, Germany

the time intervals we can describe such a relation by a second condition and then we can take the conjunction or the disjunction of both conditions. For the second condition we have two possibilities. Either the tasks t, t' must be belong to the same class T_i or the tasks must be belong to different classes.

If we describe the compatibility relations by graphs we get a classification of the four problems as coloring problems. Given two graphs G_i with the same set of vertices the union $\cup(G_1, G_2)$ is defined by the union of the edges and the intersection $\cap(G_1, G_2)$ by the intersection of the edges sets. We have an edge e in the complement G^c of a graph G if and only if e is not an edge in G. The join of two disjoint graphs $+(G_1, G_2)$ is equal to the complement of the graph $\cup(G_1^c, G_2^c)$. The first compatibility relation can be described by the complement G_{int}^c of the interval graph described above which is called a cointerval graph. If tasks must be belong to the same class, the second relation is described by the disjoint union $\cup_i \overline{T}_i$ of complete graphs with the tasks T_i as vertices. In the other case we get the complement of this graph. The conjunction can be represented as an intersection and the disjunction as an union of these graphs. The classification of the compatibility graphs for the four problems is given in the following table.

	same task class	different task classes
conjunction	$\cap(G_{int}^c, \cup_i \overline{T}_i)$	$\cap(G_{int}^c, (\cup_i \overline{T}_i)^c)$
disjunction	$\cup(G_{int}^c, \cup_i \overline{T}_i)$	$\cup(G_{int}^c, (\cup_i \overline{T}_i)^c)$

We can describe the disjoint union of complete graphs by a set of labels $Z_m = \{1, \ldots, m\}$ and by a labelling of vertices $c : T \to \{1, \ldots, m\}$. Let $T(U, G) = \{t \in T | c(t) \in U\}$ be the set of vertices with labelling $U \subset Z_m$ and let $c(T') = \{c(t) | t \in T'\}$ be the set of labels of the vertices $T' \subset T$ in G. The subgraph of G induced by a set $T' \subset T$ is denoted by $G|_{T'}$.

To analyse the four task problems we have considered the representation given by these graphs and the coloring problem of the four graph classes. But additionally we have considered the problem of finding a maximum independent set, a maximum clique and the problem of finding a minimum partition into cliques, respectively. We denote the corresponding optimum sizes for a graph G with $\alpha(G), \omega(G), \chi(G)$ and $\kappa(G)$. Karp [9] has shown that these optimization problems are NP-complete for general graphs. Since here only subclasses of all graphs occur, it is important to know how difficult the subproblems are. The complexity results for graph problems given different graph classes are summarized by Johnson [8].

An important graph class are the perfect graphs which are defined as follows. A graph $G = (T, E)$ is perfect if two conditions are satisfied.

$$\begin{array}{ll} (1.) & \omega(G|_{T'}) = \chi(G|_{T'}) \quad \text{for all } T' \subset T \\ (2.) & \alpha(G|_{T'}) = \kappa(G|_{T'}) \quad \text{for all } T' \subset T. \end{array}$$

From Lovász [10] it is known that only one of both conditions must be satisfied, because (1.) and (2.) are equivalent. Using the ellipsoid method Grötschel, Schrijver and Lovász [5] have given polynomial algorithms for the four graph optimization problems. Simpler and combinatorial methods are known for subclasses of the perfect graphs. For an overview we refer to Golumbic [3]

2 Application

In this section we give some practical problems which correspond to the task problems. Let T be a set of operations which describe transfers between registers and functional units. Then each set T_i in the partition into task classes are transfers with the same data value. The compatibility of transfers is described by the disjunction of two relations. Two transfers are compatible if their time intervals are non-overlapped or if they transport the same data value. In this case it is possible to put both transfers on the same bus or interconnecting channel. Therefore the compatibility graph is equal to the union of a cointerval graph and the union of complete graphs. A cograph is a graph which can be generated with disjoint union and join starting with single-vertex graphs. If we consider transfers in a program with if-then-else constructions and with unit-times we get the union of a cograph and the disjoint union of complete graphs [6]. There it was shown that the problem of finding a maximum clique or maximum compatible set of transfers can be solved in $O(|T|^2)$ steps and that the problem of finding a partition into a minimum number of cliques or minimum number of buses is NP-complete.

We get also the union of a cointerval graph and an union of complete graphs in another problem. For that consider a program in which for each $1 \leq i \leq m$ exactly one of operations in T_i is choosen by a case-statement with a control variable x_i. If the m case-statements are executed parallel and independent, two operations are compatible if their time intervals are non-overlapped or if they belong to the same set T_i. In both cases they can not be executed parallel at the same time.

Another situation is given by a program in which we have only one case statement at the beginning with one control variable x where at value $x = i$ the operations in T_i are executed. In this program two operations of different sets T_i, T_j with $i \neq j$ can not be executed parallel. The compatibility graph here can be described by an union of the corresponding cointerval graph and the complement of a disjoint union of complete graphs. The last graph can also be described by a disjoint join of graphs with empty set of edges. The general problem with a hierarchy of finite many independent case-statements is analysed in an acompany paper [7]. In the general case we get an union of a cograph and a co-interval graph and it was shown that the problem of finding a maximum compatible set of operations can be generated in polynomial time and that the

problem of finding an assignment with a minimum number of processors remains NP-complete.

Now consider operations of a branching free program in which each operation has a type, for example an addition or a multiplication. If we consider processors which can only execute one of these operation types for example an adder or a multiplier, only two operations of the same type can be executed on the same processor. In this case operations are compatible if their time intervals are non-overlapped and if we have the same type. Therefore the compatibility graph is given as intersection of a cointerval graph and a disjoint union of complete graphs.

At last we give a job assignment problem. Given a partition of the jobs into job classes with different time durations we can search for a fair assignment of the jobs to workers. We call an assignment fair if each worker must only execute at most one job of each class. In this case we get as compatibilty graph an intersection of a co-interval graph and a disjoint join of graphs with empty set of edges.

3 Union of G_{int}^c and $\cup_i \overline{T}_i$

In this section we consider graphs which are given as union of a cointerval graph and a disjoint union of complete graphs. With this operation we can generate each cointerval graph because we can choose as second graph one without edges. But using as an cointerval graph a graph with four vertices, without edges and with task sets $T_1 = \{1, 2\}$, $T_2 = \{3, 4\}$ it is possible to generate a graph with two disjoint edges; a graph which is not a cointerval graph.

At first we consider the clique problem for these graphs. For each interval $[j, k] \in Int_n$ and each task class $i \in Z_m$ we define

$$N_i(j, k) = |\{t \in T(\{i\}, G) | [s(t), e(t)] \subset [j, k]\}|.$$

The value $N_i(j, k)$ gives the number of i - labelled vertices where the corresponding intervals lies in $[j, k]$. With these numbers we can generate the size of the maximum clique in an union of a co-interval graph and a disjoint union of complete graphs. For that we define a weighted interval graph $G_I = (Int_n, A, b)$ with edges $A = \{\{v, w\} | v \cap w \neq \emptyset\}$ and weights for the vertices defined by $b([j, k]) = \max_{1 \leq i \leq m} N_i(j, k)$. Obviously we get the following lemma.

Lemma 3.1 *The size of a maximum clique in $\cup(G_{int}^c, \cup_i \overline{T}_i)$ is equal to a maximum weighted independent set in the weighted interval graph G_I.*

For the computation of the weights we need a computation of the numbers $N_i(j, k)$.

Lemma 3.2 *The numbers $N_i(j, k)$ for all pairs $[j, k] \in Int_n$ and a fixed $i \in Z_m$ can be computed in $O(n^2)$ steps.*

Proof:
We give a recursive formula for the numbers $N_i(j,k)$ for all pairs $[j,k] \in Int_n$ and a fixed $i \in Z_m$ and write for simplification $N(j,k) = N_i(j,k)$. Let $A(j,k) = |\{t \in T | [s(t),e(t)] = [j,k]\}|$, $S(j,k) = \sum_{j \leq j' \leq k} A(j',k)$, and $Z(j,k) = \sum_{j \leq k' \leq k} A(j,k')$. These numbers can be computed in $O(n^2)$ steps. For the numbers $N(j,k)$ we have the following recursive formula.

$$N(j,k) = \begin{cases} N(j+1,k-1) + S(j,k) & \\ Z(j,k) - A(j,k) & \text{if } j+1 \leq k-1 \\ A(j,k) & \text{if } j = k \\ Z(j,k) + A(j+1,k) & \text{if } j+1 = k. \end{cases}$$

□

Theorem 3.3 *The problem CLIQUE given an union of a cointerval graph G^c_{int} and a disjoint union of m complete graphs $\cup_i \overline{T}_i$ can be solved in $O(|T|^2 m)$ steps.*

A simpler computation gives us the size of the maximum independent set. Let $\mathcal{M}_i = \{c(t) | i \in T(t), t \in T\}$ be the different labels for a fixed time step.

Lemma 3.4 *Let $G = \cup(G^c_{int}, \cup_i \overline{T}_i)$ where the second graph is a disjoint union of complete graphs. Then the following formula is satisfied:*

$$\alpha(G) = max_{1 \leq i \leq n} |\mathcal{M}_i|.$$

Proof:
An independent set in G must be independent in the co-interval graph and in the second graph. The first gives that all vertices must have a common time step and the second that the labels must be different. □

Theorem 3.5 *Given an union of a cointerval graph and a disjoint union of complete graphs the problem INDEPENDENT SET can be solved in linear time.*

Using the classifications for the cliques and the independent sets we show that the constructed graph class forms a subset of the perfect graphs.

Theorem 3.6 *Let G be the union of a cointerval graph and a disjoint union of complete graphs. Then G is perfect.*

Proof:
We prove for G that $\kappa(G) = \alpha(G)$. Since the considered graph class is closed

under the induced subgraph operation we get then that G is perfect. For each graph it is known that $\kappa(G) \geq \alpha(G)$.

Depending on the labelling sets $\mathcal{M}_i$ we prove that $\kappa(G) = \alpha(G)$. For that we construct recursively a partition into cliques. For step one take for each label ℓ in $\mathcal{M}_1$ one clique $\{t \in T | 1 = s(t), \ell = c(t)\}$ with label ℓ. After step one we have $|\mathcal{M}_1| \leq \alpha(G)$ cliques. We assume that there is a partition into k cliques for the first i steps with $k \leq \alpha(G)$. Let $C_1, \ldots, C_k$ be the cliques where the last used label is denoted with $c_{last}(C_i)$. We assume that these labels are different.

Now we consider the labelling set $\mathcal{M}_{i+1}$. If a label $\ell \in \mathcal{M}_{i+1}$ is equal to one of the last labels $c_{last}(C_j)$ we take the corresponding vertices with this label at step $i+1$ into clique C_j. If the condition is not satisfied the corresponding vertices with label ℓ must have start-time $s(t) = i+1$. In this case we must take a set $C_{j'}$ and a last label $\ell' = c(C_{j'})$ which occurres not in $\mathcal{M}_{i+1}$. Then the vertices in C'_j have end-time $e(t) \leq i$ and we can add the vertices with label ℓ to clique $C_{j'}$. If there is no further set $C_{j'}$ we must construct a new clique.

Together we need only $max(k, |\mathcal{M}_{i+1}|)$ cliques and the last labels of the corresponding cliques are all different. Using this induction we get a partition with at most $\alpha(G)$ cliques. Therefore G is perfect. □

Theorem 3.7 *Let G be the union of a cointerval graph and a disjoint union of complete graphs. Then the problems of testing $\kappa(G) \leq k$ and $\chi(G) \leq k$ can be solved in polynomial time.*

Proof:
By using the equality $\kappa(G) = \alpha(G)$ and $\chi(G) = \omega(G)$ for a perfect graph. □

Corollary 3.8 *Let $T = T_1 \cup \ldots \cup T_m$ be a partition into tasks with intervals $[s(t), e(t)]$ for each task. Assume that two tasks are compatible if and only if the time intervals are non overlapped or if the classes of the tasks are the same. Then the problem of finding a maximum compatible set of tasks can be solved in $O(|T|^2 m)$ steps and the problem of finding an assignment into a minimum number of compatible tasks can be solved in $O(|T|m)$ steps.*

4 Union of G^c_{int} and $(\cup_i \overline{T}_i)^c$

In this section we consider graphs which are given as union of a cointerval graph and the complement of a disjoint union of complete graphs. The second graph is also a disjoint join of graphs without edges. Since it is possible to choose as second a graph without edges this graph class contains all cointerval graphs.

But we can show that each constructed graphs G is a cointerval graph. To show this assertion we prove that the complement G^c of such graph which is given

as intersection of an interval graph with a disjoint union of complete graphs is an interval graph. For that we use the equality:

$$G^c = \cap(G_{int}, \cup_i \overline{T}_i) = \cup_i G_{int}|_{T_i}.$$

Since interval graphs are closed under the disjoint union and induced subgraph operation we get an interval graph.

Theorem 4.1 *The class of graphs which are given as union of a cointerval graph and a disjoint join of graphs without edges is equal to the class of all cointerval graphs.*

Corollary 4.2 *The problems INDEPENDENT SET, CLIQUE, COLORING and PARTITION INTO CLIQUES can be solved in linear time for an union of a cointerval graph and a disjoint join of graphs without edges.*

Corollary 4.3 *Let $T = T_1 \cup \ldots \cup T_m$ be a partition into tasks with intervals $[s(t), e(t)]$ for each task. Assume that two tasks are compatible if and only if the time intervals are non overlapped or if the classes of the tasks are different. Then the problem of finding a maximum compatible set of tasks and the problem of finding an assignment into a minimum number of compatible tasks can be solved in linear time.*

5 Intersection of G^c_{int} and $\cup_i \overline{T}_i$

In this section we consider graphs which are given as intersection of a cointerval graph and a disjoint union of complete graphs. This graph class contains all cointerval graphs because we can choose as second graph a complete graph. But using as an cointerval graph a complete graph with four vertices and with task sets $T_1 = \{1, 2\}$ and $T_2 = \{3, 4\}$ it is possible to generate two disjoint edges; a graph which is not a cointerval graph.

But we can show that the constructed graphs are perfect. Since the second graph is an disjoint union of complete graphs we get the equality:

$$\cap(G^c_{int}, \cup_i \overline{T}_i) = \cup_i G^c_{int}|_{T_i}.$$

Since the cointerval graphs are closed under the induced subgraph operation we get the disjoint union of cointerval graph. Cointerval graphs are not closed under the disjoint union but perfect graphs. Therefore we get a subclass of the perfect graphs.

Theorem 5.1 *The problems CLIQUE, INDEPENDENT SET, COLORING and PARTITION INTO CLIQUES can be solved in linear time for an intersection of a cointerval graph and a disjoint union of complete graphs.*

Proof:
There are linear time algorithms for the four problems and the cointerval graphs [4]. Since we have the equalties for the disjoint unions of graphs:

$$\alpha(\cup_i G_i) = \sum_i \alpha(G_i)$$
$$\omega(\cup_i G_i) = max_i\, \omega(G_i)$$

we have also linear time algorithms for the four graph optimization problems applied to the disjoint union of cointerval graphs. □

Corollary 5.2 *Let $T = T_1 \cup \ldots \cup T_m$ be a partition into tasks with intervals $[s(t), e(t)]$ for each task. Assume that two tasks are compatible if and only if the time intervals are non overlapped and if the classes of the tasks are equal. Then the problem of finding a maximum compatible set of tasks and the problem of finding an assignment into a minimum number of compatible tasks can be solved in linear time.*

6 Intersection of G^c_{int} and $(\cup_i \overline{T}_i)^c$

In this section we consider graphs which are given as intersection of a cointerval graph and a disjoint join of graphs with empty set of edges. It is simpler to go over to the complement of this graph because this is the union of an interval graph and a disjoint union of complete graphs.

At first observe that the generated graphs in this class are not all perfect. We get a path of length five by using the intervals $[i, i+1]$ for each vertex $i \in \{1, \ldots, 5\}$ and a cycle of length five if we put the first and last vertex in the same task set $T_1 = \{1, 5\}$ and the other vertices in other sets $T_i = \{i\}$ for $i \in \{2, 3, 4\}$.

Now let us consider the clique problem for $\cup(G_{int}, \cup_i \overline{T}_i)$ and let us assume that n discrete times are used. For each label $i \in \{1, \ldots, m\}$ and each pair of times $j, k \in \{1, \ldots, n\}$ we define numbers

$$N_i(j,k) = \begin{cases} |\{t \in T(\{i\}, G) | [j,k] \subset T(t)\}| & \text{if } j \leq k \\ |\{t \in T(\{i\}, G) | j \geq s(t), k \leq e(t)\}| & \text{otherwise.} \end{cases}$$

With these numbers we get the following formula for the size of a maximum clique in $\cup(G_{int}, \cup_i \overline{T}_i)$.

Lemma 6.1 *Let G be the union of an interval graph G_{int} and a disjoint union $\cup_i \overline{T}_i$ of complete graphs. Then*

$$\omega(G) = \max_{[j,k] \in Int_n, 1 \leq i \leq m} \Big(\sum_{1 \leq h \leq m, h \neq i} N_h(j,k) + max\{N_i(j,k), N_i(j,k)\} \Big).$$

Proof:
Let C be a clique of G. If $C = C_1 \cup \ldots \cup C_m$ with $C_i \neq \emptyset$ and $c(C_i) = \{i\}$, then for each pair $t \in C_j$ and $t' \in C_k$ with $j \neq k$ the intersection of the intervals for t and t' is not empty.

Let $s_i = max\{s(t)|t \in C_i\}$ and $e_i = min\{e(t)|t \in C_i\}$. Then the conditions $s_j \leq e_k$ for all pairs $1 \leq j \neq k \leq m$ are necessary and sufficient in order that C is a clique. If an inequality is not satisfied w.l.o.g $s_1 > e_2$ then the intersection of at least one pair of intervals must be empty and C can not be a clique. For the other direction consider w.l.o.g. $s_1 \leq e_2, s_2 \leq e_1$ and show that each pair of intervals intersect. Let $t \in C_1$, $t' \in C_2$. Using $s_1 \geq s(t)$, $s_2 \geq s(t')$ and $e_1 \leq e(t)$, $e_2 \leq e(t')$ we have $e(t') \geq e_2 \geq s_1 \geq s(t)$ and $e(t) \geq e_1 \geq s_2 \geq s(t')$. Therefore the intervals of t and t' intersect.

Now we consider $s_{max} = \max_{1 \leq i \leq m} s_i$ and $e_{min} = \min_{1 \leq i \leq m} e_i$ and the following two cases:

Case 1: $s_{max} \leq e_{min}$.
For each label i the conditions $s_i \leq s_{max} \leq e_{min} \leq e_i$ are satisfied. In this case we can enlarge the clique C using the smaller interval $[s_{max}, e_{min}]$ for each label into a clique C' because $N_i(s_i, e_i) \leq N_i(s_{max}, e_{min})$. The size of a clique of the clique C' is given by $\sum_{i=1}^{m} N_i(s_{max}, e_{min})$.

Case 2: $s_{max} > e_{min}$.
From the definition of s_{max}, e_{min} and from the inequalities $s_j \leq e_k$ for each pair $j \neq k$ we see that there is at least one label i with $e_{min} = e_i$ and $s_{max} = s_i$. For the other labels $\ell \neq i$ we have $s_\ell \leq e_i = e_{min} < s_{max} = s_i \leq e_\ell$. In this case we can also enlarge C into a clique C' using intervals $[e_{min}, s_{max}]$ for each ℓ, becvause $N_\ell(s_{ell}, e_\ell) \leq N_{ell}(e_{min}, s_{max})$. The size of the enlarged clique C' can be computed by $N_i(s_{max}, e_{min}) + \sum_{j=1,(j \neq i)}^{m} N_j(e_{min}, s_{max})$. □

We have also shown another result about each clique C. We can give a pair $[j, k] \in Int_n$ and eventually a label $i \in Z_m$ such that the constructed clique C' in the proof for this parameters contains C. Now we give a formula for the numbers $N_i(j, k)$.

Lemma 6.2 *The numbers $N_i(j, k)$ for all pairs $j, k \in \{1, \ldots, n\}$ and a fixed $i \in \{1, \ldots, m\}$ can be computed in $O(n^2)$ steps.*

Proof:
Without loss of generality we assume that there is only one label and for simplification we denote $N(j, k) = N_i(j, k)$. We consider two cases: $j \leq k$ and $j > k$. In the first case we can give a recursive structure for the numbers $N(j, k)$. Let $A(j, k) = |\{t \in T|s(t) = j, e(t) = k\}|$, $S(j, k) = \sum_{1 \leq j' \leq j} A(j', k)$,

$Z(j,k) = \sum_{k \le k' \le n} A(j,k')$. These numbers can be computed in $O(n^2)$ steps and the numbers $N(j,k)$ by the following formula:

$$N(j,k) = \begin{cases} N(j-1,k+1) + S(j,k) & \\ +Z(j,k) - A(j,k) & \text{if } 1 < j \le k < n \\ Z(1,k) & \text{if } 1 = j \\ S(j,n) & \text{if } k = n \end{cases}$$

In the other case $N(j,k) = |T| - |\{t \in T, e(t) < k\}| - |\{t \in T, s(t) > j\}|$. This computation can be done also in $O(n^2)$ steps. □

Theorem 6.3 *Let G be the union of an interval graph G_{int} and a disjoint union $\cup_i \overline{T}_i$ of m complete graphs. Then the clique problem for G can be solved in $O(|T|^2 m)$ steps.*

Now we consider the problem of finding a maximum independent set in G. Each independent set in G must be independent in the interval graph and in the other graph. This means, that we search here for a chain of different labelled and non overlapping intervals in the interval graph. If follows directly that the size $\alpha(G)$ can be bounded by the number of labels m. Let $I(U) = \cup_{t \in U} [s(t), e(t)]$ be the union of the intervals for a set $U \subset T$ and let

$$\mathcal{M}_i = \{c(U) | U \subset T \text{ independent with } I(U) \subset [1,i]\} \subset 2^m$$

be the set of all different labelled chains between time step 1 and i. Then it is easy to prove the following recursive formula.

Lemma 6.4

$$\begin{aligned} \mathcal{M}_0 &= \emptyset \\ \mathcal{M}_{i+1} &= \mathcal{M}_i \cup \{M \cup \{c(t)\} | s(t) = k, e(t) = i+1, M \in \mathcal{M}_{k-1}\} \end{aligned}$$

If we apply this recursive structure we produce at most $n\, 2^m$ sets and we need at most $O(n^2\, 2^m)$ operations to compute $\mathcal{M}_n$. We see in the next theorem that it is not to expect that we find a maximum independent set in G in polynomial time unless $P = NP$.

Theorem 6.5 *Let G be the union of an interval graph G_{int} and a disjoint union $\cup_i \overline{T}_i$ of m complete graphs. Then the problem of finding a maximum independent set in G can be solved in polynomial time, if m is fixed. In general the problem remains NP-complete.*

Proof:
The first assertion follows directly from the recursive structure in the lemma

before. For the second one we can give a transformation from the satisfiability problem to the problem of finding a maximum independent set in a graph which is the union of an interval graph and a disjoint union of complete graphs. Let α be a formula in conjunctive normalform with $\alpha = c_1 \wedge \ldots \wedge c_m$ and with exactly three literals for each clause: $c_i = (y_{i1} \vee y_{i2} \vee y_{i3})$ and $y_{ij} \in \{x_1, \overline{x}_1, \ldots, x_n, \overline{x}_n\}$. We define an interval graph $G_{int} = (T, E)$ given by intervals $[s(t), e(t)] \subset [1, n(2m+1)]$ for the vertices and a labelling $c : T \to Z_{m+n}$ as follows:

$$
\begin{array}{lcl}
T & = & \{a_i, \overline{a_i} | 1 \leq i \leq n\} \cup \{b_{kj} | 1 \leq k \leq m, 1 \leq j \leq 3\} \\
s(a_i) & = & (i-1)(2m+1)+1 \\
e(a_i) & = & (i-1)(2m+1)+m+1 \\
s(\overline{a_i}) & = & (i-1)(2m+1)+m+1 \\
e(\overline{a_i}) & = & (i-1)(2m+1)+2m+1 \\
s(b_{kj}) & = & \begin{cases} (i-1)(2m+1)+k & y_{kj} = \overline{x_i} \\ (i-1)(2m+1)+m+1+k & y_{kj} = x_i \end{cases} \\
e(b_{kj}) & = & s(b_{kj}) \\
c(t) & = & \begin{cases} m+i & t = a_i, \overline{a_i} \\ k & t = b_{kj} \end{cases}
\end{array}
$$

A maximum independent set U in $\cup(G_{int}, \cup_i \overline{T}_i)$ must be independent in the interval graph and the second graph. Therefore exactly one of the vertices $a_i, \overline{a_i}$ are elements of U and such a choice codes a valuation of the variables $x_i, \overline{x_i}$ in α. If a_i is choosen, then a vertex b_{kj} with $y_{kj} = \overline{x_i}$ can not be an element of U. But it is possible to include vertices with $y_{kj} = x_i$. The second graph gives us that exactly one of the three vertices b_{k1}, b_{k2}, b_{k3} which correspond to a clause must be include in the set U, because U is maximum. Therefore there is an independent set of size $m+n$ in $\cup(G_{int}, \cup_i \overline{T}(\{i\}, G))$ if and only if α is satisfiable. □

Corollary 6.6 *Let $D = (T, A)$ be an interval ordered digraph with labelling $c : T \to Z_m$. The problem of finding a directed path through D of length m with different labels is NP-complete.*

This path search is a subproblem of another problem, which is called PATH WITH FORBIDDEN PAIRS. In that problem there is given a digraph $D = (T, A)$, two specified vertices $t_1, t_2 \in T$ and a collection $C = \{(a_i, b_i) | a_i, b_i \in T, 1 \leq i \leq k\}$. The question is, if there is path from t_1 to t_2 which contains at most one vertex from each pair in C. It was shown [1] that this problem is NP-complete.

Now we analyse the problem CLIQUE PARTITION. We known that each clique C can be enlarged to another clique which is constructed by a pair $[j, k]$ and eventually a label $i \in Z_m$, see Lemma 6.1. Therefore we can also search for a partition into cliques which are computed by these parameters. The combinations for choosing a label and a pair are $O(mn^2)$ and for a partition with k cliques we have $O(m^k n^{2k})$ combinations.

Theorem 6.7 *Let G be the union of an interval graph G_{int} and a disjoint union $\cup_i \overline{T}_i$ of m complete graphs. For fixed m or k the problem of finding a partition into k cliques for G is polynomial solvable. In general the problem remains NP-complete.*

Proof:
From the definition of G we see that $\kappa(G) \leq m$. If m or k are fixed there are at most polynomial many combinations for a partition. For the NP-completness proof we can take a construction given by Garey and Johnson [2] p. 69. The constructed graph there can be represented as a graph in this class. □

At last we consider the coloring problem for these graphs. A partition into k independent sets corresponds to a choose of k chains of different labelled and non overlapping intervals which covers all vertices of the graph. The number of different labelled chains is at most 2^m and the number of different independent sets is at most n^{2^m}. For fixed k we get the following assertion.

Theorem 6.8 *Let G be the union of an interval graph G_{int} and a disjoint union $\cup_i \overline{T}_i$ of m complete graphs. For fixed k the problem of finding a k coloring for G is NP-complete.*

Proof:
We give a transformation from the 3-coloring problem for graphs with maximum degree four which is NP-complete [2] to the 3-coloring problem in graph described above. Let $G = (V, E)$ be an instance of a 3-coloring problem. At first we define for each vertex $v_i \in V$ an interval graph H_i with vertices $\{a_i, b_i, t_{i1}, t_{i2}, t_{i3}, t_{i4}\}$ and intervals $[1, 4]$ for vertices a_i, b_i and $[j, j]$ for the vertices t_{ij}. The graph H_i is three-colorable, but not two-colorable and for each 3-coloring of H_i and for the vertices called outlets with degree two we get

$$f(t_{i1}) = f(t_{i2}) = f(t_{i3}) = f(t_{i4}).$$

Since the interval graphs are closed under the disjoint union, the graph $G_{int} = \cup_i H_i$ is an interval graph.

Now we can construct a labelling for the vertices of G_{int}. Let $E = \{e_1, \ldots, e_m\}$ be the edges of G. Iteratively we take an edge $e_j = \{v_k, v_{k'}\}$ with $v_k, v_{k'} \in V$. In the corresponding graphs $H_k, H_{k'}$ we take the first outlet with degree two and put the both outlets in the set T_j. After that procedure we assign the rest vertices in sets $T_{|E|+1}, \ldots, T_{6|V|-|E|}$ where each set contains exacly one of rest vertices. So we get a graph G' which is an union of an interval graph G_{int} and a disjoint union $\cup_i \overline{T}_i$ of complete graphs. Now we have an edge $\{v_k, v_{k'}\}$ in G iff H_k and $H_{k'}$ are connected by an edge given in the second graph. Therefore we have the equivalence that G is 3-colorable if and only if G' is 3-colorable. □

Corollary 6.9 *Let $T = T_1 \cup \ldots \cup T_m$ be a partition into tasks with intervals $[s(t), e(t)]$ for each task. Assume that two tasks are compatible if and only if the time intervals are non overlapped and if the classes of the tasks are different. Then the problem of finding a maximum compatible set of tasks is NP-complete, but can be solved in polynomial time for constant m. The decision-problem if there is a partition into three compatible sets of tasks is NP-complete.*

We note that the complexity of the partition problem for constant m is open.

7 Conclusion

At last we give an overview about the complexity of the task problems. Depending on the compatibility relation we get either polynomial algorithms or that the problem is NP-complete.

	same task class	different task classes
conjunction	P	NP complete
disjunction	P	P

References

[1] H.N. GABOW, S.N. MAHESHWARI and L. OSTERWEIL, On two problems in the generation of program test paths, *IEEE Transaction on Software Engineering*, **SE-2** (1976) pp. 227 – 231.

[2] M.R. GAREY and D.S. JOHNSON, "Computers and Intractability: A Guide to the Theory of NP-Completness," Freeman, San Francisco, 1979.

[3] M.C. GOLUMBIC, "Algorithmic graph theory and perfect graphs," Academic Press, New York, 1980.

[4] U.I. GUPTA, D.T. LEE and J.Y.-T. LEUNG, Efficient algorithms for interval graphs and circular arc graphs, *Networks*, **12** (1982), pp. 459 – 467.

[5] M. GRÖTSCHEL, A. SCHRIJVER and L.LOVÁSZ, The ellipsoid method and its consequences in combinatorial optimization, *Combinatorica* **1** (1981), 169-197.

[6] K. JANSEN, Transfer flow graphs, to appear in *Discrete Mathematics*.

[7] K. JANSEN, Processor optimization for flow graphs, to appear in *Theoretical Computer Science* (1992).

[8] D.S. JOHNSON, The NP-completeness column: on ongoing guide, *Journal of Algorithms* **6** (1985), pp. 434 – 451.

[9] R.M. KARP, Reducibility among combinatorial problems, in: "Complexity of Computer Computations," pp. 85 – 103, R.E. Miller and J.W. Thatcher (eds.), Plenum Press, New York, 1972.

[10] L. LOVÁSZ, Normal hypergraphs and the perfect graph conjecture, *Discrete Mathematics* **2** (1972), pp- 253 – 267.