Exploiting the structure of conflict graphs in high level synthesis

Klaus Jansen

Abstract. In this paper we analyze the computational complexity of a processor optimization problem. Given operations with interval times in a branching flow graph, the problem is to find an assignment of the operations to a minimum number of processors. We analyze the complexity of this assignment problem for flow graphs with a constant number of program traces and a constant number of processors.

Keywords: independent set, chromatic number, high level synthesis

Classification: 68R10, 05C15

1. Introduction

High level synthesis is the generation of a register transfer level description from a behaviour description. In one step of the synthesis, operations in a scheduled flow graph are assigned to modules or processors. To do this, a conflict graph for the operations is generated. Then, the assignment problem of the operations to a minimum number of processors is equivalent to the coloring problem of the conflict graph. The coloring problem is NP-complete for general graphs [3]. As a result, heuristics are usually used to perform the coloring [8], [11]. The cited heuristics are intended for general graphs, but the conflict graphs are not necessarily general graphs.

Therefore, the first goal is to get a complete classification of the generated graphs. Using a model of a flow graph with independent branches, some generated graph classes are known [4], [5]. In this paper, we consider a more general case with interdependence between the branches. Then, in general, all undirected graphs can be generated, but usually we have further restrictions on the problem instances with

- a constant number of execution traces (a trace is a set of operations executed in dependence to the control of the branches),
- a constant number of processors,
- one execution step assigned to each operation.

After the analysis of the structure of the corresponding conflict graphs, we consider two optimization problems restricted to the corresponding graph class. The first problem is to compute a maximum compatible set of operations, and the second relates with the assignment of the operations to a minimum number of processors.

The paper is organized as follows. Section 2 gives a collection of new graph theoretical results for the independent set and the coloring problem for restricted graph classes. These results are used later for the processor optimization problem. In Section 3 we give a classification of the conflict graphs which correspond to flow graphs with a constant number of execution traces. Then, we consider the optimization problems mentioned above for the classified graph classes.

2. Graph theoretical results

Given a graph $G = (V, E)$, a set $U \subset V$ is independent iff each pair $v, v' \in U$, $v \neq v'$ is not connected by an edge. The size of a maximum independent set of a graph G is called the independence number of G and is denoted by $\alpha(G)$. A k-coloring of a graph G is a mapping $f : V \to \{1, \dots, k\}$ with $f(v) \neq f(v')$ for each pair $v, v' \in V$, $v \neq v'$ with $\{v, v'\} \in E$. The minimum number of colors to color a graph is called the chromatic number of G and is denoted by $\chi(G)$.

Interval graphs are graphs that can be modeled as a set of intervals on the real line, with an edge between two vertices if their corresponding intervals share a point. A graph $G = (V, E)$ is complete, if each pair of vertices $v, v' \in V$, $v \neq v'$ is connected by an edge $\{v, v'\} \in E$. Let $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ be graphs. Their union $\cup(G_1, G_2)$ and their intersection $\cap(G_1, G_2)$ is defined by:

$$\cup(G_1, G_2) = (V_1 \cup V_2, E_1 \cup E_2),$$
$$\cap(G_1, G_2) = (V_1 \cap V_2, E_1 \cap E_2).$$

For a sequence of graphs $G_1 = (V_1, E_1), \dots, G_m = (V_m, E_m)$ the graph $G = (V, E) = \cup(G_1, \dots, G_m)$ is obtained by iterative union of the graphs $G_1, \dots, G_m$.

A path decomposition, see [9], of a graph $G = (V, E)$ is a pair (X, I) with a family $X = \{V_i | i \in I\}$ of subsets of V and a linear list $I = [1, \dots, m]$ such that

- $\cup_{i \in I} V_i = V$,
- for each $\{x, y\} \in E$ there is an $i \in I$ with $\{x, y\} \subset V_i$,
- for each $x \in V$, the set $I_x = \{i \in I \mid x \in V_i\}$ forms a subinterval of I.

The pathwidth of a path decomposition is defined as $\max_{i \in I} |V_i| - 1$ and the pathwidth of a graph G is the minimum pathwidth over all path decompositions of G. We note that a path decomposition for a graph with constant pathwidth can be found in linear time [2]. For these graphs the following result is proved in [1].

Proposition 2.1. *Let $G = (V, E)$ be a graph with constant pathwidth. Then, the problems of finding a maximum independent set and a minimum coloring are solvable in linear time $O(|V|)$.* □

2.1 Polynomial results

In this subsection, we give a collection of polynomial results for the independent set and coloring problem on graphs constructed by an intersection of two special graphs.

Theorem 2.2. *Let ℓ be a constant and let G be an intersection of an interval graph and a union of ℓ complete graphs. Then, the independence number $\alpha(G)$ can be found in polynomial time.*

PROOF: First, we may assume that each vertex $x \in V$ is assigned an interval $I_x \subset [1, 2 \cdot |V|]$ such that we have an edge between vertices x, y, $x \neq y$ in the interval graph, if and only if $I_x \cap I_y \neq \emptyset$. Let G_i be the subgraph of G induced by the vertices $\{x \in V | i \in I_x\}$. Since the second graph is given as a union of ℓ complete graphs, the independence number $\alpha(G_i) \leq \ell$. Hence, the number of independent sets in each graph G_i can be bounded by the polynomial $O(n^\ell)$.

Given a graph $G = \cup_{1 \leq i \leq m} G_i$ and intervals I_x for each vertex, an acyclic digraph $D = (V', E')$ with positive edge weights will be constructed. The search for a maximum independent set of G is equivalent to the search for a maximum weighted path in the digraph. Such a path in an acyclic digraph can be constructed in $O(|V'|^2)$ steps (see Lawler [6]).

Let $\mathbf{U}_i$ be the collection of all independent sets in G_i, including the empty set. Let G_0 and $G_{2 \cdot n+1}$ be graphs with no vertices, so that $\mathbf{U}_0 = \mathbf{U}_{2 \cdot n+1} = \{\emptyset\}$. The digraph $D = (V', E')$ is defined as follows: Let V' be the disjoint union of the sets $\mathbf{U}_i$ (i.e. an independent set in G appears once for each G_i containing it). We will view the vertex from $\mathbf{U}_0$ as the source s and the vertex from $\mathbf{U}_{2 \cdot n+1}$ as the sink t. For $0 \leq i \leq 2 \cdot n$, E' contains the edges (U, U') if and only if $U \in \mathbf{U}_i$, $U' \in \mathbf{U}_{i+1}$, and if there is an independent set in $G_i \cup G_{i+1}$ whose intersection with V_i and V_{i+1} is U and U', respectively. For each edge (U, U') it follows that vertices which are contained in U and in V_{i+1} are elements of U' and that vertices which are contained in U' and in V_i are elements of U. The weight of an edge (U, U') is $|U' \setminus U|$. Because $\alpha(G_i)$ is less than or equal to a constant ℓ, this digraph can be constructed in polynomial time.

Clearly, each directed path in D has a corresponding independent set of G and each independent set of G gives a path in D. Therefore, $\alpha(G)$ can be computed in polynomial time. □

Theorem 2.3. *Let G_1 be a disjoint union of m complete graphs and let G_2 be a non-disjoint union of a constant number ℓ of complete graphs. Then, the problem of finding $\alpha(G)$ and $\chi(G)$ of the intersection of G_1 and G_2 can be solved in linear and polynomial time, respectively.*

PROOF: Let G_1 be a disjoint union of m complete graphs with vertex sets $K_1, \ldots, K_m$. We define $G^{(i)}$ as the subgraph of G induced by the set K_i. Since G is a disjoint union of the graphs $G^{(i)}$, we get $\alpha(G) = \sum_{i=1}^m \alpha(G^{(i)})$ and $\chi(G) = \max_{1 \leq i \leq m} \chi(G^{(i)})$ Hence, we must consider the problems only for the graphs which are given as a non-disjoint union of at most ℓ, with ℓ constant, complete graphs.

(a) $\alpha(G)$: Let $H = (V, E)$ be a non-disjoint union of ℓ complete graphs with vertex sets H_i. We construct for H a graph $\overline{H}$ with a constant number of vertices (at most 2^ℓ). For each subset $A \subset \{1, \ldots, \ell\}$ such that there is a vertex $x \in V$

with $x \in H_i$ for $i \in A$ and $x \notin H_i$ for $i \notin A$, we take a vertex v_A in the graph $\overline{H}$. The computation of the vertex set of $\overline{H}$ can be done in linear time $O(|V|)$. We connect two vertices v_A and $v_{A'}$ with $A \neq A'$ in $\overline{H}$, if and only if their corresponding subsets are not disjoint. Then, the independence number $\alpha(H) = \alpha(\overline{H})$. Since $\overline{H}$ has only a constant number of vertices, the problem of finding a maximum independent set in $\overline{H}$ is solvable in linear time $O(|V|)$.
(b) $\chi(G)$: We define for each vertex v_A of $\overline{H}$ corresponding to a subset $A \subset \{1, \ldots, \ell\}$ a weight

$$w_A = |\{v \in V \mid v \in V_i, i \in A, v \notin V_{i'}, i' \notin A\}|.$$

The weight w_A gives the number of vertices lying in exactly the complete graphs corresponding to set A. Then, a k coloring of H is equivalent to a collection of k independent sets in $\overline{H}$ — it is allowed to take an independent set several times — such that each vertex v_A in $\overline{H}$ is covered at least w_A times. The graph $\overline{H}$ contains no more than a constant number $\overline{\ell} \leq 2^{2^{\ell}}$ of independent sets $U_1, \ldots, U_{\overline{\ell}}$. Each feasible k-coloring of H or a collection of k independents sets in $\overline{H}$ can be described as a function $f : \{1, \ldots, \overline{\ell}\} \rightarrow \{0, \ldots, k\}$ such that

- For each vertex v_A, $\sum_{1 \leq i \leq \overline{\ell}, v_A \in U_i} f(i) \geq w_A$ and
- $\sum_{1 \leq i \leq \overline{\ell}} f(i) = k$.

Since the number of feasible functions f can be bounded by the polynomial $k^{\overline{\ell}}$, the problem of computing $\chi(H)$ is solvable in polynomial time. □

2.2 NP-completeness

In this subsection we give a NP-completeness result for the coloring problem on a small class of graphs.

Theorem 2.4. *Let G be an intersection of an interval graph and a union of two complete graphs. Then, the problem of finding $\chi(G)$ is NP-complete.*

PROOF: Clearly, any coloring problem is in NP. We give a transformation from the numerical 3-dimensional matching problem (see e.g. [3]) to the coloring problem. An instance of the matching problem is given by three sets $W = \{w_1, \ldots, w_m\}$, $X = \{x_1, \ldots, x_m\}$ and $Y = \{y_1, \ldots, y_m\}$ and a bound $Z \in \mathbb{N}$ such that $\sum_{i=1}^{m} [w_i + x_i + y_i] = m \cdot Z$. The problem is to decide whether there exists a partition of $W \cup X \cup Y$ into m disjoint sets A_i such that each A_i contains exactly one element from each of W, X and Y and such that for each $1 \leq i \leq m$, $\sum_{a \in A_i} a = Z$. We assume that w_m is the largest integer in the set W (i.e. $w_m \geq w_i$ for each $1 \leq i \leq m$).

We first give the construction of a set of intervals for the interval graph.

1. For each $w_i \in W$, take an interval $a_i = [0, w_i + i]$,
2. for each $w_i \in W$, $x_j \in X$, take an interval $b_{i,j} = [w_i + i + 1, w_i + x_j + (m+1) + (j\,Z)]$,

3. for each $x_j \in X$,$y_k \in Y$, take an interval $c_{j,k} = [(j+1)Z - y_k + m + 2, (m+1)(Z+1)+k]$,
4. for each $y_k \in Y$, take an interval $d_k = [(m+1)(Z+1)+k+1, (m+1)(Z+2)+1]$,
5. for each $w_i \in W$, take intervals $e_{i,\ell} = [0, w_i + i]$ for $1 \le \ell \le m-1$,
6. for each $1 \le j \le m$ and $1 \le \ell \le m-1$, take intervals $f_{j,\ell} = [1, jZ+m+1]$ and $g_{j,\ell} = [(j+1)Z + m + 1, (m+1)(Z+2)]$.
7. for each $1 \le k \le m$ and $1 \le \ell \le m-1$, take intervals $h_{k,\ell} = [(m+1)(Z+1)+k+1, (m+1)(Z+2)+1]$,
8. for $1 \le i \le m$, take an interval $p_i = [0, (m+1)(Z+2)+1]$,
9. for $1 \le i \le m(m-1)$, take intervals $r_i = [w_m + m + 1, (m+1)(Z+2)]$, $q_i = [(m+1)(Z+2)+1, (m+1)(Z+2)+1]$, $r'_i = [1, (m+1)(Z+1)+1]$ and $q'_i = [0, 0]$.

An example is shown in Figure 1, where we have $w_1 = 1$, $w_2 = 2$, $x_1 = 1$, $x_2 = 1$, $y_1 = 2$, $y_2 = 3$ and $Z = 5$, $m = 2$. Since $m-1$ is equal to one, the second index for the intervals e, f, g and h is removed.

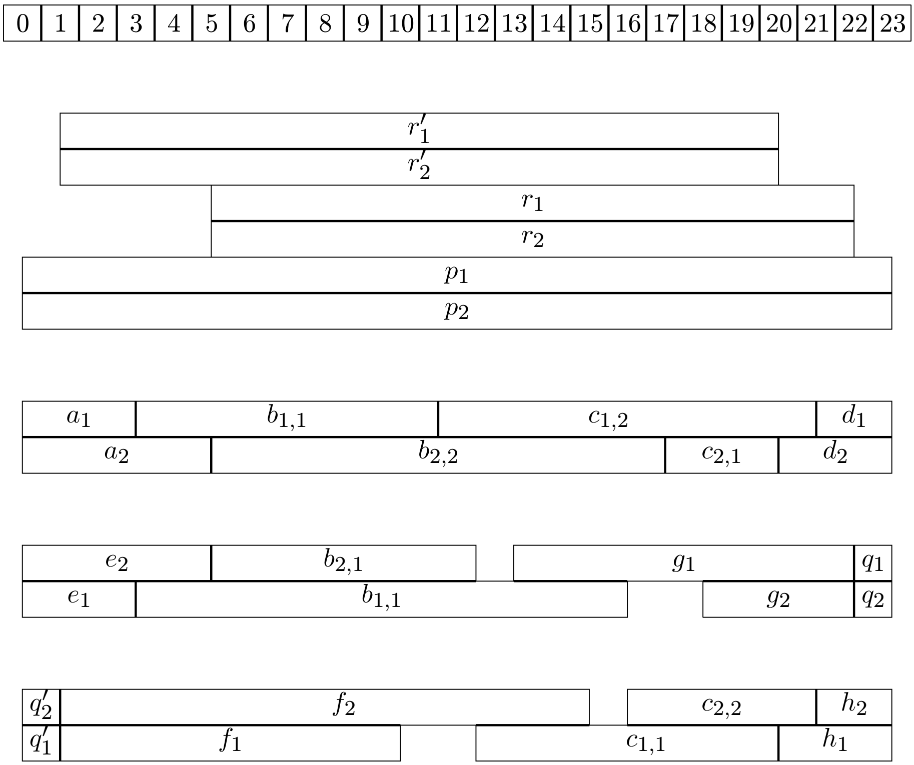

Figure 1: Example for the construction of an interval graph

Denote the set of all intervals a_i $(1 \le i \le m)$ by A. In a similar way, define sets $B, C, D, E, F, G, H, P, R, R', Q$ and Q'; each of these sets contains all intervals with the same letter. First, let us consider which sets of vertices form cliques in the interval graph. These are $A \cup E \cup P \cup Q'$, $A \cup E \cup F \cup P \cup R'$, $B \cup F \cup P \cup R'$, $P \cup R \cup R'$, $C \cup G \cup P \cup R$, $D \cup G \cup H \cup P \cup R$, $D \cup H \cup P \cup Q$ and possibly further sets depending on the instance.

Now we define two complete graphs with the following vertex sets

(1) $A \cup B \cup C \cup D \cup E \cup F \cup G \cup H \cup Q \cup Q'$,

(2) $E \cup H \cup P \cup R \cup R' \cup Q \cup Q'$.

Then, the intersection of the union of these two graphs and the interval graph is the input graph of the coloring problem. The sizes of the sets are $|A| = |D| = |P| = m$, $|B| = |C| = m^2$ and the sizes of the other sets are $|E| = |F| = |G| = |H| = |Q| = |Q'| = |R| = |R'| = m(m-1)$. In total, we get $10m^2 - 5m$ vertices.

In the next part we assume that the input graph has a partition into $2m^2 - m$ independent sets $U_1, \ldots, U_{2m^2-m}$ which induces a $2m^2 - m$ coloring and we show that there is a solution of the matching problem. We split the proof into three claims. The first claim is to distribute a part of the vertices into three groups of independent sets $\{U_1, \ldots, U_m\}$, $\{U_{m+1}, \ldots, U_{m^2}\}$ and $\{U_{m^2+1}, \ldots, U_{2m^2-m}\}$. Using this distribution, we show that m vertices of A, B and C must lie in the first group and that the remaining $m^2 - m$ vertices of B and C must lie in the second and third group, respectively. The second claim shows that several groups of vertices must lie together in the same independent sets. Using this result we can show that each independent set $U_1, \ldots, U_m$ of the first group contains a triple of the form $a_i, b_{i,j}, c_{j,k}$ which corresponds directly to one set $A = \{w_i, x_j, y_k\}$ of the matching problem. The third claim proves that each element of W, X and Y is contained in these sets. Then, the equivalence between a solution of the matching problem and the coloring problem is shown.

Claim 2.5. *We can assume that the independent sets $U_1, \ldots, U_{2m^2-m}$ have the following form:*

(1) $\{a_i, p_i, d_{\beta(i)}\} \subset U_i$ *for each* $1 \le i \le m$, *where* $\beta : \{1, \ldots, m\} \to \{1, \ldots, m\}$ *is a permutation.*

(2) $E \cup G \cup Q \cup R \subset \bigcup_{i=m+1}^{m^2} U_i$.

(3) $F \cup H \cup Q' \cup R' \subset \bigcup_{i=m^2+1}^{2m^2-m} U_i$.

Since F covers the third group of independent sets and G covers the second, and since the sets $F \cup B$ and $C \cup G$ are cliques, we get for the vertices $b_{i,j} \in B$ and $c_{j,k} \in C$

- $B \subset \bigcup_{i=1}^{m^2} U_i$,
- $C \subset \bigcup_{i=1}^{m} U_i \cup \bigcup_{i=m^2+1}^{2m^2-m} U_i$.

If we delete the vertices of the independent sets $U_{m+1}, \dots, U_{2m^2-m}$, we have exactly m vertices of each of the sets A, B, C, D and P. We now study 'cuts' between two sets of vertices in the graph. Consider the following three cuts:

$$\begin{array}{lll} (1) & A \cup E & \text{and} \quad B \\ (2) & B \cup F & \text{and} \quad C \cup G \\ (3) & C & \text{and} \quad D \cup H. \end{array}$$

The possibilities for all three cuts are described in the next claim.

Claim 2.6. *The following three assertions are true:*

(1) *Fix i, and let U be an independent set with $b_{i,j} \in U \cap B$. Then, there is a vertex $a_i \in A$ with $a_i \in U$ or there is a vertex $e_{i,\ell} \in E$, $1 \leq \ell \leq m-1$ with $e_{i,\ell} \in U$.*

(2) *Fix j, and let U be an independent set with $b_{i,j} \in U \cap B$ or with $f_{i,j} \in U \cap F$. Then, there is a vertex $c_{j,k} \in C$, $1 \leq k \leq m$ with $c_{j,k} \in U$ or there is a vertex $g_{j,\ell} \in G$, $1 \leq \ell \leq m-1$ with $g_{j,\ell} \in U$.*

(3) *Fix k, and let U be an independent set with $c_{j,k} \in U \cap C$. Then, there is a vertex $d_k \in D$ with $d_k \in U$ or there is a vertex $h_{k,\ell} \in H$ with $h_{k,\ell} \in U$.*

From this analysis of cuts, we may now assume that the independent sets contain pairs of intervals, as illustrated in the following table. Take, for example, there is no independent set which contains $\{a_i, b_{i',\ell}\}$ or $\{e_{i,\ell}, b_{i',\ell'}\}$ for $i \neq i'$.

first vertex	second vertex	
a_i or $e_{i,-}$	$b_{i,-}$	$1 \leq i \leq m$
$b_{-,j}$ or $f_{j,-}$	$c_{j,-}$ or $g_{j,-}$	$1 \leq j \leq m$
$c_{-,k}$	d_k or $h_{k,-}$	$1 \leq k \leq m$

Now consider the graph after deleting the independent sets $U_{m+1}, \dots, U_{2m^2-m}$. We now have exactly m vertices of each of the sets A, B, C, D and P and m independent sets U_i with $\{a_i, p_i, d_{\beta(i)}\} \subset U_i$, $1 \leq i \leq m$ where β is a permutation.

Claim 2.7. *The first m independent sets have the form:*

$$U_i = \{a_i, p_i, b_{i,\alpha(i)}, c_{\alpha(i),\beta(i)}, d_{\beta(i)}\},$$

where $\alpha, \beta : \{1, \dots, m\} \to \{1, \dots, m\}$ are permutations.

Now we can prove that there is a partition of $W \cup X \cup Y$ into sets A_i with exactly one element of W, X and Y and with $\sum_{a \in A_i} a = Z$ iff the constructed one branching graph has a $2m^2 - m$ coloring.

Let $U_1, \dots, U_{2m^2-m}$ be such a coloring. Then we can assume that the second and third group of independent sets contain the vertices $E \cup F \cup G \cup H \cup Q \cup Q' \cup$

$R \cup R'$. Then, using the analysis above, the independent sets U_i, $1 \leq i \leq m$ have the form

$$U_i = \{a_i, b_{i,\alpha(i)}, c_{\alpha(i),\beta(i)}, d_{\beta(i)}, p_i\}$$

where α, β are permutations of $\{1, \ldots, m\}$. Using the fact that each set U_i is an independent set, we have $w_i + x_{\alpha(i)} + y_{\beta(i)} \leq Z$. Since the $\alpha(i)$ and $\beta(i)$ are permutations of $\{1, \ldots, m\}$, we get

$$\sum_{i=1}^{m} w_i + x_{\alpha(i)} + y_{\beta(i)} = \sum_{i=1}^{m} w_i + x_i + y_i.$$

Applying the equality $\sum_{i=1}^{m} w_i + \sum_{j=1}^{m} x_j + \sum_{k=1}^{m} y_k = mZ$, we get for each $1 \leq i \leq m$,

$$w_i + x_{\alpha(i)} + y_{\beta(i)} = Z.$$

Again, because the mappings $\alpha(i), \beta(i)$ are permutations, we can use these sets $A_i = \{w_i, x_{\alpha(i)}, y_{\beta(i)}\}$ as a solution of the numerical matching problem.

Conversely, w.l.o.g. we may assume that the sets $A_i = \{w_i, x_i, y_i\}$, $1 \leq i \leq m$ form a solution of the numerical matching problem. For the first m sets, we choose

$$U_i = \{a_i, b_{i,i}, c_{i,i}, d_i, p_i\}.$$

The interval a_i lies on the left side to $b_{i,i}$. To prove that $b_{i,i}$ lies on the left side to $c_{i,i}$ we compare the right endpoint of $b_{i,i}$ with the left endpoint of $c_{i,i}$. Using the fact that $w_i + x_i + y_i = Z$, we get $w_i + x_i + (iZ) < (i+1)Z - y_i + 1$. Therefore the sets $U_1, \ldots, U_m$ are independent.

Then, the vertex sets $B' = \{b_{i,j} | 1 \leq i \neq j \leq m\}$ and $C' = \{c_{i,j} | 1 \leq i \neq j \leq m\}$ are not covered. Construct for each $b_{i,j} \in B'$ an independent set. To do this, we take vertices $e_{i,\ell}, g_{j,\ell'}$, q_k, $r_{k'}$ which are not covered and put them together in one set U. Clearly, this set is independent. The construction is correct, because each index i and j appears only $(m-1)$ times in B'. Similarly, we construct independent sets for the set C'. For each $c_{i,j} \in C'$ we take vertices $f_{i,\ell}, h_{j,\ell'}$, q'_k, $r'_{k'}$ which are not covered. After these steps all vertices are covered and we have $2m^2 - m$ independent sets. This completes the proof of the theorem. □

3. Incompatibility graphs

3.1 The problem definition

A flow graph is an acyclic digraph $D = (V, E)$ with vertex set V and edge set E. The vertex set V is a union of a set of operations Op and a set of branching vertices $F \cup J$ where F is a set of fork vertices and where J is a set of join vertices. For the control, each fork vertex $f \in F$ is assigned a boolean function b_f over a set of control variables S. The functions describe the interdependence between the branches. There exist different hardware-design systems which use the information of the conflict graph corresponding to a branching-free flow graph. In this case as a conflict graph [7], [10] we get an interval graph.

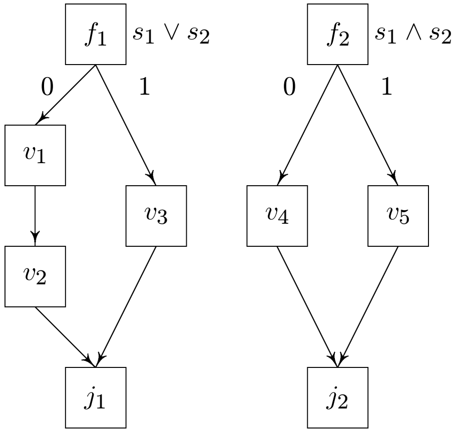

Figure 2: A flow graph with two branches

The simplest flow graph is a digraph with only one operation vertex. Given single operation flow graphs the class of all branching flow graphs can be generated recursively by the following two operations. The first operation connects two disjoint flow graphs $D_i = (V_i, E_i)$ for $i = 0, 1$ with any subset of edges from

$$\{(v, v')|d_{out}(v) = d_{in}(v') = 0, v \in V_0, v' \in V_1\},$$

where $d_{out}(v)$ ($d_{in}(v)$) denotes the out-degree (in-degree) of the vertex v. Using this operation two flow graphs can be connected or identified as independent parts.

The second operation constructs a branch, which is identified by two unique vertices, a fork vertex $f \in F$ and a join vertex $j \in J$. Given two disjoint flow graphs $D_i = (V_i, E_i)$ for $i = 0, 1$ and two new branching vertices f and j, a new flow graph $D = (V, E)$ is generated. The vertex set V is equal to $V_0 \cup V_1 \cup \{f, j\}$ and the edge set E is given by

$$E_0 \cup E_1 \cup \{(f, v)|v \in V_i, d_{in}(v) = 0\} \cup \{(v, j)|v \in V_i, d_{out}(v) = 0\}.$$

To specify two different branching parts, we give the directed edges $e = (f, v) \in E$ a weight $w_e = i$ for $v \in V_i$. Using this operation, the fork vertex f has in-degree $d_{in}(f) = 0$, the join vertex j has out-degree $d_{out}(j) = 0$ and all other vertices lie on a directed path from f to j.

Using a pair of branching vertices f and j, we can divide the flow graph into two different parts. Let $V(f, j)$ be the operations lying on a directed path from f to j, and let $V_i(f, j)$ for $i = 0, 1$ be the operations in $V(f, j)$ which can be reached over an i-weighted directed edge $(f, v) \in E$. Depending on an assignment of the control variables only one of the sets $V_0(f,j)$ or $V_1(f,j)$ can be executed. For a control function $\psi : S \to \{0,1\}$, the set of executed operations for ψ is defined by

$$Op_\psi = Op \setminus \bigcup_{f \in F} V(f,j)_{1-b_f(\psi(s_1),\ldots,\psi(s_{|S|}))}.$$

A set of operations Op_ψ is also called an execution trace. An example of a flow graph with operation set $Op = \{v_1,\ldots,v_5\}$, two branches and corresponding boolean functions $s_1 \vee s_2$, $s_1 \wedge s_2$ is given in Figure 2. The execution traces for the flow graph are $\{v_1,v_2,v_4\}$, $\{v_3,v_4\}$ and $\{v_3,v_5\}$; the set $\{v_1,v_2,v_5\}$ is not possible.

In a schedule for the flow graph, each operation $v \in Op$ is assigned an interval I_v on the positive real line such that for each control function ψ and each pair of operations $v,v' \in Op_\psi$, $v \neq v'$ with directed path from v to v' in the digraph and $x \in I_v$, $y \in I_{v'}$, we have $x < y$. For simplification we assume that the endpoints of the intervals are positive integers. A feasible schedule for our example is given by $I_{v_1} = I_{v_5} = [1,1]$, $I_{v_2} = [2,2]$ and $I_{v_3} = I_{v_4} = [1,2]$.

We denote $Op_z = \{v \in Op | z \in I_v\}$ as the operations at timestep $z \in Z_m = \{1,\ldots,m\}$. For each timestep z and each control function ψ the set of executed operations for ψ at timestep z is denoted by $Op_{\psi,z} = Op_\psi \cap Op_z$.

For each scheduled flow graph a conflict graph $IC = (Op,E)$ is defined with an edge $e = (v,v') \in E$ between two different operations $v,v' \in Op$, if and only if there is at least one trace Op_ψ with $\{v,v'\} \subset Op_\psi$ and $z \in I_v \cap I_{v'}$. In other words, if there is at least one execution trace Op_ψ which executes two operations v,v' at a time step $z \in I_v \cap I_{v'}$, then these operations cannot be assigned to the same processor. The graph IC_z, the so called local conflict graph for the time step $z \in Z_m$, is the subgraph of IC induced by the set Op_z. The conflict graph for our example is given in Figure 3.

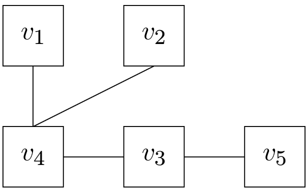

Figure 3: The conflict graph to the flow graph in Figure 2

Our goal is to determine a minimum number of processors for a scheduled flow graph. This corresponds to a partition of the operation set Op in compatible sets $U_1,\ldots,U_k$ with minimum $k \in \mathbb{N}$ which in turn is equivalent to the problem of finding a minimum coloring of the conflict graph IC. We note that for general boolean functions b_f the problem whether two operations are compatible is NP-hard. We can simulate the SAT-problem in the boolean functions such that there is an execution trace ψ with $\{v, v'\} \in Op_\psi$ iff the SAT problem is satisfiable. However, we consider here the case that the boolean functions are not too complicated (e.g. $b_f(s_1, \ldots, s_{|S|}) \in S$.) Then, the conflict graph can be computed in polynomial time.

3.2 Structure of conflict graphs

The difficulty of the optimization problems lies in the conflict graphs. Since in the general case each undirected graph can be a conflict graph [5], and since the coloring problem is known to be NP-complete [3], the processor assignment problem is also NP-complete. In the following we analyze the structure of the conflict graphs for scheduled flow graphs with a constant number of execution traces.

Theorem 3.1. *Let D be a flow graph with operations $v \in Op$ where the number of execution traces is bounded by a constant ℓ and let I_v, $v \in Op$ be intervals in a feasible schedule. Then, the conflict graph $IC = (Op, E)$ is the intersection of an interval graph G_1 and a graph G_2 which is a union of ℓ complete graphs.*

Proof: This result follows directly from the definition of the conflict graph. We have an edge between two operations v, v' with $v \neq v'$ if and only if two conditions are satisfied. The first condition $I_v \cap I_{v'} \neq \emptyset$ corresponds to the interval graph G_1 and the second to a union of a constant number of complete graphs; one complete graph for each execution trace Op_ψ. Since we only have an edge if both conditions are satisfied, the conflict graph is the intersection of both graphs. □

From this classification, it follows that each independent set in the local conflict graph IC_z has size at most ℓ. In other words, the independence number $\alpha(IC_z)$ is bounded by the constant ℓ. In the following we analyze the case that each operation is assigned an interval I_v which consists of one execution step. In other words, $I_v \in \{1, \ldots, m\}$ for each operation $v \in Op$. In this case, we say that the operations have unit times.

Theorem 3.2. *Let D be a flow graph with unit-times $I_v \in \{1, \ldots, m\}$, $v \in Op$ and with ℓ execution traces. Then, the conflict graph IC is the intersection of graphs G_1 and G_2 where G_1 is a disjoint union of m complete graphs and where G_2 is a non-disjoint union of ℓ complete graphs.*

Proof: This follows, because the interval graph G_1 is a disjoint union of m complete graphs; one complete graph for each execution step $i \in \{1, \ldots, m\}$. □

In many applications we search for an assignment of the operations to a small constant number of processors. This corresponds to the case that the conflict graph can be colored with a constant number of colors.

Theorem 3.3. *Let D be a flow graph with a constant number ℓ of execution traces. If the chromatic number of the conflict graph IC is also bounded by a constant, then IC has constant pathwidth.*

Proof: First, the graph IC is an intersection of an interval graph G_1 and a graph G_2 which is a union of ℓ complete graphs. Let us consider the local conflict graphs IC_z and let us take the path decomposition corresponding to the time steps $1, \ldots, m$. Each local conflict graph is a union of at most ℓ complete graphs. Since the number of vertices in each graph $G = (V, E)$ is less or equal than the product of the chromatic number $\chi(G)$ and the independence number $\alpha(G)$, we get $|Op_z| \leq \alpha(IC_z) \times \chi(IC_z)$. Since $\alpha(IC_z) \leq \ell$ and since $\chi(IC_z) \leq \chi(IC)$, the number of operations in each local conflict graph is bounded by $\ell \times \chi(IC)$. This implies that the pathwidth of the conflict graph is constant. $\square$

3.3 Complexity results

First, we analyze the complexity of the problem of finding a maximum compatible set of operations. Clearly, a maximum compatible set is a maximum independent set in the conflict graph. As consequence of our graph theoretical results, we get directly:

Theorem 3.4. *Given a scheduled flow graph with a constant number of execution traces and the corresponding conflict graph IC, the problem of finding a maximum compatible set of operations is solvable in polynomial time. If all operations have unit-times or if $\chi(IC)$ is bounded by a constant the problem is solvable in linear time.* $\square$

We now analyze the complexity of the problem of finding an assignment of the operations in a flow graph to a minimum number of processors. The first result is quite disappointing, because it does not allow a generalization of results for branch-free graphs to flow graphs with one, two or more branches, unless $P = NP$. The result follows directly from Theorem 2.4.

Theorem 3.5. *Given a scheduled flow graph with two execution traces and the corresponding conflict graph, the problem of finding an assignment of the operations to a minimum number of processors is NP-complete.*

$\square$

As consequence of our polynomial results, we get:

Theorem 3.6. *Given a scheduled flow graph with a constant number of execution traces and the corresponding conflict graph IC, the problem of finding an assignment of unit-time operations to a minimum number of processors is solvable in polynomial time. If $\chi(IC)$ is bounded by a constant, the problem with non unit-time operations is solvable in linear time.* $\square$

References

[1] Arnborg S., Proskurowski A., *Linear time algorithms for NP-hard problems on graphs embedded in k-trees*, TRITA-NA-8404, Dept. of Num. Anal. and Comp. Sci, Royal Institute of Technology, Stockholm, Sweden, 1984.

[2] Bodlaender H.L., *A linear time algorithm for finding tree-decompositions of small treewidth*, Report RUU-CS-92-27, Dept. of Computer Sci., Utrecht University, 1992.

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

[4] Jansen K., *Processor optimization for flow graphs*, Theor. Comput. Sci. **104** (1992), 285–298.

[5] ———, *The allocation problem in hardware design*, Disc. Appl. Math. **43** (1993), 37–46.

[6] Lawler E.L., *Combinatorial Optimization: Networks and Matroids*, Rinehard and Winston, New York, 1976.

[7] Pfahler P., *Übersetzermethoden zur automatischen Hardware-Synthese*, Thesis, University of Paderborn, FRG, 1988.

[8] Rajan J.V., *Automatic synthesis of data paths in digital systems*, PhD thesis, Carnegie Mellon University, Dec 1988.

[9] Robertson N., Seymour P., *Graph minors. I. Excluding a forest*, J. Comb. Theory **B 35** (1983), 39–61.

[10] Springer D.L., Thomas D.E., *Exploiting the special structure of conflict and compatibility graphs in high-level synthesis*, ICCAD (1990), 254–257.

[11] Tseng C.J., Siewiorek D.P., *Automated synthesis of data paths in digital systems*, IEEE Trans. Comp.-Aided Design **5** (1986), 379–395.

Fachbereich 11 – Mathematik, FG Informatik, Universität Duisburg, 47 048 Duisburg, Germany

(*Received* May 7, 1993, *revised* November 5, 1993)