Decidable Boundedness Problems for Hyperedge–Replacement Graph Grammars

Annegret Habel *, Hans-Jörg Kreowski *, Walter Vogler **

Abstract

Consider a class $\mathcal{C}$ of hyperedge–replacement graph grammars and a numeric function on graphs like the number of edges, the degree (i.e., the maximum of the degrees of all nodes of a graph), the number of simple paths, the size of a maximum set of independent nodes, etc. Each such function induces a Boundedness Problem for the class $\mathcal{C}$: Given a grammar HRG in $\mathcal{C}$, are the function values of all graphs in the language $L(HRG)$, generated by HRG, bounded by an integer or not? We show that the Boundedness Problem is decidable if the corresponding function is compatible with the derivation process of the grammars in $\mathcal{C}$ and if it is composed of maxima, sums, and products in a certain way. This decidability result applies particularly to the examples listed above.

1. Introduction

Context–free graph grammars (like edge– and hyperedge–replacement grammars as investigated, e.g., by Bauderon and Courcelle [BD 87] or in [HK 85+87b] or like boundary NLC grammars as introduced by Rozenberg and Welzl [RW 86a]) have been studied intensively for some time now because of – at least – two reasons:
(1) Although their generative power is intentionally restricted, they cover many graph languages interesting from the point of view of applications as well as of graph theory (for example, certain types of flow diagrams, PASCAL syntax diagrams, certain types of Petri nets, graph representations of functional expressions, series–parallel graphs, outerplanar graphs, k–trees, graphs with cyclic bandwidth $\leq k$).
(2) Of all classes of graph grammars discussed in the literature, they seem to render the most attractive theory with a variety of results on structure, decidability and complexity (see, e.g., Arnborg, Lagergren and Seese [ALS 88], Bauderon and Courcelle [BC 87, Co 87], Della Vigna and Ghezzi [DG 78], Lautemann [La 88], Lengauer and Wanke [LW 88], Rozenberg and Welzl [86a+b], Slisenko [Sl 82], and [HK 83 + 85 + 87b, HKV 87, Kr 79]).

In particular, Courcelle [Co 87], Arnborg et al. [ALS 88], Lengauer and Wanke [LW 88], and [HKV 87] present syntactic and semantic conditions such that, for a graph property P satisfying the conditions, the following hold for all context–free graph grammars of the types considered in the respective papers:
(1) It is decidable whether (or not) some graph with property P is generated.
(2) It is decidable whether (or not) all generated graphs have property P.
(3) It is decidable in linear time whether (or not) a generated graph represented by a derivation (or something equivalent) has property P.
The results apply to properties such as connectivity, planarity, k–colorability, existence of Hamiltonian and Eulerian paths and cycles.

* Universität Bremen, Fachbereich Mathematik und Informatik, Postfach 330440, D-2800 Bremen 33
** Technische Universität München, Institut für Informatik, Postfach 202420, D-8000 München 2

Based on the framework of hyperedge–replacement graph grammars, we continue this line of consideration in this paper. We are going to investigate the decidability of a different type of problems concerning functions on graphs and above all numeric quantities like the numbers of nodes, edges and paths, the node degree, maximum and minimum lengths of paths and cycles, etc. The kind of question we ask for a class of grammars may be called *Boundedness Problem*. It is as follows:

(4) Is it decidable whether (or not), concering a particular quantity, the values of all graphs generated by a grammar are bounded?

For example, we want to know whether the node degree or the number of paths grow beyond any bound within a graph language. In the main result, we show that such a Boundedness Problem is decidable for a class of hyperedge–replacement grammars if the corresponding quantity function is built up by maxima, sums and products and if the function is compatible with the derivation process of the given grammars. Examples of this kind are the bounded–node–degree problem, the bounded–maximum–path–length problem, the bounded–maximum–number–of–paths problem and others. It should be mentioned here that the only result of the same nature occurring in the literature is the decidability of the bounded–degree problem for NLC grammars (see [JRW 86]).

The paper is organized in the following way. Sections 2 and 3 comprise the preliminaries on (hyper)graphs and hyperedge–replacement grammars as needed. In Section 4, we discuss several examples of numeric functions which are compatible with the derivation process of our grammars in a certain way. In Section 5, we introduce the general notion of compatible functions, and we relate them with our earlier notion of compatible predicates [HKV 87]. Finally, we show in the main result in Section 6 that the Boundedness Problem corresponding to a numeric function is decidable if the function is pointwise defined as the maximum of sums and products and if it is compatible.

Except for the proof of the main theorem in Section 6, we omit proofs in this version. They can be found in the long version of the paper which will appear elsewhere. While the general results work for arbitrary classes of hyperedge–replacement grammars, we have to admit that most of our examples are formulated for the class of edge–replacement grammars. But we are confident that all of them can be adapted to more general classes of hyperedge–replacement grammars.

2. Preliminaries

This section provides the basic notions on graphs and hypergraphs as far as needed in the paper. The key construction is the replacement of some hyperedges of a hypergraph by hypergraphs yielding an expanded hypergraph. In our approach, a hyperedge is an atomic item with an ordered set of incoming tentacles and an ordered set of outgoing tentacles where each tentacle grips at a node through the source and target functions. Correspondingly, a hypergraph is equipped with two sequences of distinguished nodes so that it is enabled to replace a hyperedge.

2.1 Definition (hypergraphs)

1. Let C be an arbitrary, but fixed alphabet, called a set of *labels* (or *colors*).
2. A *hypergraph* over C is a system (V, E, s, t, l) where V is a finite set of *nodes* (or *vertices*), E is a finite set of *hyperedges*, $s : E \to V^*$ and $t : E \to V^*$ [1] are two mappings assigning a sequence of *sources* $s(e)$ and a sequence of *targets* $t(e)$ to each $e \in E$, and $l : E \to C$ is a mapping *labeling* each hyperedge.
3. A hyperedge $e \in E$ of a hypergraph (V, E, s, t, l) is called an (m, n)*–edge* for some $m, n \in \mathbb{N}$ if $|s(e)| = m$ and $|t(e)| = n$. [2] The pair (m, n) is the *type* of e, denoted by $type(e)$. e is said to be *well–formed* if its sources and targets are pairwise distinct.

[1] For a set A, A^* denotes the set of all words over A, including the empty word λ.

[2] For a word $w \in A^*$, $|w|$ denotes its length.

4. A *multi-pointed hypergraph* over C is a system $H = (V, E, s, t, l, begin, end)$ where the first five components define a hypergraph over C and $begin, end \in V^*$. Components of H are denoted by $V_H, E_H, s_H, t_H, l_H, begin_H, end_H$, respectively. The set of all multi-pointed hypergraphs over C is denoted by $\mathcal{H}_C$.
5. $H \in \mathcal{H}_C$ is said to be an (m,n)-*hypergraph* for some $m, n \in \mathbb{N}$ if $|begin_H| = m$ and $|end_H| = n$. The pair (m,n) is the *type* of H, denoted by $type(H)$. H is said to be *well-formed* if all hyperedges are well-formed and the begin-nodes and end-nodes of H are pairwise distinct.
6. Let $H \in \mathcal{H}_C$, $begin_H = begin_1 \dots begin_m$ and $end_H = end_1 \dots end_n$ with $begin_i, end_j \in V_H$ for $i = 1, \dots, m$ and $j = 1, \dots, n$. Then $EXT_H = \{begin_i | i = 1, \dots, m\} \cup \{end_j | j = 1, \dots, n\}$ denotes the set of *external nodes* of H. Moreover, $INT_H = V_H - EXT_H$ denotes the set of *internal* nodes of H.

Remarks: 1. There is a 1–1–correspondence between hypergraphs and (0,0)–hypergraphs so that hypergraphs may be seen as special cases of multi-pointed hypergraphs.
2. An (m,n)–hypergraph over C with (1,1)–edges only is said to be an (m,n)-*graph*. The set of all (1,1)–graphs over C is denoted by $\mathcal{G}_C$.

2.2 Definition (special hypergraphs)

1. A multi-pointed hypergraph H is said to be a *singleton* if $|E_H| = 1$ and $|V_H - EXT_H| = 0$. $e(H)$ refers to the only hyperedge of H and $l(H)$ refers to its label.
2. A singleton H is said to be a *handle* if $s_H(e) = begin_H$ and $t_H(e) = end_H$. If $l_H(e) = A$ and $type(e) = (m,n)$ for some $m, n \in \mathbb{N}$, then H is called an (m,n)-*handle* induced by A.

Remark: Given $H \in \mathcal{H}_C$, each hyperedge $e \in E_H$ induces a handle $e^\bullet$ by restricting the mappings s_H, t_H, and l_H to the set $\{e\}$, restricting the set of nodes to those ones occurring in $s_H(e)$ and $t_H(e)$, and choosing $begin_{e^\bullet} = s_H(e)$ and $end_{e^\bullet} = t_H(e)$.

2.3 Definition (subhypergraphs and isomorphic hypergraphs)

1. Let $H, H' \in \mathcal{H}_C$. Then H is called a *(weak) subhypergraph* of H', denoted by $H \subseteq H'$, if $V_H \subseteq V_{H'}$, $E_H \subseteq E_{H'}$, and $s_H(e) = s_{H'}(e)$, $t_H(e) = t_{H'}(e)$, $l_H(e) = l_{H'}(e)$ for all $e \in E_H$.
[Note that nothing is assumed on the relation of the distinguished nodes.]
2. Let $H, H' \in \mathcal{H}_C$ and $i_V : V_H \to V_{H'}$, $i_E : E_H \to E_{H'}$ be bijective mappings. Then $i = (i_V, i_E) : H \to H'$ is called an *isomorphism* from H to H' if $i_V^*(s_H(e)) = s_{H'}(i_E(e))$, $i_V^*(t_H(e)) = t_{H'}(i_E(e))$, $l_H(e)) = l_{H'}(i_E(e))$ for all $e \in E_H$ as well as $i_V^*(begin_H) = begin_{H'}$, $i_V^*(end_H) = end_{H'}$ [3]. H and H' are said to be *isomorphic*, denoted by $H \cong H'$, if there is an isomorphism from H to H'.

Now we are ready to introduce how hypergraphs may substitute hyperedges. An (m,n)–edge can be replaced by an (m,n)–hypergraph in two steps:
(1) Remove the hyperedge,
(2) add the hypergraph except the external nodes and hand over each tentacle of a hyperedge (of the replacing hypergraph) which grips to an external node to the corresponding source or target node of the replaced hyperedge.
Moreover, an arbitrary number of hyperedges can be replaced simultaneously in this way.

2.4 Definition (hyperedge replacement)

Let $H \in \mathcal{H}_C$ be a multi-pointed hypergraph, $B \subseteq E_H$, and $repl : B \to \mathcal{H}_C$ a mapping with $type(repl(b)) = type(b)$ for all $b \in B$. Then the *replacement* of B in H through $repl$ yields the multi-pointed hypergraph X given by

[3] For a mapping $f : A \to B$, the free symbolwise extension $f^* : A^* \to B^*$ is defined by $f^*(a_1 \dots a_k) = f(a_1) \dots f(a_k)$ for all $k \in \mathbb{N}$ and $a_i \in A$ $(i = 1, \dots, k)$.

- $V_X = V_H + \sum_{b \in B}(V_{repl(b)} - EXT_{repl(b)})$, [4]
- $E_X = (E_H - B) + \sum_{b \in B}(E_{repl(b)})$,
- each hyperedge of $E_H - B$ keeps its sources and targets,
- each hyperedge of $E_{repl(b)}$ (for all $b \in B$) keeps its internal sources and targets and the external ones are handed over to the corresponding sources and targets of b, i.e.,
  $s_X(e) = h^*(s_{repl(b)}(e))$ and $t_X(e) = h^*(t_{repl(b)}(e))$ for all $b \in B$ and $e \in E_{repl(b)}$
  where $h : V_{repl(b)} \to V_X$ is defined by $h(v) = v$ for $v \in V_{repl(b)} - EXT_{repl(b)}$,
  $h(b_i) = s_i$ $(i = 1, \ldots, m)$ for $begin_{repl(b)} = b_1 \ldots b_m$ and $s_H(b) = s_1 \ldots s_m$,
  $h(e_j) = t_j$ $(j = 1, \ldots, n)$ for $end_{repl(b)} = e_1 \ldots e_n$ and $t_H(b) = t_1 \ldots t_n$.
- each hyperedge keeps its label,
- $begin_X = begin_H$ and $end_X = end_H$.

The resulting multi–pointed hypergraph X is denoted by $REPLACE(H, repl)$.

Remark: The construction above is meaningful and determines (up to isomorphism) a unique hypergraph X if h is a mapping. This is automatically fulfilled whenever the *begin*–nodes and *end*–nodes of each replacing hypergraph are pairwise distinct. If one wants to avoid such a restriction, one has to require that the following application condition is satisfied for each $b \in B$: If $begin_{repl(b)} = x_1 \ldots x_m$ and $end_{repl(b)} = x_{m+1} \ldots x_{m+n}$ as well as $s_H(b) = y_1 \ldots y_m$ and $t_H(b) = y_{m+1} \ldots y_{m+n}$, then, for $i, j = 1, \ldots, m+n$, $x_i = x_j$ implies $y_i = y_j$.

3. Hyperedge–Replacement Grammars and Languages

In this section we give a short summary of the basic notions on hyperedge–replacement grammars generalizing edge–replacement grammars as investigated e.g. in [HK 83+85] and context–free string grammars. Details and examples can be found in [HK 87a+b].

Based on hyperedge replacement, one can derive multi–pointed hypergraphs from multi–pointed hypergraphs by applying productions of a simple form.

3.1 Definition (productions and derivations)

1. Let $N \subseteq C$. A *production* (over N) is an ordered pair $p = (A, R)$ with $A \in N$ and $R \in \mathcal{H}_C$. A is called *left–hand side* of p and is denoted by $lhs(p)$, R is called *right–hand side* and is denoted by $rhs(p)$. The *type* of p, denoted by $type(p)$, is given by the type of R.
2. Let $H \in \mathcal{H}_C$, $B \subseteq E_H$, and P be a set of productions. A mapping $prod : B \to P$ is called a *production base* in H if $l_H(b) = lhs(prod(b))$ and $type(b) = type(rhs(prod(b)))$ for all $b \in B$.
3. Let $H, H' \in \mathcal{H}_C$ and $prod : B \to P$ be a production base in H. Then H *directly derives* H' *through* $prod$ if H' is isomorphic to $REPLACE(H, repl)$ where $repl : B \to \mathcal{H}_C$ is given by $repl(b) = rhs(prod(b))$ for all $b \in B$. We write $H \underset{P}{\Longrightarrow} H'$ or $H \Longrightarrow H'$ in this case.
4. A sequence of direct derivations $H_0 \Longrightarrow H_1 \Longrightarrow \ldots \Longrightarrow H_k$ is called a *derivation* from H_0 to H_k (of length k). Additionally, in the case $H \cong H'$, we speak of a *derivation* from H to H' of length 0. A derivation from H to H' is shortly denoted by $H \overset{*}{\underset{P}{\Longrightarrow}} H'$ or $H \overset{*}{\Longrightarrow} H'$. If the length of the derivation should be stressed, we write $H \overset{k}{\underset{P}{\Longrightarrow}} H'$ or $H \overset{k}{\Longrightarrow} H'$.
5. A direct derivation through $prod : \emptyset \to P$ is called a *dummy*. [5] A derivation is said to be *valid* if at least one of its steps is not a dummy.

---

[4] The sum symbols $+$ and $\sum$ denote the disjoint union of sets; the symbol $-$ denotes the set–theoretic difference.

[5] A production base $prod : B \to P$ in H may be *empty*, i.e., $B = \emptyset$. In this case $H \Longrightarrow H'$ through $prod$ implies $H \cong H'$, and there is always a trivial direct derivation $H \Longrightarrow H$ through $prod$.

Remarks: 1. The application of a production $p = (A, R)$ of type (m, n) to a multi-pointed hypergraph H requires the following two steps only:
(1) Choose a hyperedge e of type (m, n) with label A.
(2) Replace the hyperedge e in H by R.
2. Some significant properties of direct derivations are: On the one hand, the definition of a direct derivation includes the case that no hyperedge is replaced. This dummy step derives a hypergraph isomorphic to the initial one. On the other hand, it includes the case that all hyperedges are replaced in one step. Moreover, whenever some hyperedges can be replaced in parallel, they can be replaced one after the other leading to the same derived hypergraph.

Using the introduced concepts of productions and derivations hyperedge-replacement grammars and languages can be introduced in a straightforward way.

3.2 Definition (hyperedge-replacement grammars and languages)

1. A *hyperedge-replacement grammar* is a system $HRG = (N, T, P, Z)$ where $N \subseteq C$ is a set of *nonterminals*, $T \subseteq C$ is a set of *terminals*, P is a finite set of *productions* over N, and $Z \in \mathcal{H}_C$ is the *axiom*. The class of all hyperedge-replacement grammars is denoted by $\mathcal{HRG}$.
2. HRG is said to be *typed* if there is a mapping $\mathit{ltype} : N \cup T \to \mathbb{N} \times \mathbb{N}$ such that, for each production $(A, R) \in P$, $\mathit{ltype}(A) = \mathit{type}(R)$ and $\mathit{ltype}(l_R(e)) = \mathit{type}(e)$ for all $e \in E_R$ and $\mathit{ltype}(l_Z(e)) = \mathit{type}(e)$ for all $e \in E_Z$. HRG is said to be *well-formed* if the right-hand sides of the productions are well-formed and all hyperedges in Z are well-formed.
3. The *hypergraph language* $L(HRG)$ *generated by* HRG consists of all hypergraphs which can be derived from Z applying productions of P and which are terminally labeled:

$$L(HRG) = \{H \in \mathcal{H}_T \mid Z \overset{*}{\underset{P}{\Longrightarrow}} H\}.$$

Remarks: 1. Even if one wants to generate graph languages rather than hypergraph languages, one may use nonterminal hyperedges because the generative power of hyperedge-replacement grammars increases with the maximum number of tentacles of a hyperedge involved in the replacement (see [HK 87b]).
2. Without effecting the generative power, we will assume in the following that N and T are finite, $N \cap T = \emptyset$, and Z is a singleton with $l(Z) \in N$. Furthermore, we will assume that the hyperedge-replacement grammars considered in this paper are typed and well-formed.

The results presented in the following sections are mainly based on some fundamental aspects of hyperedge-replacement derivations. Roughly speaking, hyperedge-replacement derivations cannot interfere with each other as long as they handle different hyperedges. On the one hand, a collection of derivations of the form $e^\bullet \overset{*}{\Longrightarrow} H(e)$ for $e \in E_R$ can be simultaneously embedded into R leading to a single derivation $R \overset{*}{\Longrightarrow} H$. On the other hand, restricting a derivation $R \overset{*}{\Longrightarrow} H$ to the handle $e^\bullet$ induced by the hyperedge $e \in E_R$ one obtains a so-called "restricted" derivation $e^\bullet \overset{*}{\Longrightarrow} H(e)$ where $H(e) \subseteq H$. Finally, restricting a derivation to the handles induced by the hyperedges, and subsequently embedding them again returns the original derivation. In other words, hyperedge-replacement derivations can be distributed to the handles of the hyperedges without losing information. We state and use this result in the following recursive version concerning terminal hypergraphs which are derivable from handles.

3.3 Theorem

Let $HRG = (N, T, P, Z)$ be a typed and well-formed hyperedge-replacement grammar, $A \in N \cup T$, and $H \in \mathcal{H}_T$. Then there is a derivation $A^\bullet \Longrightarrow R \overset{k}{\Longrightarrow} H$ for some $k \geq 0$ [6] if and only if $A^\bullet \Longrightarrow R$ and, for each $e \in E_R$, there is a derivation $l_R(e)^\bullet \overset{k}{\Longrightarrow} H(e)$ with $H(e) \subseteq H$ such that $H \cong \mathit{REPLACE}(R, \mathit{repl})$ with $\mathit{repl}(e) = H(e)$ for $e \in E_R$.

[6] For a symbol $A \in N \cup T$ with $\mathit{ltype}(A) = (m, n)$, $A^\bullet$ denotes an (m, n)-handle induced by A. [Note that (m, n)-handles induced by a symbol A are isomorphic].

Remarks: 1. The derivation $l_R(e)^\bullet \stackrel{k}{\Longrightarrow} H(e)$ may be valid or not. In the first case, it has the same form as the original derivation, but it is shorter as the original one. In the latter case, $H(e)$ is isomorphic to $e^\bullet$ (resp. $l_R(e)^\bullet$) and hence a terminal handle.

2. Given a derivation $R \stackrel{k}{\Longrightarrow} H$, the derivation $l_R(e)^\bullet \stackrel{k}{\Longrightarrow} H(e)$ for each $e \in E_R$ is called the *fibre* of e and — the other way round — the given derivation is the *joint embedding* of its fibres.

4. Some Graph–Theoretic Functions Compatible With Derivations

A hyperedge–replacement grammar as a generating device specifies a (hyper)graph language. Unfortunately, in a finite amount of time, the generating process only produces a finite section of the language explicitly (and even this may consume much time). Hence one may wonder what the hyperedge–replacement grammar can tell us about the generated language. As a matter of fact, by Theorem 3.3, we have the following nice situation. Given a hyperedge–replacement grammar and an arbitrary terminal (hyper)graph H with derivation $A^\bullet \Longrightarrow R \stackrel{*}{\Longrightarrow} H$, we get a decomposition of H into "smaller" components which are derivable from the handles of the hyperedges in R. If one is interested in values of graph–theoretic functions of derived (hyper)graphs, one may ask how a certain value of a derived (hyper)graph depends on values of the components. A function is said to be "compatible" with the derivation process of hyperedge–replacement grammars if it can be computed for each derived (hyper)graph H by computing the values (or related values) for the components and composing the values to the value of H.

In this section, we pick up several graph–theoretic functions and show that they are "compatible" with the replacement process of hyperedges. A formal definition of compatibility is given in the next section. We discuss the number of nodes and hyperedges, the number of paths and cycles, the length of a shortest path, the length of a longest simple path,and the minimum and maximum degree.

Let $HRG = (N, T, P, Z)$ be a typed and well–formed hyperedge–replacement grammar, $H \in \mathcal{H}_T$, $A^\bullet \Longrightarrow R \stackrel{*}{\Longrightarrow} H$ a derivation of H in HRG, and, for $e \in E_R$, $l_R(e)^\bullet \stackrel{*}{\Longrightarrow} H(e)$ be the fibre of $R \stackrel{*}{\Longrightarrow} H$ induced by e. Then the number of nodes in H can be computed from the number of nodes in R and the number of internal nodes in the $H(e)$'s. Similarly, the number of internal nodes can be computed. Even simpler, the number of hyperedges in H can be determined by the number of hyperedges in the $H(e)$'s.

4.1 Theorem (Number of Nodes and Hyperedges)

For a hypergraph $H \in \mathcal{H}_C$, let $|V_H|$ denote the number of nodes, $|INT_H|$ the number of internal nodes, and $|E_H|$ the number of hyperedges in H. Then

$$|V_H| = |V_R| + \sum_{e \in E_R} |INT_{H(e)}|,$$

$$|INT_H| = |INT_R| + \sum_{e \in E_R} |INT_{H(e)}|,$$

$$|E_H| = \sum_{e \in E_R} |E_{H(e)}|.$$

Remarks: 1. Similarly, the composed function *size* given by $size(H) = |V_H| + |E_H|$ can be handled. It makes use of the auxiliary function *intsize* given by $intsize(H) = |INT_H| + |E_H|$.

2. The density function $dens$ given by $dens(H) = \frac{|E_H|}{|V_H|}$ if $|V_H| > 0$ (and $dens(H) = \diamond$ [7] otherwise) can also be expressed in such a way:

$$dens(H) = \frac{\sum_{e \in E_R} |E_{H(e)}|}{|V_R| + \sum_{e \in E_R} |INT_{H(e)}|}$$

The expression for computing $dens(H)$ makes use of the possibility to compute the number of internal nodes as well as the number of hyperedges of the $H(e)$'s. It does not make use of the density of some of the $H(e)$'s. □

For simplifying the technicalities, we restrict our following consideration to the class $\mathcal{ERG}$ of edge–replacement grammars in the sense of [HK 83+85]. To be more explicit, a typed and well–formed hyperedge–replacement grammar $HRG = (N, T, P, Z)$ is in $\mathcal{ERG}$ if and only if the right–hand sides of the productions as well as the axiom are (1,1)–graphs. Note that, in this case, each $G \in L(HRG)$ is a (1,1)–graph, i.e., a graph with two distinguished nodes $begin_G$ and end_G.

Let G be a graph. A *path* joining v_0 and v_n is a sequence $p = v_0, e_1, v_1, e_2, \ldots, e_n, v_n$ of alternating nodes and edges such that for $1 \leq i \leq n$, v_{i-1} and v_i are the nodes incident with e_i. If $v_0 = v_n$ then p is said to be a *cycle*. If in a path each node appears once, then the sequence is called a *simple* path. If each node appears once except that $v_0 = v_n$ and $n \geq 3$ then p is a *simple cycle*. The *length* of a path or a cycle p, denoted $length(p)$, is the number of edges it contains. "e *on* p" denotes the fact that e occurs in p.

4.2 Theorem (Number of Simple Paths, Minimum and Maximum Simple–Path Length)

For a (1,1)–graph G, let $PATH_G$ denote the set of simple paths joining $begin_G$ and end_G and $numpath(G)$ the number of these paths in G. Moreover, let $minpath(G)$ and $maxpath(G)$ denote the minimum resp. maximum simple–path length, if any (and $minpath(G) = maxpath(G) = \diamond$ otherwise [8]). Then

$$numpath(G) = \sum_{p \in PATH_R} \prod_{e \text{ on } p} numpath(G(e)),$$

$$minpath(G) = \min_{p \in PATH_R} \sum_{e \text{ on } p} minpath(G(e)),$$

$$maxpath(G) = \max_{p \in PATH_R} \sum_{e \text{ on } p} maxpath(G(e)).$$

The number of simple cycles, the minimum cycle length, and the maximum simple–cycle length of a graph can be determined using the computation of the number of simple paths, the minimum path length, and the maximum simple–path length, respectively.

---

[7] $dens$, $minpath$, and $maxpath$ are defined to be functions with values in $\mathbb{N} \cup \{\diamond\}$, the set of all non-negative integers plus a special symbol $\diamond$. We use this special symbol $\diamond$, if the considered function has no sensible integer value. We calculate with $\diamond$ as follows: $\forall\, i \in I\ \forall\, n_i \in \mathbb{N} \cup \{\diamond\}$,

- $\sum_{i \in I} n_i = \diamond$ and $\prod_{i \in I} n_i = \diamond$ if and only if $n_j = \diamond$ for some $j \in I$,
- $\min_{i \in I} n_i = \min_{i \in I'} n_i$ and $\max_{i \in I} n_i = \max_{i \in I'} n_i$ for $I' = \{i \in I | n_i \neq \diamond\}$, and $\min_{i \in I} n_i = \diamond$ and $\max_{i \in I} n_i = \diamond$ for $I = \emptyset$.

4.3 Theorem (Number of Simple Cycles, Minimum and Maximum Simple–Cycle Length)

For a (1,1)–graph G, let $CYCLE_G$ denote the set of simple cycles and $numcycle(G)$ the number of these cycles in G. Moreover, let $mincycle(G)$ and $maxcycle(G)$ denote the minimum resp. maximum simple–cycle length, if any; otherwise, let $mincycle(G) = \diamond = maxcycle(G)$. Then

$$numcycle(G) = \sum_{c \in CYCLE_R} \prod_{e \text{ on } c} numpath(G(e)) + \sum_{e \in E_R} numcycle(G(e)),$$

$$mincycle(G) = \min \left\{ \min_{c \in CYCLE_R} \sum_{e \text{ on } c} minpath(G(e)) \ , \ \min_{e \in E_R} mincycle(G(e)) \right\},$$

$$maxcycle(G) = \max \left\{ \max_{c \in CYCLE_R} \sum_{e \text{ on } c} maxpath(G(e)) \ , \ \max_{e \in E_R} maxcycle(G(e)) \right\}.$$

4.4 Theorem (Minimum and Maximum Degree)

For a graph G, let $mindegree(G)$ and $maxdegree(G)$ denote the minimum resp. maximum degree among the nodes of G. Moreover, let $minintdegree(G)$ denote the minimum degree among the internal nodes of G and $bdegree(G)$ and $edegree(G)$ the degree of $begin_G$ resp. end_G. Then

$$mindegree(G) = \min \left\{ \min_{v \in V_R} D_G(v) \ , \ \min_{e \in E_R} minintdegree(G(e)) \right\}$$

$$minintdegree(G) = \min \left\{ \min_{v \in INT_R} D_G(v) \ , \ \min_{e \in E_R} minintdegree(G(e)) \right\}$$

$$maxdegree(G) = \max \left\{ \max_{v \in V_R} D_G(v) \ , \ \max_{e \in E_R} maxdegree(G(e)) \right\}$$

$$bdegree(G) = D_G(begin_G) \text{ and } edegree(G) = D_G(end_G)$$

where, for $v \in V_R$, $D_G(v) = \sum_{e \in s_R^{-1}(v)} bdegree(G(e)) + \sum_{e \in t_R^{-1}(v)} edegree(G(e))$.

5. Compatible Functions

In this section we introduce the notion of compatible functions in such a way that all functions considered in the previous section are special cases. Roughly speaking, a function f_0 on hypergraphs is said to be compatible with the derivation process of hyperedge–replacement grammars if, for each hypergraph H and each derivation of it, the value of H, $f_0(H)$, can be computed from the values of some specific subhypergraphs $H(e)$ determined by the fibres of the derivation. As the examples will show, this view is oversimplified for most applications. To compute the value of H, it might be necessary to compute the values of some other related functions for the $H(e)$'s. Therefore, we use families of functions indexed by some finite set I and we need a mapping $assign$ which determines the values for the $H(e)$'s with respect to the different value functions.

The notion of compatible functions generalizes obviously our earlier notion of compatible predicates (see [HKV 87]). More interesting, a certain type of compatible functions that are composed of minima, maxima, sums, and products induce compatible predicates of the form: the function value of a graph exceeds a given fixed integer, or the function value does not exceed a fixed integer. Consequently, we get the decidability of the problems (1), (2), and (3) in the introduction for these predicates as a corollary.

5.1 Definition (compatible functions)

1. Let $\mathcal{C}$ be a class of hyperedge–replacement grammars, I a finite index set, VAL a set of values, $f : \mathcal{H}_C \times I \to VAL$ a function [8] , and f' a function defined on triples $(R, assign, i)$ with $R \in \mathcal{H}_C$, $assign : E_R \times I \to VAL$, and $i \in I$. Then f is called $(\mathcal{C}, f')$-*compatible* if, for all $HRG = (N, T, P, Z) \in \mathcal{C}$ and all derivations of the form $A^* \Longrightarrow R \stackrel{*}{\Longrightarrow} H$ with $A \in N$ and $H \in \mathcal{H}_T$, and for all $i \in I$,

$$f(H, i) = f'(R, assign, i)$$

where $assign : E_R \times I \to VAL$ is given by $assign(e, j) = f(H(e), j)$ for all $e \in E_R$ and all $j \in I$.
2. A function $f_0 : \mathcal{H}_C \to VAL$ is called $\mathcal{C}$-*compatible* if functions f and f' and an index i_0 exist such that $f_0 = f(-, i_0)$ and f is $(\mathcal{C}, f')$-compatible. [9]

Remark: Intuitively, a function is compatible if it can be computed for a large hypergraph derived by a fibre by computing some values for the smaller components of the corresponding shorter fibres. Such a function must be closed under isomorphisms because the derivability of hypergraphs is independent of the representation of nodes and hyperedges.

5.2 Examples

By Theorem 4.1, the following functions on hypergraphs are $\mathcal{HRG}$–compatible: the number of nodes, the number of hyperedges, and the density of a hypergraph. By Theorems 4.2-4.4, the following functions on graphs are $\mathcal{ERG}$–compatible: the number of simple paths connecting the external nodes, the minimum–path length (of paths connecting the external nodes), the maximum–simple–path length (of paths connecting the external nodes), the number of simple cycles, the minimum–cycle length, the maximum–simple–cycle length, the minimum degree, and the maximum degree of a graph.

We recall now the notion of compatible predicates and relate it with compatible functions.

5.3 Definition (compatible predicates)

1. Let $\mathcal{C}$ be a class of hyperedge–replacement grammars, I a finite index set, $PROP$ a decidable predicate[10] defined on pairs (H, i) with $H \in \mathcal{H}_C$ and $i \in I$, and $PROP'$ a decidable predicate on triples $(R, assign, i)$ with $R \in \mathcal{H}_C$, a mapping $assign : E_R \to I$, and $i \in I$. Then $PROP$ is called $(\mathcal{C}, PROP')$-*compatible* if, for all $HRG = (N, T, P, Z) \in \mathcal{C}$ and all derivations $A^* \Longrightarrow R \stackrel{*}{\Longrightarrow} H$ with $A \in N$ and $H \in \mathcal{H}_T$, and for all $i \in I$, $PROP(H, i)$ holds if and only if there is a mapping $assign : E_R \to I$ such that $PROP'(R, assign, i)$ holds and $PROP(H(e), assign(e))$ holds for all $e \in E_R$.
2. A predicate $PROP_0$ on $\mathcal{H}_C$ is called $\mathcal{C}$-*compatible* if predicates $PROP$ and $PROP'$ and an index i_0 exist such that $PROP_0 = PROP(-, i_0)$ [11] and $PROP$ is $(\mathcal{C}, PROP')$-compatible.

Remarks: 1. Intuitively, a property is compatible if it can be tested for a large hypergraph with a long fibre by checking the smaller components of the corresponding shorter fibres.
2. Examples of compatible properties are: connectivity, planarity, existence of Hamiltonian and Eulerian paths and cycles, k–colorability for each $k \geq 0$ (see [HKV 87] and [Ha 88]).

---

[8] We assume that all considered functions are *closed under isomorphisms*, i.e., for a function f, if $H \cong H'$ for some $H, H' \in \mathcal{H}_C$, then $f(H, i) = f(H', i)$ (resp. $f(H, assign, i) = f(H', assign, i)$) for all $i \in I$.

[9] For $i \in I$, $f(-, i)$ denotes the unary function defined by $f(-, i)(H) = f(H, i)$ for all $H \in \mathcal{H}_C$.

[10] We assume that all considered predicates are *closed under isomorphisms*, i.e., if a predicate Φ holds for $H \in \mathcal{H}_C$ and $H \cong H'$, then Φ holds for H', too.

[11] For $i \in I$, $PROP(-, i)$ denotes the unary predicate defined by $PROP(-, i)(H) = PROP(H, i)$ for all $H \in \mathcal{H}_C$.

3. In [HKV 87] it is shown, that, for all $\mathcal{C}$–compatible properties $PROP_0$, it is decidable whether, given any hyperedge–replacement grammar $HRG \in \mathcal{C}$, $PROP_0$ holds for some $H \in L(HRG)$ and $PROP_0$ holds for all $H \in L(HRG)$.

5.4 Theorem

Let $PROP_0$ be a $\mathcal{C}$–compatible predicate. Then the function $f_0 : \mathcal{H}_\mathcal{C} \to \{0,1\}$ given by

$$f_0(H) = \begin{cases} 1 & \text{if } PROP_0(H) \text{ holds} \\ 0 & \text{otherwise} \end{cases}$$

is $\mathcal{C}$–compatible.

Certain $\mathcal{C}$–compatible functions with values in $\mathbb{N}^\diamond = \mathbb{N} \cup \{\diamond\}$ induce specific $\mathcal{C}$–compatible predicates.

5.5 Definition

1. A function $f : \mathcal{H}_\mathcal{C} \times I \to \mathbb{N}^\diamond$ is said to be $(\mathcal{C}, \min, \max, +, \cdot)$–*compatible* if there exists an f' such that for each right–hand side R of some production in $\mathcal{C}$ and each $i \in I$, $f'(R, -, i)$ corresponds to an expression formed with variables $assign(e,j)$ $(e \in E_R,\ j \in I)$ and constants from $\mathbb{N}$ by addition, multiplication, minimum, and maximum, and f is $(\mathcal{C}, f')$–compatible. The function is $(\mathcal{C}, \max, +, \cdot)$–*compatible* if the operation min does not occur.
2. A function $f_0 : \mathcal{H}_\mathcal{C} \to \mathbb{N}^\diamond$ is $(\mathcal{C}, \min, \max, +, \cdot)$–*compatible* (resp. $(\mathcal{C}, \max, +, \cdot)$–*compatible*) if a function f and an index i_0 exist such that $f_0 = f(-, i_0)$ and f is $(\mathcal{C}, \min, \max, +, \cdot)$–compatible (resp. $(\mathcal{C}, \max, +, \cdot)$–compatible).

5.6 Theorem

Let $f_0 : \mathcal{H}_\mathcal{C} \to \mathbb{N}^\diamond$ be a $(\mathcal{C}, \min, \max, +, \cdot)$–compatible function for some class $\mathcal{C}$ of hyperedge–replacement grammars. Moreover, let $n \in \mathbb{N}^\diamond$. Then the predicates given by "$f_0(H) \leq n$" and "$f_0(H) > n$" are $\mathcal{C}$–compatible. [12]

5.7 Corollary

Let f_0 be a $(\mathcal{C}, \min, \max, +, \cdot)$–compatible function for some class $\mathcal{C}$ of hyperedge–replacement grammars. Moreover, let $n \in \mathbb{N}^\diamond$. Then, for all $HRG \in \mathcal{C}$ the following statements hold.
(1) It is decidable whether (or not) there is some $H \in L(HRG)$ with $f_0(H) \leq n$.
(2) It is decidable whether (or not), for all $H \in L(HRG)$, $f_0(H) \leq n$.
(3) It is decidable in linear time whether (or not) a generated hypergraph $H \in L(HRG)$ represented by a derivation (resp. a derivation tree) has a value $f_0(H) \leq n$.

Proof: Corollary 5.7 follows immediately from the $\mathcal{C}$–compatibility of the predicate "$f_0(-) \leq n$" (see Theorem 5.6) and the theorems for $\mathcal{C}$–compatible predicates given in [HKV 87]). □

6. A Metatheorem for Boundedness Problems

Given a graph-theoretic function f_0 and a class $\mathcal{C}$ of hyperedge–replacement grammars, we are going to study the following type of questions for all $HRG \in \mathcal{C}$: "Is it decidable whether (or not) the values of all hypergraphs generated by HRG are bounded?" The question turns out to be decidable provided that f_0 is $(\mathcal{C}, \max, +, \cdot)$–compatible. We call this result "metatheorem" because of its generic character: Whenever one can prove the $(\mathcal{C}, \max, +, \cdot)$–compatibility of a function (and

[12] We assume that, for all $n \in \mathbb{N}^\diamond$, $\diamond \leq n$.

we have given various examples in section 4), one gets a particular decision result for this function as corollary of the metatheorem.

6.1 Theorem

Let f_0 be a $(\mathcal{C}, \max, +, \cdot)$–compatible function for some class $\mathcal{C}$ of hyperedge–replacement grammars. Then, for all $HRG \in \mathcal{C}$, it is decidable whether or not there is a natural number $n \in \mathbb{N}$ such that $f_0(H) \leq n$ for all $H \in L(HRG)$.

Proof: Let f_0 be a $(\mathcal{C}, \max, +, \cdot)$–compatible function. Let f and f' be the corresponding functions over the index set I so that f is $(\mathcal{C}, f')$–compatible and $f_0 = f(-, i_0)$ for some $i_0 \in I$.
Let $HRG = (N, T, P, Z)$ be a typed and well–formed hyperedge–replacement grammar in $\mathcal{C}$. By Definition 5.1, we may assume that, for each $A \in N$, the grammar $HRG(A) = (N, T, P, A^\bullet)$ is in $\mathcal{C}$, too. ($\mathcal{C}$–compatibility is concerned with productions of a grammar, not with the axiom.)
The proof is based on the following idea. We construct a directed graph D containing all relevant information on derivations in HRG and look for certain cyclic structures in D. This enables us to decide whether or not the values may grow beyond any bound.
Let $J = \{\diamond, 0, 1, \mathit{big}\}$ and $[-] : \mathbb{N}^\diamond \to J$ be the mapping given by $[m] = \mathit{big}$ if $m \geq 2$ and $[m] = m$ otherwise. By (a generalized version of) Corollary 5.7, we can effectively determine the set $EXIST = \{(A, p : I \to J) | \exists H \in \mathcal{H}_T : A^\bullet \stackrel{*}{\Longrightarrow} H \ \wedge \ \forall j \in I : [f(H, j)] = p(j)\}$. $H \in \mathcal{H}_T$ is said to be an (A,p)–*hypergraph* if H can be derived from the handle induced by A and, for all $j \in I$, $[f(H, j)] = p(j)$. We define a directed graph D with two types of edges, called *greaterequal–edges* and *greater–edges* as follows. Let $V = \{(A, p, i) | (A, p) \in EXIST \wedge p(i) = \mathit{big}\}$ be the node set of D. The edge set of D is determined as follows: Let $(A, R) \in P$ be a production of HRG, $q : E_R \times I \to J$ a function such that, for all $e \in E_R$, $(l_R(e), q(e, -)) \in EXIST$, $i \in I$ an index, and $[f'(R, q, i)] = \mathit{big}$. Moreover, let $p : I \to J$ be the function given by $p(j) = [f'(R, q, j)]$ for $j \in I$.
By assumption, the function f is $(\mathcal{C}, \max, +, \cdot)$–compatible. Since multiplication distributes over addition and maximum and addition distributes over maximum, we may assume that $f'(R, -, i)$ is a maximum of sums, each formed from products of constants and variables $\mathit{assign}(e, j)$ ($e \in E_R$, $j \in I$). Substitute $\mathit{assign}(e, j)$ by $q(e, j)$, if $q(e, j) \in \{\diamond, 0, 1\}$, and simplify, i.e., delete all sums that evaluate to $\diamond$, all products that evaluate to 0 and all factors that evaluate to 1.

- If some sum simply is $\mathit{assign}(e, j)$, then we add an edge from (A, p, i) to $(l_R(e), q(e, -), j)$ in D, a so called *greaterequal–edge*, denoted by $(A, p, i) \Rightarrow (l_R(e), q(e, -), j)$.
- If some sum contains $\mathit{assign}(e, j)$, but also a non–trivial factor or some other product, then we add a so called *greater–edge* from (A, p, i) to $(l_R(e), q(e, -), j)$ in D, denoted by $(A, p, i) \to (l_R(e), q(e, -), j)$.

In the following, we will show that the graph D contains all information to decide whether or not some function values grow beyond any bound. It turns out that the greater–edges of D play an important role. Remember that for each (B, p', j) in D, there is at least one derivation $B^\bullet \stackrel{*}{\Longrightarrow} G$ in HRG with $[f(G, -)] = p'$ and $f(G, j) \geq 2$. We will show that, whenever we have a derivation $B^\bullet \stackrel{*}{\Longrightarrow} G$ in HRG with $[f(G, -)] = p'$ and $f(G, j) \geq 2$ and there is a greater–edge $(A, p, i) \to (B, p', j)$ in D, then there exists a derivation $A^\bullet \stackrel{*}{\Longrightarrow} H$ in HRG with $[f(H, -)] = p$ and $f(H, i) > f(G, j)$.

Claim 1: Let $(A, p, i), (B, p', j) \in V$ and G be a (B, p')–hypergraph.
(1) If $(A, p, i) \to (B, p', j)$, then there is an (A, p)–hypergraph H with $f(H, i) > f(G, j)$.
(2) If $(A, p, i) \Rightarrow (B, p', j)$, then there is an (A, p)–hypergraph H with $f(H, i) \geq f(G, j)$.

Proof of Claim 1: Let $(A, p, i) \to (B, p', j)$ be a greater–edge in D and G be an arbitrary (B, p')–hypergraph. By construction of D, there is some production $(A, R) \in P$ and some $q : E_R \times I \to J$ such that, for all $e \in E_R$, $(l_R(e), q(e, -) \in EXIST$. Moreover, there is some $e' \in E_R$ with $l_R(e') = B$ and $q(e', -) = p'$. By definition of $EXIST$, for each $e \in E_R$, there exists a derivation $l_R(e)^\bullet \stackrel{*}{\Longrightarrow} H(e)$ such that $[f(H(e), -)] = q(e, -)$. Since $l_R(e') = B$ and G

is an (B,p')–hypergraph, there exists a derivation $l_R(e')^\bullet \stackrel{*}{\Longrightarrow} G$ such that $[f(G,-)] = p'$. Joint Embedding of the derivations $l_R(e)^\bullet \stackrel{*}{\Longrightarrow} H(e)$ for $e \in E_R - \{e'\}$ and the derivation $l_R(e')^\bullet \stackrel{*}{\Longrightarrow} G$ – instead of $l_R(e')^\bullet \stackrel{*}{\Longrightarrow} H(e')$ – into R yields a derivation $R \stackrel{*}{\Longrightarrow} H$. Combining it with the direct derivation $A^\bullet \Longrightarrow R$, we get a derivation $A^\bullet \stackrel{*}{\Longrightarrow} H$. By the $(\mathcal{C}, \max, +, \cdot)$–compatibility of f, H is an (A,p)–hypergraph: For all $j \in I$, we have $[f(H,j)] = [f'(R, assign', j)] = [f'(R, [assign'], j)]$ $= [f'(R, [assign], j)] = [f'(R,q,j)] = p(j)$ where $assign'(e,-) = assign(e,-) = f(H(e),-)$ for $e \in E_R - \{e'\}$, $assign'(e',-) = f(G,-)$, and $assign(e',-) = f(H(e'),-)$. Moreover, by the special choice of the edges of D, $f(H,i) = f'(R, assign', i) > assign'(e',j) = f(G,j)$. [Observe that in the sum leading to the creation of $(A,p,i) \to (B,p',j)$ all remaining variables are substituted by at least 2.] Analogously, if $(A,p,i) \Rightarrow (B,p',j)$ is a greaterequal–edge in D, we get $f(H,i) \geq f(G,j)$. □

In the following, we will look for special structures in D, called lasso structures. A subgraph L of D is called a *lasso structure* if it contains for each node a unique outgoing edge and each cycle contains a greater–edge. A node (A,p,i) of D is said to be *unbounded*, if, for all $n \in \mathbb{N}$, there is an (A,p)–hypergraph H with $f(H,i) > n$; otherwise it is said to be *bounded*.

Claim 2: Let L be a lasso structure in D. Then every (A,p,i) in L is unbounded.

Proof of Claim 2: Assume to the contrary and let k be minimal such that, for some (A,p,i) in L, for every (A,p)–hypergraph H we have $f(H,i) \leq k$. By the above claim we have for the unique successor (B,p',j) of (A,p,i) in L and every (B,p')–hypergraph H that $f(H,j) \leq k$. By choice of k, there must exist a (B,p')–hypergraph H with $f(H,j) = k$ and we have $(A,p,i) \Rightarrow (B,p',j)$. Repeating this consideration we eventually get a lasso [13] in L whose cycle has greaterequal–edges only, a contradiction. □

Claim 3: There exists a lasso structure L in D containing all unbounded (A,p,i).

Proof of Claim 3: Let k be the maximal $f(H,i)$, where H is an (A,p)–hypergraph such that (A,p,i) is bounded, but at least 2. Moreover, let $\Phi(k)$ be determined as follows: Note that each $f'(R,-,i)$ with R a right–hand side of some production of HRG, $i \in I$, can be expressed as a maximum of sums of products of variables and constants. Evaluate each of all these sums by replacing each variable by $k \in \mathbb{N}^\diamond$, and let $\Phi(k)$ be the maximum of these values plus 1. In the following, we define a subgraph L of D iteratively using sets OK and NOK, such that the following properties hold after each step:

(1) $OK \cup NOK = \{(A,p,i) \in V | (A,p,i)$ is unbounded$\}$;
(2) $OK \cap NOK = \emptyset$;
(3) $OK \subseteq V_L \subseteq OK \cup NOK$;
(4) each node in OK has a unique outgoing edge in L;
(5) each cycle of L contains a greater–edge;
(6) each maximal path [14] of L ends with a greater–edge.

Initially, let $OK = \emptyset$, $NOK = \{(A,p,i) \in V | (A,p,i)$ is unbounded$\}$, and L be the empty graph. For the iteration step, choose a derivation $A^\bullet \stackrel{*}{\Longrightarrow} H$ of minimal length such that H is an (A,p)–hypergraph with $f(H,i) \geq \Phi(k)$ and $(A,p,i) \in NOK$. Let (A,R) be the first production of this derivation. We have hypergraphs $H(e)$, $e \in E_R$, and some $q : E_R \times I \to J$ such that $H(e)$ is an $(l_R(e), q(e,-))$–hypergraph for $e \in E_R$. $f'(R,-,i)$ in its simplified normal form is a maximum of sums, and, by definition of k and Φ, the maximum is attained for a sum containing a variable $assign(e,j)$ such that $f(H(e),j) > k$. Put $(B,p',j) = (l_R(e), q(e,-), j)$, $OK = OK \cup \{(A,p,i)\}$, $NOK = NOK - \{(A,p,i)\}$, add to L the corresponding edge from (A,p,i) to (B,p',j) and – if necessary – (A,p,i) and/or (B,p',j). The first four conditions on L given above hold true (we have $f(H(e),j) > k$, therefore, $(B,p',j) \in OK \cup NOK$). If the new edge is a greater–edge, then

[13] If we add to a path $v_1 \ldots v_n$, which has distinct nodes by definition, an edge $v_n v_i$, $i \in \{1, \ldots, n-1\}$, then the resulting graph is called a *lasso*.

[14] We call a path *maximal*, if its last node has outdegree 0.

each new cycle contains it, each new non–trivial maximal path ends with it ($(B,p',j) \notin OK$) or ends with a non–trivial maximal path that already existed ($(B,p',j) \in OK$). If the new edge is a greaterequal–edge, we must have $f(H(e),j) \geq \Phi(k)$, thus $(B,p',j) \in OK$, since we have chosen a shortest derivation. Hence any new non–trivial maximal path ends with an old one starting at (B,p',j). If there are new cycles, then we already had $(A,p,i) \in V_L$ and any edge leading to (A,p,i) is a greater–edge.
Since the set $\{(A,p,i) \in V | (A,p,i)$ is unbounded$\}$ is finite, the construction is finished after a finite number of steps. After these steps, $OK = \{(A,p,i) \in V | (A,p,i)$ is unbounded$\}$ and $NOK = \emptyset$. Moreover, by (3), (4), and (5), $V_L = OK$, each node of L has a unique outgoing edge in L, and each cycle of L contains a greater–edge. Consequently, the constructed L is a lasso structure and, since $V_L = \{(A,p,i) \in V | (A,p,i)$ is unbounded$\}$, L contains all unbounded (A,p,i). □

Now we may proceed as follows: (1) Construct the graph D for HRG. (2) Check for each subgraph of D whether it is a lasso structure. (3) Check for each lasso structure L whether it contains $(l(Z),p,i_0)$ for some $p : I \to J$.
If there is a lasso structure L in D containing $(l(Z),p,i_0)$ (for some p), then, by Claim 2, $(l(Z),p,i_0)$ is unbounded, meaning that, for all $n \in \mathbb{N}$, there is an $(l(Z),p)$–hypergraph H with $f(H,i_0) > n$. Hence, for all $n \in \mathbb{N}$, there is a hypergraph $H \in L(HRG)$ with $f_0(H) > n$.
Conversely, if, for all $n \in \mathbb{N}$, there is a hypergraph $H \in L(HRG)$ with $f_0(H) > n$, then, for all $n \in \mathbb{N}$, there is a p and an $(l(Z),p)$–hypergraph H with $f(H,i_0) > n$. Since the number of p's is finite, we can find some p such that, for all $n \in \mathbb{N}$, there is an $(l(Z),p)$–hypergraph H with $f(H,i_0) > n$. Therefore, $(l(Z),p,i_0)$ is unbounded and, by Claim 3, there exists a lasso structure containing $(l(Z),p,i_0)$. This completes the proof of the theorem. □

Combining the compatibility results of Section 4 and Theorem 6.1, one obtains a list of decidability results concerning boundedness problems.

6.2 Corollary

For each edge–replacement grammar $ERG \in \mathcal{ERG}$ and each function in the following list, it is decidable whether (or not) the function values of the graphs in $L(ERG)$ grow beyond any bound: the number of nodes, the number of edges, the number of simple paths connecting the external nodes, the number of simple cycles, the maximum–simple–path length of paths connecting the external nodes, the maximum–simple–cycle length, and the maximum degree of a graph.

Proof: The statements follow directly from the theorems 4.1-4.4 and 6.1. □

Remarks: 1. Remember that the functions "number of nodes" and "number of hyperedges" are compatible for arbitrary hyperedge–replacement grammars $HRG \in \mathcal{HRG}$.
2. Although we avoided the troublesome technicalities in this paper, we are convinced that the other considerations of this section work for more general types of hyperedge–replacement grammars, too. For example, all the statements should hold even if the class $\mathcal{ERG}$ is replaced by the class of all hyperedge–replacement grammars which generate ordinary graph languages and use hyperedges with a bounded number of tentacles as nonterminals. We even think that the considered functions are compatible for arbitrary hyperedge–replacement grammars if their definition is properly adapted to hypergraphs.

Finally, let us mention that some problems — like the connectivity problem, the maximum–clique–size problem, and the chromatic–number problem — are trivial in the following sense: for all hyperedge–replacement grammars HRG, there is a bound (depending only on HRG) such that the function values of all graphs do not exceed the bound. This knowledge can be used to show that other boundedness problems — as the minimum–clique–covering problem and the maximum–indepentent–set problem — are decidable.

The *clique partition number* of a graph G, $C(G)$, is the smallest number of cliques that form a partition of the node set V_G. A set of nodes in a graph G is *independent* if no two of them are adjacent. The largest number of nodes in such a set is called the *independence number* of G and is denoted by $I(G)$.

6.3 Theorem

For each hyperedge–replacement grammar $HRG \in \mathcal{HRG}$ generating a set of graphs, it is decidable whether (or not) the clique partition number and the independence number of graphs in $L(HRG)$ grows beyond any bound.

Proof: Since for each hyperedge–replacement grammar HRG, the maximum clique size is bounded on $L(HRG)$, say by $c(HRG) \geq 1$, and, for each $G \in L(HRG)$,

$$\frac{|V_G|}{c(HRG)} \leq C(G) \leq |V_G|,$$

the clique partition number is bounded on $L(HRG)$ if and only if the number of nodes is bounded on $L(HRG)$. Since for each hyperedge–replacement grammar HRG the chromatic number is bounded on $L(HRG)$, say by $k(HRG) \geq 1$, for each $H \in L(HRG)$, the maximum number of equally colored nodes in a $k(HRG)$–coloring of G, $MAX(G)$, is a lower bound of $I(G)$. On the other side, $|V_G| \leq k(HRG) \cdot MAX(G)$. Thus,

$$\frac{|V_G|}{k(HRG)} \leq I(G) \leq |V_G|.$$

Therefore, the independence number is bounded on $L(HRG)$ if and only if the number of nodes is bounded on $L(HRG)$. □

7. Discussion

Each class $\mathcal{C}$ of graph grammars and each function f on graphs with integer values establish a Boundedness Problem:

> Is it decidable, for all graph languages $L(GG)$ generated by GG in $\mathcal{C}$, whether or not there is a bound n such that $f(G) \leq n$ for all $G \in L(GG)$?

In this paper, we have been able to show that the Boundedness Problem is solvable for classes of hyperedge–replacement grammars and functions that are compatible with the derivation process and where the values of derivable graphs are composed of maxima, sums, and products of component values. Although this result applies to a variety of examples it seems to be strangely restricted. Further research should clarify the situation:

(1) We would expect that the metatheorem holds under more general or modified assumptions. Especially, we would like to know how functions given by minima or differences or divisions work.

(2) We suspect that certain combinations of arithmetic operations are not allowed. For instance, maxima and minima seem to antagonize each other — at least sometimes.

(3) Compatible functions are defined for arbitrary domains. But we have got significant results only for boolean and integer values. What about other domains? How can be arbitrary compatibility be exploited? How do other meaningful interpretations look like?

References

[ALS 88] S. Arnborg, J. Lagergren, D. Seese: Problems Easy for Tree–Decomposable Graphs, Proc. ICALP'88, Lect. Not. Comp. Sci. 317, 38-51, 1988

[BC 87] M. Bauderon, B. Courcelle: Graph Expressions and Graph Rewriting, Math. Systems Theory 20, 83-127, 1987

[Co 87] B.Courcelle: On Context–Free Sets of Graphs and Their Monadic Second–Order Theory, Lect. Not. Comp. Sci. 291, 133-146, 1987

[DG 78] P. Della Vigna, C. Ghezzi: Context–Free Graph Grammars, Inf. Contr. 37, 207-233, 1978

[Ha 88] A. Habel: Graph–Theoretic Properties Compatible with Graph Derivations, to appear in: Proc. Graph–Theoretic Concepts in Computer Science 1988 (WG'88), Lect. Not. Comp. Sci., 1988

[HK 83] A. Habel, H.-J. Kreowski: On Context–Free Graph Languages Generated by Edge Replacement, Lect. Not. Comp. Sci. 153, 143-158, 1983

[HK 85] A. Habel, H.-J. Kreowski: Characteristics of Graph Languages Generated by Edge Replacement, University of Bremen, Comp. Sci. Report No. 3/85, also in: Theor. Comp. Sci. 51, 81-115, 1987

[HK 87a] A. Habel, H.-J. Kreowski: May We Introduce to You: Hyperedge Replacement, Lect. Not. Comp. Sci. 291, 15-26, 1987

[HK 87b] A. Habel, H.-J. Kreowski: Some Structural Aspects of Hypergraph Languages Generated by Hyperedge Replacement, Proc. STACS'87, Lect. Not. Comp. Sci. 247, 207-219, 1987

[HKV 87] A. Habel, H.-J. Kreowski, W. Vogler: Metatheorems for Decision Problems on Hyperedge Replacement Graph Languages, to appear in Acta Informatica, short version with the title "Compatible Graph Properties are Decidable for Hyperedge Replacement Graph Languages" in: Bull. EATCS 33, 55-62, 1987

[JRW 86] D. Janssens, G. Rozenberg, E. Welzl: The Bounded Degree Problem for NLC Grammars Is Decidable, Journ. Comp. Syst. Sci. 33, 415-422, 1986

[Kr 79] H.-J. Kreowski: A Pumping Lemma for Context–Free Graph Languages, Lect. Not. Comp. Sci. 73, 270-283, 1979

[La 88] C. Lautemann: Decomposition Trees: Structured Graph Representation and Efficient Algorithms, Proc. CAAP'88, Lect. Not. Comp. Sci. 299, 28-39, 1988

[LW 88] T. Lengauer, E. Wanke: Efficient Analysis of Graph Properties on Context–Free Graph Languages, Proc. ICALP'88, Lect. Not. Comp. Sci. 317, 379-393, 1988

[RW 86a] G. Rozenberg, E. Welzl: Boundary NLC Graph Grammars — Basic Definitions, Normal Forms, and Complexity, Inf. Contr. 69, 136-167, 1986

[RW 86b] G. Rozenberg, E. Welzl: Graph Theoretic Closure Properties of the Family of Boundary NLC Graph Languages, Acta Informatica 23, 289-309, 1986

[Sl 82] A.O. Slisenko: Context–Free Graph Grammars as a Tool for Describing Polynomial–Time Subclasses of Hard Problems, Inf. Proc. Lett. 14, 52-56, 1982