Constant-memory validation of streaming XML documents against DTDs

Luc Segoufin and Cristina Sirangelo

INRIA and Université Paris 11

Abstract. In this paper we investigate the problem of validating, with constant memory, streaming XML documents with respect to a DTD. Such constant memory validations can only be performed for some but not all DTDs. This paper gives a non trivial interesting step towards characterizing those DTDs for which a constant-memory on-line algorithm exists.

1 Introduction

The Extended Markup Language (XML) is emerging as the standard for data exchange on the Web. Many applications require on-line processing of large amounts of data in XML format using limited memory. Such processing includes querying XML documents, computing running aggregates of streams of numerical data, and validating XML documents against given Document Type Definitions (DTDs). For each query, for each aggregate and for each DTD, one issue is then to see what would be the minimal amount of memory which is really needed in order to process it on-line.

In this paper we are concerned with those validation problems that can be processed on-line and using a constant amount of memory. The problem of validating XML documents against a given DTD is to find out whether the document conforms the specification given by the DTD. We consider only simple DTDs that do not have any integrity constraint, and we want to perform this validation on-line. As we consider only simple DTDs, data values are not relevant for validation, and we can view our XML document as a stream of symbols representing the sequence of opening/closing tags of the document. Given such a stream, in a single pass and using a fixed amount of memory, depending on the DTD, but not on the size of the XML document, we want to be able to tell whether the document conforms the DTD or not. In other words, we are looking for a finite-state automaton (FSA) performing a pass on the XML document, as it streams through the network, and testing conformance with the DTD. An easy observation shows that this is not always possible for all DTDs ([4]).

As pointed out in [4], a FSA can certainly not check that the document is well-formed. By this we mean that the sequence of opening/closing tags is well balanced. But even if we take this for granted, and this is what we are going to do in this paper, many DTDs cannot be validated on-line using a FSA.

In this paper we tackle the question of finding those DTDs that can be validated on-line using a FSA. We call such DTDs *streamable* . The main questions we address are: Which are the streamable DTDs? Is it decidable whether a DTD is streamable? If a DTD is streamable can we compute a FSA which performs the validation?

We don't provide a full answer to these questions, but we make a significant step towards answering them.

A simple observation made in [4] shows that if a DTD is not recursive then it is streamable. When the DTD is recursive, a FSA gets immediately lost in the depth of the tree and a first intuition that one could have is that it can only check locally whether two successive tags are consistent with those appearing in the DTD. This was the approach taken in [4]. Given a DTD τ, a *local-automaton* [1] for τ can be constructed which checks that each two successive letters are consistent with those appearing in τ. The hope was to prove that a DTD is streamable iff the set of trees accepted by the local-automaton for τ equals the set of trees valid for τ. In [4] it was shown that this is indeed the case for so called "fully-recursive" DTDs, but the paper ended with an example of a DTD showing that doing modulo-2 counting on the number of occurrences of two successive letters could be necessary to validate it on-line.

We thus generalize the notion of local-automaton by extending it using an arbitrary finite group operation on the occurrences of two successive letters. In that respect, a modulo-counting operation corresponds to the case finite groups generated by a single element. Given any finite group H and any DTD τ we define a notion of H-local-automaton for τ which extends local-automaton by combining it with computation in H. We conjecture that H-local-automata capture the notion of streamability: a DTD τ is streamable iff there exists a finite group H such that the *H-local-automaton* for τ defines the same set of trees than τ. We give a necessary and sufficient condition on a DTD τ to admit a H-local-automaton. This condition is expressed in terms of a word problem for finite groups. Unfortunately we don't know yet whether this condition is decidable or not. Recall that the word problem for finite groups is undecidable in general [3, 5].

We also provide a decidable necessary and sufficient criterion on the DTDs τ for which there exists a finite commutative group H such that the H-local-automaton for τ defines the same set of trees as τ.

Maybe one of our most interesting contribution lies in the concepts we develop here in order to obtain our results. We believe that those will eventually be sufficient for finding the right characterization. We also think that they could be used in other contexts.

Related work This paper can be seen as a continuation of [4]. In [4] several necessary conditions were given for a DTD to be streamable. We reuse one of them in an essential way in this paper, while the others will follow from our results. Those conditions were obtained using the notion of local-automaton that is also the starting brick of our construction here. In [4] a decidable characterization of DTDs streamable by a local-automaton was also given. Here we extend this result by providing a decidable characterization of DTDs streamable using a H-local-automaton for some finite commutative group H. The techniques we use in this paper are completely different than the one used in [4]. We have good hope that these new techniques could be pushed to eventually obtain the complete characterization of streamable DTDs.

[1] This automaton was called "standard automaton" in [4], but we believe that this terminology is misleading

The work of [4] was also continued in [2]. In this paper some limited amount of memory was allowed by using restricted pushdown automata instead of FSA.

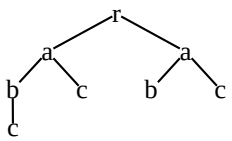
Testing whether a DTD is streamable can be seen as the problem of deciding which subclass of regular tree languages a FSA could accept when trees are coded à la XML, using a well-formed sequence of opening and closing tags. With this coding the string $ab\bar{b}c\bar{c}\bar{a}$ codes the tree rooted in a node labeled with a and having two children, the left one labeled with b and the right one labeled with c. The same question naturally arises with any other coding for trees. For instance one could use the functional coding which codes the tree above with the string $a(b()c())$. Using this coding one could now ask which are the regular tree languages a FSA could recognize. It is easy to see that this class is strictly contained in the class of tree languages recognized by a FSA using the XML coding. The reason is that when reading a closing bracket in the functional coding the FSA does not know the label of the node this bracket closes, while this is known in the case of the XML coding. In [1] a decidable characterization of streamable languages using the functional coding was given. It seems quite difficult to extend their ideas to the XML coding.

The paper is organized as follows. After introducing the necessary background notations in Section 2, we define in Section 3 the central notions of this paper: Graph of a DTD and separating group for a DTD. In Section 4 we define, for any finite group H, the notion of H-local-automaton for a DTD and show that the existence of a separating group for a DTD is equivalent to the existence of a H-local-automaton accepting exactly all the valid trees for this DTD. Finally in Section 5 we give a decidable characterization of those DTDs having a H-local-automaton, for some finite commutative group H, which accepts exactly all the valid trees.

2 Notations

We fix a finite set of labels Σ.

Trees. A *tree* with labels in Σ is a finite unranked ordered tree whose nodes have labels from Σ. To capture the on-line behavior, we will manipulate trees via string representations corresponding to a depth-first traversal or, equivalently, to the sequence of opening/closing tags of the document represented by t. To this end we view Σ as the set of opening tag symbols while $\bar{\Sigma} = \{\bar{a} \mid a \in \Sigma\}$ is the set of closing tag symbols. Now the string representation of a tree t is the string, also denoted by t, defined by induction as: if t has a single node of label a, then $t = a\bar{a}$. It t consists of a root labeled a and subtrees $t_1 \dots t_k$ then t is the string $a\ t_1 \dots t_k\ \bar{a}$.

For instance the string representation of the tree 

is the string $rabc\bar{c}\bar{b}bc\bar{c}\bar{a}ab\bar{b}bc\bar{c}\bar{a}\bar{r}$. We denote by $\mathcal{L}_{\text{tree}}$ the set of (string representation of) trees.

DTDs. A DTD consists of an extended context-free grammar where each rule associates to a label $a \in \Sigma$ a regular expression r_a over Σ, together with a distinguished initial symbol. A tree t is conform to a DTD τ (or t is valid w.r.t τ) if the label of its root is the label of the initial symbol of τ and, for each node $x \in t$ of label a, the sequence of labels of the children of x form a word of r_a. For instance the tree above is valid for the DTD[2]: $\begin{array}{l} r \to a^* \\ a \to bc \\ b \to c^? \\ c \to \epsilon \end{array}$. Since regular expressions are closed under union, we can assume w.l.o.g. that each DTD has a unique rule $a \to r_a$ for each symbol $a \in \Sigma$.

Each DTD τ, defines a language of trees, denoted $\mathcal{L}(\tau)$ consisting of all (string representation of) trees valid for τ.

Streaming. We are interested in DTDs τ whose membership problem can be solved using a finite memory device assuming that (the string representation of) the input tree is well formed (is in $\mathcal{L}_{\text{tree}}$). More formally we say that a DTD τ is *streamable* if there exists a regular language R over Σ such that $\mathcal{L}(\tau) = \mathcal{L}_{\text{tree}} \cap R$. If R is such that $\mathcal{L}(\tau) = \mathcal{L}_{\text{tree}} \cap R$ we say that R *recognizes* τ. For instance the DTD τ: $r \to a, a \to a \mid \epsilon$ is streamable as $\mathcal{L}(\tau) = \{ra^n\bar{a}^n\bar{r} \mid n \in \mathbb{N}\}$ which is $\mathcal{L}_{\text{tree}} \cap ra^*\bar{a}^*\bar{r}$. On the other hand it is not too difficult to show that the DTD $r \to aa, a \to a \mid \epsilon$ is not streamable. We are looking for a decidable characterization of streamable DTDs. In order to do so we associate in Section 3 a graph to any DTD and show in Section 4 how to construct from this graph a family of automata that could recognize the corresponding DTD.

3 DTDs, graphs and groups

In this Section we introduce the machinery necessary for stating our results.

Decomposition of a DTD. Given a DTD τ, we define a pre-order $\leq_\tau$ on Σ as follows. A label b is a successor of a label a relative to τ if there is a word w of r_a containing the label b. We then simply set $\leq_\tau$ as the reflexive transitive closure of this successor relation. This pre-order induces an equivalence relation $\sim_\tau$ on Σ: $a \sim_\tau b$ if $a \leq_\tau b \wedge b \leq_\tau a$. The set C_τ of equivalence classes of $\sim_\tau$ is now partially ordered by $\leq_\tau$.

Example 1. If τ is the following DTD: $\begin{array}{ll} r \to abc & a \to c \\ c \to edc \mid \epsilon & b \to a \\ d \to ad \mid ed \mid eb \mid \epsilon & e \to b \end{array}$
C_τ contains two equivalence classes, $\{r\}$ and $\{a, b, c, d, e\}$ and $r \leq_\tau a$.

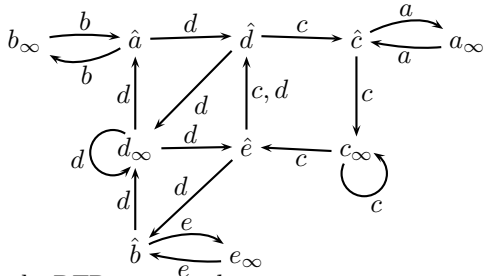
Graph of a DTD. We now define the central notion used in this paper. For each class **c** of C_τ, we construct the labelled directed graph $G_\tau(\mathbf{c})$, denoted as the *graph of* **c** relative to τ. The intuition is that the graph of **c** relative to τ codes all the transitions between two successive letters of **c** occurring in τ.

[2] $c^?$ is an abbreviation for $(c|\epsilon)$.

More formally, the set of vertices of $G_\tau(\mathbf{c})$ is defined as $\{\hat{a} | \ a \in \mathbf{c}\} \cup \{a_\infty \mid a \in \mathbf{c}\}$. The nodes in $\{\hat{a} \mid a \in \mathbf{c}\}$ are called *inner nodes*. If v is a node of $G_\tau(\mathbf{c})$, $l(v)$ denotes the label $a \in \mathbf{c}$ such that $v = \hat{a}$ or $v = a_\infty$. Given three labels a, b, d of **c**, there is an edge of label d from $\hat{a}$ to $\hat{b}$ in $G_\tau(\mathbf{c})$ whenever there is a word $w = w_1 a w_2 b w_3$ in r_d such that all the labels occurring in w_2 are not in **c** (by definition they must belong to classes **c'** of C_τ with $\mathbf{c} <_\tau \mathbf{c}'$). Given two labels a, d of **c**, there is an edge of label d from d_∞ to $\hat{a}$ whenever there is a word $w = w_1 a w_2$ in r_d such that all the labels occurring in w_1 are not in **c**. Given two labels a, d of **c**, there is an edge of label d from $\hat{a}$ to d_∞ whenever there is a word $w = w_1 a w_2$ in r_d such that all the labels occurring in w_2 are not in **c**. Given a label d of **c**, there is an edge of label d from d_∞ to d_∞ whenever there is a word w in r_d such that all the labels occurring in w are not in **c**. No other edges occurs in $G_\tau(\mathbf{c})$. We view $G_\tau(\mathbf{c})$ as a simple directed graph. That is, whenever there are several edges, with different labels, going from vertex $\hat{a}$ to vertex $\hat{b}$, we replace them with a single edge whose label is the union of all the previous labels. The graph G_τ is the disjoint union of all $G_\tau(\mathbf{c})$, $\mathbf{c} \in C_\tau$.

We illustrate this central concept with three examples that will be our running examples for the paper.

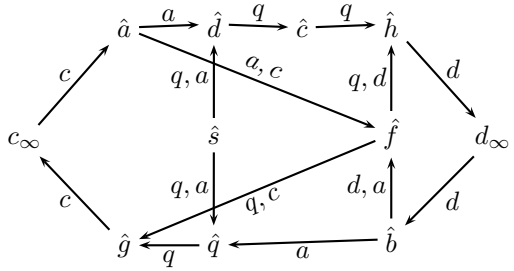
Continuation of Example 1. The graph of this DTD is (ignoring the trivial class containing only r):



Example 2. Consider now the DTD τ:

$$
\begin{array}{ll}
r \to abc & b \to a \\
a \to ad \mid af \mid sd \mid sq \mid bf \mid bq \mid \epsilon & f \to q \\
q \to sdch \mid sqg \mid fh \mid fg \mid \epsilon & g \to q \\
c \to afg & h \to q \\
d \to bfh & s \to q
\end{array}
$$

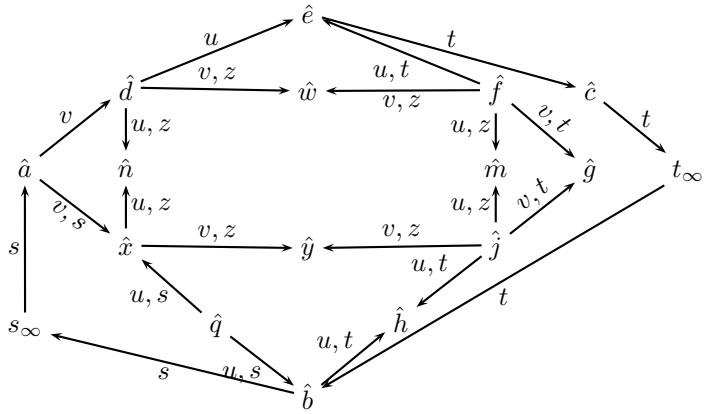
C_τ contains again two equivalence classes, one for $\{r\}$ and one for the remaining letters (note that the last 5 rules are only here to make all the symbols but r equivalent according to $\sim_\tau$, they don't affect much the graph and are irrelevant for the rest of the paper). The graph for this DTD looks like this. For the sake of simplicity we have ignored the nodes $a_\infty, b_\infty, f_\infty, g_\infty, h_\infty, s_\infty, q_\infty$ and their corresponding edges which will not be relevant in the sequel.



Example 3. Our third running example is with the DTD:

$r \to abc$	$j \to m$	$y \to j$
$u \to de \mid fe \mid fm \mid bh \mid jh \mid jm \mid qb \mid qxn \mid dn$	$w \to y$	$a \to u$
$z \to xn \mid dn \mid dw \mid fw \mid fm \mid jm \mid jy \mid xy$	$b \to t$	$m \to s$
$t \to bh \mid jh \mid jg \mid fg \mid fec$	$h \to u$	$c \to z$
$v \to adw \mid axy \mid jy \mid jg \mid fg \mid fw$	$f \to g$	$n \to w$
$s \to ax \mid qx \mid qb$	$g \to v$	$d \to f$
$e \to t$	$q \to u$	$x \to n$

Again C_τ contains two equivalence classes, one with only r and one with all the other labels. All the last rules are again irrelevant for the rest of the paper, they are only needed to have a unique class containing all the symbols but r. The graph is depicted below:



Paths. Given the graph G of a DTD, a path p is an arbitrary sequence of vertices such that for any two consecutive vertices of this sequence, there is an edge between them (not necessarily from the first one to the second one). A path p is *directed* if it traverses the edges in the direction induced by it. A path is *simple* if it traverses a vertex at most once. A path is a *cycle* if it starts and ends at the same vertex of G. A path of G_τ is *internal* if all its nodes, besides the first and last one, are inner nodes.

Continuation of Example 1. The path $b_\infty \hat{a} \hat{d} \hat{c}$ is simple, internal and directed. The path $\hat{d}\hat{e}d_\infty\hat{a}\hat{d}$ is a simple non-directed non-internal cycle.

Languages of internal paths. For each edge of G_τ from vertex $\hat{a}$ to vertex $\hat{b}$ we define $\mathcal{L}_\tau(\hat{a}, \hat{b})$ as the set of words $w \in \Sigma^*$, such that all letters of w are in a class strictly higher that the class of a, and there exists a label d of the edge such that $w_1 a w b w_2 \in r_d$ for some arbitrary strings w_1 and w_2. Similarly we define $\mathcal{L}_\tau(a_\infty, \hat{b})$, $\mathcal{L}_\tau(\hat{a}, b_\infty)$ and $\mathcal{L}_\tau(a_\infty, a_\infty)$. We extend this notion to any directed internal path p. Assume p is $v_1 \cdots v_n$. We set $\mathcal{L}_\tau(p) = \mathcal{L}_\tau(v_1, v_2)\, l(v_2)\, \mathcal{L}_\tau(v_2, v_3)\, l(v_3) \cdots\ l(v_{n-1})\, \mathcal{L}_\tau(v_{n-1}, v_n)$. Finally for each symbol d, let $\mathcal{L}_\tau(d)$ to be the union over all directed internal paths p, starting and ending in d_∞, of $\mathcal{L}_\tau(p)$. Note that we take this union over all directed internal paths, not just the simple ones (the path can go several time through the same node).

Continuation of Example 1. In this example we have $\mathcal{L}_\tau(d) = \{ad, eb, ed, \epsilon\}$ and $\mathcal{L}_\tau(c) = \{edc, \epsilon\}$.

NECESSARY CONDITION: A DTD τ satisfies condition $(*)$ if for all d, $\mathcal{L}_\tau(d) = r_d$.

All the three DTDs given in Example 1,2 and 3 satisfy the condition $(*)$. On the other hand, if we replace in the DTD of Example 1 the line $d \rightarrow ad \mid ed \mid eb \mid \epsilon$ with $d \rightarrow ad \mid eb \mid \epsilon$ the edges of the underlying graph remain the same (the new graph differs from the previous one only by the label of edge $(\hat{e}, \hat{d})$ which no longer contains d) but the DTD no longer satisfies the condition $(*)$. It follows from Theorem 1 below that this DTD is not streamable.

The following result is a rephrasing of the first necessary condition for streamability proved in [4].

Theorem 1. *If a DTD τ is streamable then it satisfies $(*)$.*

Based on this result, in the sequel we will represent our DTDs using their graph representation. We aim at characterizing those graphs that represent streamable DTDs. In order to do this we use the notions of *monochromatic cycles* and *dangerous cycles* that we introduce now.

Monochromatic cycles and dangerous cycles. The set MCycles(τ) of *monochromatic cycles of* τ is the set of all simple directed cycles p of G_τ such that there is a label $a \in \mathbf{c}$ which occurs as a label of all edges traversed by p.

A directed path of $G_\tau(\mathbf{c})$ from $\hat{a}$ to $\hat{b}$, is a *source path* if it consists of a simple internal directed path from $\hat{a}$ to d_∞, followed by a simple internal directed path from d_∞ to $\hat{b}$, for some label d of **c**. A directed cycle p of G_τ is *dangerous* if the following holds:

- p traverses successively vertices $\hat{a}_n, \cdots, \hat{a}_1$ of G_τ by forming a source path from $\hat{a}_j$ to $\hat{a}_{j-1}$, $\forall j\ 1 < j < n+1$, and by forming a simple internal directed path p' from $\hat{a}_1$ to $\hat{a}_n$, where $a_1, \cdots, a_n$ is a sequence of labels of some class **c** such that all labels in the sequence are distinct, except possibly a_1 and a_n.
- There exists a label d of class **c**' $<_\tau$ **c** and a word w of r_d with $w = w_1 a_1 w_2 \cdots w_n a_n w_{n+1}$, where $w_1, \cdots, w_{n+1}$ are arbitrary strings over Σ.
- Either $w_1 a_1 \mathcal{L}_\tau(p') a_n w_{n+1} \nsubseteq r_d$ or for all internal directed paths p_{1n} in G_τ from $\hat{a_1}$ to $\hat{a_n}$, $a_1 w_2 a_2 \cdots w_n a_n \notin a_1 \mathcal{L}_\tau(p_{1n}) a_n$

We denote by DCycles(τ) the set of dangerous cycles of τ.

Continuation of Example 1. In this example the set MCycles(τ) contains the monochromatic cycles for d and c: $d_\infty \hat{a} \hat{d} d_\infty$, $d_\infty \hat{e} \hat{d} d_\infty$, $d_\infty \hat{e} \hat{b} d_\infty$, $d_\infty d_\infty$, $c_\infty \hat{e} \hat{d} \hat{c} c_\infty$, and $c_\infty c_\infty$. The graph also contains the dangerous cycle $d_\infty \hat{a} \hat{d} \hat{c} c_\infty \hat{e} \hat{b} d_\infty$. This cycle is dangerous because: 1) $r \rightarrow abc$, $\hat{b} d_\infty \hat{a}$ and $\hat{c} c_\infty \hat{e} \hat{b}$ are dangerous, and 2) $adc \notin r_r$.

Continuation of Example 2. In this example some of the monochromatic cycles in the set MCycles(τ) are: $c_\infty \hat{a} \hat{f} \hat{g} c_\infty$, $d_\infty \hat{b} \hat{f} \hat{h} d_\infty$ and $q_\infty \hat{s} \hat{d} \hat{c} \hat{h} q_\infty$. The graph also contains

the dangerous cycle $c_\infty \hat{a}\hat{d}\hat{c}\hat{h}d_\infty \hat{b}\hat{q}\hat{g}c_\infty$. This cycle is dangerous because: 1) $r \to abc$, $\hat{b}\hat{q}\hat{g}c_\infty\hat{a}$ and $\hat{c}\hat{h}d_\infty\hat{b}$ are dangerous, and 2) $adc \notin r_r$.

Continuation of Example 3. In this example some of the the monochromatic cycles in the set MCycles(τ) are: $s_\infty\hat{a}\hat{x}s_\infty$ $v_\infty\hat{a}\hat{d}\hat{w}v_\infty$ $u_\infty\hat{f}\hat{m}u_\infty$.

The graph has only one dangerous cycle $s_\infty\hat{a}\hat{d}\hat{e}\hat{c}t_\infty\hat{b}s_\infty$. This cycle is dangerous because 1) $r \to abc$, $\hat{b}s_\infty\hat{a}$ is dangerous, $\hat{c}t_\infty\hat{b}$ is also dangerous, and 2) $adec \notin r_r$.

Groups versus graphs. Let H be a finite group, G be a directed graph and μ be a mapping from the set of edges of G to H. This induces a mapping, which we also denote by μ, between sequences of edges of G into H such that $\mu(e_1e_2) = \mu(e_1)\cdot\mu(e_2)$ where $\cdot$ is the group operation of H. In particular the mapping μ induces a homomorphism between the directed paths of G to H: If $p = v_1\cdots v_n$ is a directed path of G then $\mu(p) = \mu((v_1,v_2))\mu((v_2,v_3))\cdots\mu((v_{n-1},v_n))$.

Given a DTD τ, a separating group for τ is a finite group H together with a mapping μ from G_τ to H such that $\forall p \in$ MCycles(τ), $\mu(p) = 1$ and $\forall p \in$ DCycles(τ), $\mu(p) \neq 1$ where 1 is the neutral element of H.

Continuation of Example 1. In this example there is no separating group for τ. Indeed assume there is a finite group H and a mapping μ from G_τ to H such that $\forall p \in$ MCycles(τ), $\mu(p) = 1$. Let x bet the edge $(\hat{e},\hat{d})$. One label of x is d and by hypothesis all monochromatic cycles of d are mapped to 1 by μ. Simple algebraic computation shows that then $\mu(x) = \mu(p_d)$ where p_d is the path $(\hat{e},\hat{b},d_\infty,\hat{a},\hat{d})$. Now c is also a label of x and similar algebraic computation shows that $\mu(x)\mu(p_c) = 1$ where p_c is the path $(\hat{d},\hat{c},c_\infty,\hat{e})$. This implies that $\mu(p) = 1$ where p is the path $p_d\cdot p_c$. But p is exactly the dangerous cycle of τ! Therefore any mapping μ sending all monochromatic cycles to the identity of H will also send a dangerous cycle to 1.

Continuation of Example 2. In this example there is a separating group for τ. Let H be the group of order 3 generated by one element: $H = \{1, x, x^2\}$ with $x^3 = 1$. Let μ be the mapping sending all edges to 1 except for the q-labeled edges $e_1 = (\hat{d},\hat{c})$, $e_2 = (\hat{q},\hat{g})$, $e_3 = (q_\infty,\hat{s})$. For those three edges we set $\mu(e_1) = \mu(e_2) = x$ and $\mu(e_3) = x^2 = x^{-1}$. Now one can verify that we do have $\mu(p) = 1$ for all $p \in$ MCycles(τ) (this is trivial for all monochromatic cycles of label different than q and can be done by hand for the others) but that $\mu(p) = x^2 \neq 1$ for the dangerous cycle p. Moreover, one can verify that the group H together with the mapping μ also separates all the other dangerous cycles of the graph from the monochromatic cycles. In particular it can be checked that for all dangerous cycles θ of the graph other than p, $\mu(\theta) = x \neq 1$.

Continuation of Example 3. In this example also there is a separating group for τ. Let H be any finite non-commutative group with α and β two elements of H which do no commute ($\alpha\beta \neq \beta\alpha$ or equivalently $\alpha^{-1}\beta^{-1}\alpha\beta \neq 1$). Let μ be the mapping that sends all edges to 1 except for $\mu((\hat{j},\hat{m})) = \mu((\hat{d},\hat{e})) = \mu((\hat{y},z_\infty)) = \mu((u_\infty,\hat{f})) = \mu((v_\infty,\hat{a})) = \alpha$, $\mu((\hat{f},\hat{g})) = \mu((\hat{c},t_\infty)) = \mu((\hat{w},v_\infty)) = \beta$, $\mu((\hat{x},\hat{y})) = \mu((z_\infty,\hat{j})) = \mu((\hat{e},u_\infty)) = \mu((\hat{m},u_\infty)) = \alpha^{-1}$, $\mu((v_\infty,\hat{f})) = \mu((t_\infty,\hat{f})) = \beta^{-1}$, and $\mu((\hat{a},\hat{d})) = \alpha^{-1}\beta^{-1}$. One can verify that we do have $\forall p \in$ MCycles(τ), $\mu(p) = 1$ but that for the dangerous cycle p we have $\mu(p) = \alpha^{-1}\beta^{-1}\alpha\beta \neq 1$.

The notions of monochromatic and dangerous cycles are motivated by the following result showing a sufficient condition for a DTD to be streamable. It is possible that this condition is also necessary, see Section 6.

Theorem 2. *If a DTD τ satisfies $(*)$ and has a separating group, then τ is streamable.*

Theorem 2 follows from Theorem 4 below that shows how to construct, from a separating group for τ, a FSA that recognizes τ. Note that, although Theorem 4 is constructive, we do not know yet how to decide the existence of such a separating group nor whether we can construct it if it exists. In Section 5 we will construct such a separating group for a special case of DTDs and in Section 6 we will indicate the difficulty of testing the existence of a separating group.

4 Groups and automata

Let τ be a DTD verifying $(*)$. Let H be a finite group and μ be a mapping from edges of G_τ to H. From τ, H and μ we construct an automaton $A(\tau, H, \mu)$, called the *H-local-automaton* for τ which combines local tests on two consecutive symbols with operations in H. If H is a separating group for τ we show that $A(\tau, H, \mu)$ recognizes τ.

Let 1 be the neutral element of H and $\cdot$ be its group operation. Consider again the partition C_τ of Σ and its preorder $\leq_\tau$. For $d \in \Sigma$, recall the definition of $\mathcal{L}_\tau(d)$ given in Section 3.

For each class $\mathbf{c} \in C_\tau$ and $d \in \mathbf{c}$, we define an automaton $A_\mu(\mathbf{c}, d)$ by induction on $\leq_\tau$ as follows. The intuition is that $A_\mu(\mathbf{c}, d)$ is only concerned with symbols in classes higher or equal to $\mathbf{c}$ relative to $\leq_\tau$ and that it checks locally consistency with τ while simulating the product in H for successive pairs of symbols in $\mathbf{c}$: For each sequence of two successive symbols in $\mathbf{c}$ it checks whether this sequence is plausible in τ (by inspecting $G_\tau(\mathbf{c})$) and, if this is the case, simulates the product in H using this pair and μ. When a symbol in a higher class is read, local consistency with τ and the previous symbol read is checked and a subcomputation for the new class is started. Each subcomputation should start simulating the product in H at its neutral element 1 and ends only when the current value of this product is 1. In summary only local tests are performed against the DTD except for the product in H which is the only information which is carried over the tree.

Checking that any sequence of two successive symbols in $\mathbf{c}$ is plausible in τ can be read from $G_\tau(\mathbf{c})$: It amounts to check that, for any $a, b \in \mathbf{c}$, if $\bar{a}b$ occurs then $(\hat{a}, \hat{b}) \in G_\tau(\mathbf{c})$, if ab occurs then $(a_\infty, \hat{b}) \in G_\tau(\mathbf{c})$, if $a\bar{b}$ occurs then $a = b$ and $(a_\infty, a_\infty) \in G_\tau(\mathbf{c})$, and, if $\bar{a}\bar{b}$ occurs then $(\hat{a}, b_\infty) \in G_\tau(\mathbf{c})$.

The simulation of the product in H is done as follows. For each class $\mathbf{c} \in C_\tau$ and $d \in \mathbf{c}$, let $A_H(\mathbf{c}, d)$ be the automaton simulating the product in H for edges in $G_\tau(\mathbf{c})$ while ignoring the symbols not related to $\mathbf{c}$. It is defined formally as follows. Its states are elements of $(\Sigma \cup \bar{\Sigma}) \times H$. Its initial state is $(d, 1)$. When reading a symbol $\delta \in (\Sigma \cup \bar{\Sigma})$, it has a transition from (α, h) to (β, h') exactly when one of the condition below is satisfied.

– $\delta \in \mathbf{c}$, $\beta = \delta$, $\alpha = \bar{y}$ for $y \in \mathbf{c}$, $(\hat{y}, \hat{\delta})$ is an edge e of G_τ, and $h \cdot \mu(e) = h'$.
– $\delta \in \mathbf{c}$, $\beta = \delta$, $\alpha \in \mathbf{c}$, $(\alpha_\infty, \hat{\delta})$ is an edge e of G_τ, and $h \cdot \mu(e) = h'$.
– $\delta \in \bar{\mathbf{c}}$, $\beta = \delta$, $\alpha \in \mathbf{c}$, $\delta = \bar{\alpha}$, $(\alpha_\infty, \alpha_\infty)$ is an edge e of G_τ, and $h \cdot \mu(e) = h'$.
– $\delta \in \bar{\mathbf{c}}$, $\beta = \delta$, $\alpha \in \bar{\mathbf{c}}$, $(\hat{y'}, x'_\infty)$ is an edge e of G_τ where x' and y' are such that $\delta = \bar{x'}$ and $\alpha = \bar{y'}$, and $h \cdot \mu(e) = h'$.
– If $\delta \notin (\mathbf{c} \cup \bar{\mathbf{c}})$, $\alpha = \beta$ and $h = h'$ (those letters are ignored).

The set of final states of $A_H(\mathbf{c}, d)$ are all the states $(\bar{a}, h)$, $a \in \mathbf{c}$, such that $(\hat{a}, d_\infty)$ is an edge of $G_\tau(\mathbf{c})$ such that $h \cdot \mu((\hat{a}, d_\infty)) = 1$, together with all the states (d, h), such that (d_∞, d_∞) is an edge of $G_\tau(\mathbf{c})$ such that $h \cdot \mu((d_\infty, d_\infty)) = 1$.

It now remains to perform the local tests on how the class can interleave. This is also read from G_τ.

Let $\mathbf{c}$ be a maximal class of C_τ and let $d \in \mathbf{c}$. This is the simple case. Because the class is maximal, there is no interleaving authorized and $A_\mu(\mathbf{c}, d)$ needs only to simulate the product in H. In this case we let $A_\mu(\mathbf{c}, d) = A_H(\mathbf{c}, d)$.

Let now $\mathbf{c}$ be an arbitrary class and $d \in \mathbf{c}$. Assuming the definition of $A_\mu(\mathbf{c}', d')$ for each class $\mathbf{c}$' and element $d' \in \mathbf{c}'$ such that $\mathbf{c} <_\tau \mathbf{c}$', we define $A_\mu(\mathbf{c}, d)$. In this case we have to worry about symbols in higher classes and check for local consistency with τ. This is done as follows. We define next an automata $A_\tau(\mathbf{c}, d)$ that does this local consistency tests then we set $A_\mu(\mathbf{c}, d)$ as the product of $A_\tau(\mathbf{c}, d)$ with $A_H(\mathbf{c}, d)$. For each edge e of $G_\tau(\mathbf{c})$, recall the definition of $\mathcal{L}(e)$ as given in Section 3. For each edge e of $G_\tau(\mathbf{c})$, let A_e be the deterministic minimal automaton for $\mathcal{L}(e)$ and assume that these automata have pairwise disjoint sets of states. We build on these automata to construct $A_\tau(\mathbf{c}, d)$. $A_\tau(\mathbf{c}, d)$ contains all the states of the A_e together with one state q_x per symbol $x \in \Sigma \cup \bar{\Sigma}$. Let $e = (\alpha, \beta)$ be an edge of $G_\tau(\mathbf{c})$, let q_0^e be the initial state of A_e and F^e be its set of accepting states. For any transition (q, d', q') of A_e, where $d' \in \mathbf{c}'$, we add in $A_\tau(\mathbf{c}, d)$ a fresh new copy of $A_\mu(\mathbf{c}', d')$ with initial state q_0 and accepting set of states F, and we add in $A_\tau(\mathbf{c}, d)$ the transitions (q, d', q_0) and $(q_f, \bar{d'}, q')$ for any $q_f \in F$. Depending on α and β we also add in $A_\tau(\mathbf{c}, d)$ the following transitions (with $a = l(\alpha)$ and $b = l(\beta)$).

– If α and β are inner nodes, we add transitions $q_{\bar{a}} \xrightarrow{\epsilon} q_0^e$ and $q_f^e \xrightarrow{b} q_b$ for any $q_f^e \in F^e$.
– If α is an inner node but β is not, we add transitions $q_{\bar{a}} \xrightarrow{\epsilon} q_0^e$ and $q_f^e \xrightarrow{\bar{b}} q_{\bar{b}}$ for any $q_f^e \in F^e$.
– If β is an inner node but α is not, we add transitions $q_a \xrightarrow{\epsilon} q_0^e$ and $q_f^e \xrightarrow{b} q_b$ for any $q_f^e \in F^e$.
– If both α and β are not inner nodes (then $\alpha = \beta$), we add transitions $q_a \xrightarrow{\epsilon} q_0^e$ and $q_f^e \xrightarrow{\bar{b}} q_{\bar{b}}$ for any $q_f^e \in F^e$.

The initial state of $A_\tau(\mathbf{c}, d)$ is the state q_d. The set of accepting states of $A_\tau(\mathbf{c}, d)$ are all the accepting states of the automata A_e where e is an edge ending in d_∞. This finishes the construction of $A_\tau(\mathbf{c}, d)$ and therefore of $A_\mu(\mathbf{c}, d)$.

Now set $A(\tau, H, \mu)$ as $A_\mu(\mathbf{c}, r)$ where r is the initial symbol of the DTD and $\mathbf{c}$ is the class of r. When H is the trivial group with one element, the $A(\tau, H, \mu)$ is exactly

what was call "standard automaton" in [4]. We are now ready to state the main result of this paper.

Theorem 3. *Assume that τ satisfies $(*)$. There exists a separating group for τ iff there exists a finite group H and a mapping μ such that $A(\tau, H, \mu)$ recognizes τ.*

The proof of Theorem 3 is given in two steps. Theorem 4 below shows that if H is a separating group for τ then $A(\tau, H, \mu)$ recognizes τ. Next, Theorem 5 shows how to compute a separating group from a H-local-automaton recognizing τ.

Theorem 4. *If τ verifies $(*)$ and there exists a separating group H for τ via the mapping μ, then $A(\tau, H, \mu)$ recognizes τ.*

Proof. The proof of this theorem is very technical and will appear in the full version of this paper. We only outline it here. One first shows that if H and μ are such that for each $p \in \text{MCycles}(\tau)$ $\mu(p) = 1$, then all trees valid with respect to τ are accepted by $A(\tau, H, \mu)$. This is done by induction on $\leq_\tau$ by noticing that in a valid tree, any sequence of labels of the children of a node induces a monochromatic cycle in G_τ.

The other direction, showing that $A(\tau, H, \mu)$ accepts only valid trees is more complicated and requires that H is a separating group and that τ verifies $(*)$. The proof is again done by induction on $\leq_\tau$. We only illustrate here the requirement on dangerous cycles on an example. Assume the DTD has initial symbol r with the rule $r \to abc$ and that a, b, c are symbols in the same class **c** and that no other classes are in τ . By construction $A(\tau, H, \mu)$ performs three successive "calls", to $A_\mu(\mathbf{c}, a)$, $A_\mu(\mathbf{c}, b)$, then $A_\mu(\mathbf{c}, c)$. Assume we have shown by induction that all trees accepted by $A_\mu(\mathbf{c}, a_i)$ are valid for τ, we show that this is the case for $A_\mu([r], r)$. Let t be a tree accepted by $A(\tau, H, \mu)$. We decompose the string t into $s_1 s_2 s_3$ where s_i is the substring read by $A_\mu(\mathbf{c}, a_i)$, where $a_1 = a, a_2 = b, a_3 = c$. Assume moreover that those substrings are as depicted in Figure 1.

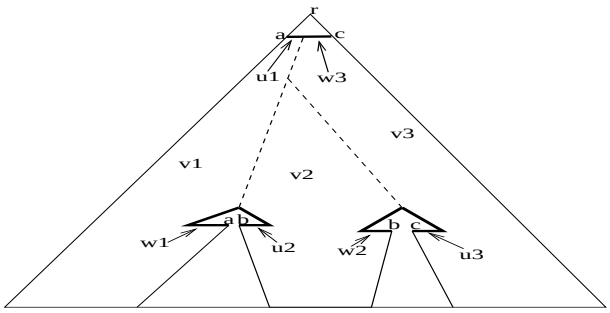


Fig. 1. Illustration of the run of $A(\tau, H, \mu)$. We have $s_i = u_i v_i w_i$, where u_i is the substring containing all the trees in s_i whose root is either the first symbol of s_i or one of its siblings, w_i is the substring containing all the trees in s_i whose root is either the last symbol of s_i or one of its siblings and v_i is the remaining part of s_i.

We further decompose s_i into $u_i v_i w_i$ as depicted in Figure 1. Let ν_i be the sequence of tree nodes consisting of the roots of the trees in u_i followed by the first node processed in v_i. Similarly let ω_i be the sequence of tree nodes consisting of the last node processed in v_i followed by the roots of the trees in w_i. As each of s_i is read by $A_\mu(\mathbf{c}, a_i)$, for all i, ν_i and ω_i correspond to paths in $G_\tau(\mathbf{c})$, which we denote as α_i and β_i, respectively. This implies that the path p formed by the path concatenation $\beta_3 \cdot \alpha_3 \cdot \beta_2 \cdot \alpha_2 \cdot \beta_1 \cdot \alpha_1$ is a cycle in τ.

By abuse of notations we denote by $\mu(s)$ the image by μ of the path induced by the sequence of symbols of s. We know that $\mu(s_1) = \mu(s_2) = \mu(s_3) = 1$. This implies – by denoting as v_i' and v_i'' the first and the last symbol of v_i, respectively – that $\mu(v_i) = \mu(u_i v_i')^{-1} \mu(v_i'' w_i)^{-1}$. By induction we know that any tree t' entirely contained in a s_i is valid for τ and therefore is such that $\mu(t') = 1$. This implies that $\mu(u_i v_i') = \mu(\alpha_i)$ and $\mu(v_i'' w_i) = \mu(\beta_i)$. Therefore $\mu(v_1)\mu(v_2)\mu(v_3) = \mu(p)^{-1}$. Now notice that there exist valid forests f_1, f_2 such that $v_1 f_1 v_2 f_2 v_3$ is a valid tree rooted in a label of class $\mathbf{c}$, thus $\mu(v_1 f_1 v_2 f_2 v_3) = 1$. This implies $\mu(v_1)\mu(v_2)\mu(v_3) = 1$. As H was a separating group for τ, this shows that p cannot be a dangerous cycle. Therefore (a, b) and (b, c) must be edges of $G_\tau(\mathbf{c})$ and the sequence of labels in the path $\alpha_1 \beta_3$ must be valid below the root. This implies that t is valid for τ. The general case requires more case analysis but the overall idea is the same. □

Theorem 5. *Assume H and μ are such that $A(\tau, H, \mu)$ recognizes τ. Then there exists a finite group H', constructible from τ, H and μ, such that H' is a separating group for τ. Moreover if H is commutative then H' is also commutative.*

Proof. The proof will appear in the full version of this paper. □

5 Commutative Separating Groups

In this section we provide a necessary and sufficient decidable condition for the existence of a commutative separating group for a DTD. Note that by Theorem 5 this implies a necessary and sufficient decidable condition for the existence of a H-local-automaton, H commutative, that recognizes a given DTD.

Throughout this section we consider a DTD τ satisfying the condition $(*)$. Let X_τ be the set of edges of G_τ and m the cardinality of X_τ. Let n be the number of cycles in MCycles(τ). Let $X_\tau = \{x_1, \ldots, x_m\}$ and MCycles(τ) $= \{\pi_1, \ldots, \pi_n\}$.
We first fix some more notations on linear algebra. For any $r \in \mathbb{N}$, let $\mathbb{Z}^r$ denote the set of vectors of integers of size r, and $\mathbb{N}^r$ denote the set of vectors of non-negative integers of size r. For any $\bar{y} \in \mathbb{Z}^r$, the i-th component of $\bar{y}$ will be denoted as $\bar{y}[i]$.The maximum absolute value occurring in $\bar{y}$ is denoted by $\max \bar{y}$.

For each $k \in \mathbb{N}$, and each $y, z \in \mathbb{N}$, $y \equiv_k z$ denotes that y and z agree modulo k and $[y]_k$ is the number between 0 and $k-1$ equivalent to y modulo k, and $[\bar{y}]_k$ denotes its extension to $\mathbb{N}^r$. Moreover, for each $k \in \mathbb{N}$, the vector $\bar{k}^r$ is the vector of $\mathbb{N}^r$ which has all its components set to k. For each $i \leq r$, the vector ϵ_i^r denotes the vector of $\mathbb{N}^r$ having 1 on the i-th component, and 0 everywhere else.
For each path π of G_τ and for each edge $x_i \in X_\tau$ we denote as $|\pi|_{x_i}$ the number of times that π traverses x_i according to the edge orientation, and by $|\pi|_{x_i^-}$ the number

of times that π traverses x_i in reverse direction. To each path π of G_τ, we associate a vector $\bar{\pi}$ of $\mathbb{Z}^m$ such that, for each $i = 1, \ldots, m$, $\bar{\pi}[i] = |\pi|_{x_i} - |\pi|_{x_i^-}$. Notice that if π is a directed path, $\bar{\pi}$ is a vector of $\mathbb{N}^m$, and if π is a simple path, then $\bar{\pi} \in \{-1, 0, 1\}$.

We denote as $\overline{\text{MCycles}(\tau)}$ the subgroup of $\mathbb{Z}^m$ generated by $\{\bar{\pi}_1, \ldots, \bar{\pi}_n\}$, that is $\overline{\text{MCycles}(\tau)} = \{\bar{y} \in \mathbb{Z}^m \mid \bar{y} = \sum_{i=1}^n \alpha_i \cdot \bar{\pi}_i,\ \ \alpha_i \in \mathbb{Z},\ \ i = 1, .., n\}$.

All this notation is motivated by the following result:

Theorem 6. *Given a DTD τ satisfying $(*)$, there exists a commutative separating group for τ if and only if $\forall p \in DCycles(\tau)$, $\bar{p} \notin \overline{MCycles(\tau)}$.*

Proof. We first prove the "only if" part. Let $p \in \text{DCycles}(\tau)$ be such that $\bar{p} = \sum_{i=1}^n \alpha_i \cdot \bar{\pi}_i$, $\alpha_i \in \mathbb{Z}$ for $i = 1, .., n$.
Let D be the set of indices $i \in 1..n$ such that $\alpha_i \geq 0$, and R be the set of indices $i \in 1..n$ such that $\alpha_i < 0$, and let γ_i be the absolute value of α_i, for each $i = 1, .., n$. Then the following equality holds: $\bar{p} + \sum_{i \in R} \gamma_i \cdot \bar{\pi}_i = \sum_{i \in D} \gamma_i \cdot \bar{\pi}_i$

Suppose, by contradiction, that there exists a commutative separating group H with associated mapping μ from G_τ to H.
By commutativity of H, $\mu(p) = \prod_{j=1}^m \mu(x_j)^{\bar{p}[j]}$ and $\mu(\pi_i) = \prod_{j=1}^m \mu(x_j)^{\bar{\pi}_i[j]}$ for each $i = 1, .., n$. Let h_R be the element of H obtained as $h_R = \prod_{i \in R} \mu(\pi_i)^{\gamma_i}$; by commutativity of H, $h_R = \prod_{j=1}^m \mu(x_j)^{k_j}$, where $k_j = \sum_{i \in R} \gamma_i \cdot \bar{\pi}_i[j]$, for $j = 1, .., m$.
Similarly, Let h_D be the element of H obtained as $h_D = \prod_{i \in D} \mu(\pi_i)^{\gamma_i}$; by commutativity of H, $h_D = \prod_{j=1}^m \mu(x_j)^{k'_j}$, where $k'_j = \sum_{i \in D} \gamma_i \cdot \bar{\pi}_i[j]$, for $j = 1, .., m$.

Together with $\bar{p} + \sum_{i \in R} \gamma_i \cdot \bar{\pi}_i = \sum_{i \in D} \gamma_i \cdot \bar{\pi}_i$ this immediately implies that $\mu(p) = h_D \cdot h_R^{-1}$. As H is a separating group for τ, we have $h_R = h_D = 1$ therefore $\mu(p) = 1$ which is a contradiction.

We will now prove the "if" part. For each $p \in \text{DCycles}(\tau)$ we show that there exists a finite commutative group H_p and a homomorphism μ_p from G_τ to H_p, such that $\mu_p(p) \neq 1$ and $\mu_p(\pi_i) = 1$ for $i = 1, \ldots, n$. The desired commutative separating group for τ is the product of all H_p, for $p \in \text{DCycles}(\tau)$.

Given $p \in \text{DCycles}(\tau)$ such that $p \notin \overline{\text{MCycles}(\tau)}$ we effectively construct a separating commutative group for p.
For each edge x_j of X_τ, let D_j be the set of cycles of MCycles(τ) which traverse the edge x_j: $D_j = \{i \in 1..n \mid \bar{\pi}_i[j] = 1\}$.

For a vector $\bar{y} = (y_1, \ldots, y_m)$ over $\mathbb{N}^m$, we define the following formula of Presburger arithmetic:

$$\psi(\bar{y}) = \exists \alpha_1, \ldots, \alpha_n \bigwedge_{j=1}^m \left(y_j = \sum_{i \in D_j} \alpha_i \right)$$

By construction, ψ is satisfied by the vector associated to each cycle of MCycles(τ) and is not satisfied by $\bar{p}$; We will refer to this property saying that ψ is a separating formula for $\bar{p}$.

By Presburger quantifier elimination procedure, ψ can be transformed into an equivalent quantifier free formula $\varphi(\bar{y}) := \varphi_e(\bar{y}) \wedge \varphi_c(\bar{y})$, where, using matrix notation, φ_e and φ_c are formulas of the form:

$$\varphi_e(\bar{y}) := (A\bar{y} = B\bar{y}) \qquad \varphi_c(\bar{y}) := (C\bar{y} \equiv_\delta D\bar{y})$$

where $A, B \in \mathbb{N}^{m_e \times m}$ and $C, D \in \mathbb{N}^{m_c \times m}$, for some $m_e, m_c \in \mathbb{N}$.

From φ we define a new separating formula for $\bar{p}$, $\varphi_k(\bar{y})$, whose terms use only the modulo-congruence operator $\equiv_k$, for some $k \in \mathbb{N}$. We consider two cases:

1. $\varphi_c(\bar{p})$ is false. Then $\varphi_k(\bar{y}) := \varphi_c(\bar{y})$ and $k = \delta$.
2. $\varphi_e(\bar{p})$ is false. We choose $k = \max(A\bar{p} - B\bar{p}) + 1$ and take $\varphi_k(\bar{y}) := (A\bar{y} \equiv_k B\bar{y})$.

In both cases we have $\varphi_k(\bar{y}) := (E\bar{y} \equiv_k F\bar{y})$, where E, $F \in \mathbb{N}^{r \times m}$ for some $r \in \mathbb{N}$, and the vector of p does not satisfy the formula while it is satisfied by any cycles in MCycles(τ). We now construct, from $\varphi_k(\bar{y})$, a commutative finite group H_p and a homomorphism μ_p from the paths of G_τ to H_p, such that $\mu_p(p) \neq 1$ and $\mu_p(\pi_i) = 1$ for all $i = 1 \dots n$.

Let K be the commutative finite group whose elements are the pairs $(\bar{z}_1, \bar{z}_2) \in \mathbb{N}^r \times \mathbb{N}^r$ such that $\bar{z_1} < \bar{k}^r \wedge \ \bar{z_2} < \bar{k}^r$, with group composition defined by $(\bar{z}_1, \bar{z}_2) + (\bar{z}_3, \bar{z}_4) = ([\bar{z}_1 + \bar{z}_3]_k, [\bar{z}_2 + \bar{z}_4]_k)$. We define a mapping μ_K from the set of edges of G_τ to K as follows: for each edge $x_i \in X_\tau$, $\mu_K(x_i) = ([E\epsilon_i^m]_k, [F\epsilon_i^m]_k)$.
Let $K_=$ be the (normal) subgroup of K, consisting of all elements of the form $(\bar{z}, \bar{z})$. Let $H_p = K/K_=$, and set μ_p as the composition of μ_K with the quotient morphism. As $[E\bar{p}]_k \neq [F\bar{p}]_k$, $\mu_K(p) = ([E\bar{p}]_k, [F\bar{p}]_k)$ is not in $K_=$ thus, $\mu_p(p) \neq 1$.

Conversely if π is a cycle of MCycles(τ), $[E\bar{\pi}]_k = [F\bar{\pi}]_k$, thus $\mu_p(\pi) = 1$. This concludes the proof of the Theorem. □

An immediate corollary of Theorem 6 and of the constructiveness of its proof together with Theorem 4, is the following:

Corollary 1. *Given a DTD τ, it is decidable whether there exists a commutative separating group for τ, and if it exists, the group, and therefore a H-local-automaton recognizing τ, can be effectively computed.*

Continuation of Example 2. In this example a commutative separating group exists. We have already given an example of such a group. This group could also be obtained from Theorem 6. Indeed one can verify that the dangerous cycle $p = b_\infty \hat{q}\hat{g}c_\infty \hat{a}\hat{d}\hat{c}\hat{h}d_\infty \hat{b}$ is not a linear combination of monochromatic cycles as defined in the statement of Theorem 6.

Continuation of Example 3. This example shows that there exist DTDs for which no commutative separating group exists, but a separating group does exist. Indeed, the dangerous cycle $p = s_\infty \hat{a}\hat{d}\hat{e}\hat{c}t_\infty \hat{b}s_\infty$ for that DTD τ can be shown to be such that $\bar{p} \in \overline{\text{MCycles}(\tau)}$ as follows. Let θ_1, θ_2, θ_3, θ_4, θ_5 be the following non-directed cycles: $\theta_1 = \hat{d}\hat{n}\hat{x}\hat{y}\hat{j}\hat{m}\hat{f}\hat{w}\hat{d}$, $\theta_2 = \hat{d}\hat{e}\hat{f}\hat{m}\hat{j}\hat{h}\hat{b}\hat{q}\hat{x}\hat{n}\hat{d}$, $\theta_3 = \hat{d}\hat{w}\hat{f}\hat{g}\hat{j}\hat{y}\hat{x}\hat{a}\hat{d}$, $\theta_4 = \hat{f}\hat{e}\hat{c}t_\infty \hat{b}\hat{h}\hat{j}\hat{g}\hat{f}$, $\theta_5 = \hat{x}\hat{q}\hat{b}s_\infty \hat{a}\hat{x}$; it's easy to check that $\bar{\theta_1}$ is a linear sum (with weights 1 and -1) of monochromatic cycles whose labels are all in z, the same holds for $\bar{\theta_2}$,$\bar{\theta_3}$, $\bar{\theta_4}$, $\bar{\theta_5}$, using monochromatic cycles of label u, v, t and s, respectively.
As we alos have $\bar{p} = \sum_{i=1..5} \bar{\theta_i}$, $\bar{p} \in \overline{\text{MCycles}(\tau)}$, thus no commutative separating group exists for that DTD.

Unfortunately, we are not yet able to extend Theorem 6 to non-commutative groups. See Section 6 for more details on this issue.

6 Discussion and conclusion

The next step is obviously to prove whether it is decidable that a DTD has a separating group or not. We have seen that this is related to the word problem for finite groups.

The word problem for finite groups is whether, given a finite set F of words and a word w over a finite alphabet of the form $A \cup A^-$, there exists a finite group H and a morphism $\mu : (A \cup A^-)^* \to H$ interpreting words in F as the identity of H but such that $\mu(w) \neq 1_H$. We are interested in the case where $F = \text{MCycles}(\tau)$ and $w \in \text{DCycles}(\tau)$. This problem is undecidable in general [5]. But we are dealing with a very special case of the word problem as MCycles(τ) and DCycles(τ) have a lot of similarities being both defined via the same graph. It is thus quite possible that this special case is decidable. Note that we are not only interested in knowing whether such a separating group exists but also in constructing it.

It would also be interesting to know whether the notion of streamability coincides with the existence of a separating group for τ. We think that this might be the case and would like to argue here in favor of this conjecture. An obvious extension of the H-local-automaton would be to allow computation in an arbitrary monoid (instead of a group). We don't think that this will extend the expressive power. Indeed assume that a DTD τ is recognized by a H-local-automaton A where H is now just a monoid. Because we are dealing with DTDs, monoid computation of A on any valid subtrees should correspond to the identity. This is because neither the context nor the content of the subtree matters for the rest of the validation, therefore the automaton does not need to remember anything (besides the fact that the subtree is valid or not). Now, because of the local tests against the DTD, each time A has partially read a subtree and has not yet rejected it, there is always a way to complete it so that it is valid for the DTD. In other words, for any monoid element m (the current monoid state of A in the subtree), there is always a m' such that $m \cdot m' = 1$: The monoid needs to be right-invertible. The reason why we took groups instead of right-invertible monoids is more technical and might not be necessary. It is because we deal with cycles (monochromatic and dangerous) and therefore work modulo cyclic permutations. This implies that if $m \cdot m' = 1$ then the cyclic permutation $m' \cdot m$ of $m \cdot m'$ should also be 1. Thus the monoid is a group.

Finally it would be interesting to go beyond the class of DTDs as defined in this paper, and start with an arbitrary regular language. This would answer an open question raised in [4, 1] and certainly requires more ideas than those presented here.

Acknowledgment. We thank Victor Vianu for all the interesting discussions we had with him on this subject.

References

1. V. Bárány and C. Löding and O. Serre. Regularity Problems for Visibly Pushdown Languages. In *STACS*, 2006.
2. C. Chitic and D. Rosu. On Validation of XML Streams Using Finite State Machines. In *WebDB*, 2004.
3. P.S. Novikov. On the algorithmic unsolvability of the word problem in group theory. In *Trudy Mat. Inst. Steklov*, 44, 1955. (in Russian)
4. L. Segoufin and V. Vianu. Streaming XML documents. In *PODS*, 2002.
5. A.M. Slobodskoi. Unsolvability of the universal theory of finite groups. In *Algebra and Logic*, 20, pp 139-156, 1981.