Project Description - Tree Automata vs Tree Decompositions

Mateus de Oliveira Oliveira[1]

Department of Informatics, University of Bergen, Norway
mateus.oliveira@uib.no

1 Tree Automata

For each $n \in \mathbb{N}$, we let $[n] = \{1, ..., n\}$. As a degenerate case, we define $[0] = \emptyset$. For each finite set S we let $\mathcal{P}(S) = \{S' \ : \ S' \subseteq S\}$ denote the set of all subsets of S.

Prefix Closed Sets. Let $[r]^*$ denote the set of all strings over $[r]$ and let λ denote the empty string. A subset $U \subseteq [r]^*$ is *prefix closed* if for every $p \in [r]^*$ and every $j \in [r]$, $pj \in U$ implies that $p \in U$. We note that the empty string λ is an element of any prefix closed subset of $[r]^*$. We say that $U \subseteq [r]^*$ is *well numbered* if for every $p \in [r]^*$ and every $j \in [r]$, the presence of pj in U implies that $p1, ..., p(j-1)$ also belong to U.

Tree-Like Sets. We say that a subset $U \subseteq [r]^*$ is *tree-like* if U is both prefix-closed and well-numbered. Let U be a tree-like subset of $[r]^*$. If $pj \in U$, then we say that pj is a *child* of p, or interchangeably, that p is the *parent* of pj. If $pu \in U$ for $u \in [r]^*$, then we say that pu is a descendant of p. For a node $p \in U$ we let $U|_p = \{pu \in U \ : \ u \in [r]^*\}$ denote the set of all descendants of p. Note that p is a descendant of itself and therefore $p \in U|_p$. A *leaf* of U is a node $p \in U$ without children. We let $leaves(U)$ be the set of leaves of U, and $leaves(U, p)$ be the set of leaves which are descendants of p. We say that a subset S of U is *connected* if there exists a unique $p \in S$ with the property that either $p = \lambda$ or $p \neq \lambda$ and the parent of p is not in S.

Terms. Let Σ be a finite set of symbols. An r*-ary term* over Σ is a function $t : Pos(t) \rightarrow \Sigma$ whose domain $Pos(t)$ is a tree-like subset of $[r]^*$.

We denote by $\text{Terms}(\Sigma)$ the set of all terms over Σ. If $t_1, ..., t_r$ are terms in $\text{Terms}(\Sigma)$, and $a \in \Sigma$, then we let $t = a(t_1, ..., t_r)$ be the term in $\text{Terms}(\Sigma)$ which is defined by setting $t(\lambda) = a$ and $t(jp) = t_j(p)$ for each $j \in [r]$ and each $p \in Pos(t_j)$.

Tree Automata: Let Σ be a finite set of symbols. A *bottom-up tree automaton* over Σ is a tuple $\mathcal{A} = (Q, \Sigma, F, \Delta)$ where Q is a set of states, $F \subseteq Q$ a set of final states, and Δ is a set of transitions of the form $(q_1, \ldots, q_r, a, q)$ with $a \in \Sigma$, and $q_1, \ldots, q_r, q \in Q$. The size of $\mathcal{A}$, which is defined as $|\mathcal{A}| = |Q| + |\Delta|$, measures the number of states in Q plus the number of transitions in Δ. The set $\mathcal{L}(\mathcal{A}, q, i)$ of all terms reaching a state $q \in Q$ in depth at most i is inductively defined as follows: If (a, q) is a transition in Δ, then a reaches state q in depth 1. If $(q_1, \ldots, q_k, a, q)$ is a transition in Δ, and $t_1, \ldots, t_k$ are terms in $Ter(\Sigma)$ such that t_j reaches state q_j in depth at most i for each $j \in [k]$, then the term $a(t_1, \ldots, t_k)$ reaches q in depth at most $i+1$. For each state $q \in Q$, we let $\mathcal{L}(\mathcal{A}, q) = \bigcup_{i \in \mathbb{N}} \mathcal{L}(\mathcal{A}, q, i)$ be the set of terms reaching state q in some finite depth. Finally, the language accepted by $\mathcal{A}$ is the set $\mathcal{L}(\mathcal{A}) = \bigcup_{q \in F} \mathcal{L}(\mathcal{A}, q)$ of terms reaching some final state of $\mathcal{A}$. For each transition $\tau = (q_1, \ldots, q_k, a, q) \in \Delta$, we say that τ has *arity* $\mathfrak{a}(\tau) = k$, q is the *consequent* of τ and for each i, q_i is an *antecedent* of τ. We let $\mathfrak{a}(\mathcal{A}) = \max_{\tau \in \Delta} \mathfrak{a}(\tau)$ be the maximum arity of a transition in Δ. For each state $q \in Q$, we let

$$\Delta(q) = \{\tau \in \Delta \; : \; q \text{ is the consequent of } \tau\} \tag{1}$$

be the set of transitions that have q as consequent. We let $\delta(q) = |\Delta(q)|$ be the *in-degree* of q and $\delta(\mathcal{A}) = \max_{q \in Q} \delta(q)$.

Runs: A *run* in $\mathcal{A} = (Q, \Sigma, F, \Delta)$ is a pair (t, ρ) where $t \in Ter(\Sigma)$, and $\rho : Pos(t) \rightarrow Q$ is a function such that for each position $p \in Pos(t)$, if $p1, \ldots, pr$ are the children of p, then the transition $(\rho(p_1), \ldots, \rho(p_r), t(p), \rho(p))$ belongs to Δ. The size of (t, ρ) is defined as $|(t, \rho)| = |Pos(t)|$. We say that (t, ρ) reaches state q if $\rho(\lambda) = q$. We say that (t, ρ) is *accepting* if (t, ρ) reaches some accepting state. We say that (t, ρ) reaches a transition $\tau \equiv (q_1, \ldots, q_r, a, q)$ if the following conditions are satisfied:

1. The root position λ of $Pos(t)$ has r children. Namely, these children are the positions $1, 2, ..., r \in Pos(t)$.
2. $t(\lambda) = a$.
3. $\rho(\lambda) = q$.
4. For each $i \in \{1, \ldots, r\}$, $\rho(i) = q_i$.

We say that $\mathcal{A}$ is *deterministic* if for each tuple of states $q_1, ..., q_r$ and each symbol $a \in \Sigma$, there exists at most one state q such that $(q_1, \ldots, q_r, a, q) \in \Delta$. We say that $\mathcal{A}$ is *unambiguous* if for each term $t \in Ter(\Sigma)$ there exists at most one ρ such that (t, ρ) is a run in $\mathcal{A}$. We note that the notion of unambiguity generalizes the notion of determinism, since any deterministic tree automaton is unambiguous. We say that a run (t', ρ') *extends* a run (t, ρ) if there exists a string $p \in [r]^*$ such that $t(u) = t'(pu)$ and $\rho(u) = \rho'(pu)$ for every position $u \in Pos(t)$.

Reachability and co-reachability: We say that a tree automaton $\mathcal{A} = (Q, \Sigma, \Delta, F)$ is *reachable* if for every state $q \in Q$ there exists a run (t, ρ) which reaches q. We say that $\mathcal{A}$ is co-reachable if each run (t, ρ) in $\mathcal{A}$ can be extended to an accepting run (t', ρ'). In the remainder of this paper, unless stated otherwise, we assume that the automata we deal with are both reachable and co-reachable. This assumption is without loss of generality since any tree automaton $\mathcal{A}$ can be transformed in time $O(|\mathcal{A}|)$ into a reachable and co-reachable tree automaton $\mathcal{A}'$ accepting the same language (i.e., $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathcal{A}')$).

2 Concrete Tree Decompositions

Graphs. A *graph* is a tuple $G = (V(G), E(G), \mathrm{Inc}(G))$ where $V(G)$ is a finite set of vertices, $E(G)$ is a finite set of edges, and $\mathrm{Inc}(G) \subseteq E(G) \times V(G)$ is an incidence relation. We say that a vertex $v \in V(G)$ is an endpoint of an edge $e \in E(G)$ if $(e, v) \in \mathrm{Inc}(G)$. We let $\mathrm{endpts}(e) = \{v : (e, v) \in \mathrm{Inc}(G)\}$ be the set of endpoints of e. In this work we only consider graphs in which each edge has precisely two endpoints. Therefore, self-loops are not allowed. On the other hand, our definition of graphs allow the presence of *multiple-edges*, i.e, distinct edges with the same pair of endpoints. A graph is *simple* if no two distinct edges have the same pair of endpoints.

Concrete Tree Decompositions. A *d-concrete bag* is a pair (B, b) where $B \subseteq [d]$ and $b \subseteq B$, with either $b = \emptyset$ or $|b| = 2$. We note that B is allowed to be empty. We let $\mathcal{B}_d$ denote the set of all d-concrete bags. Note that $|\mathcal{B}_d| = O(2^d \cdot d^2)$. We regard $\mathcal{B}_d$ as a finite alphabet that will be used to construct terms representing tree decompositions of graphs.

A *d-concrete tree decomposition* is a term $t \in \mathrm{Terms}(\mathcal{B}_d)$. We let $t(p) = (t(p).B, t(p).b)$ be the d-concrete bag at position p of t. For each $c \in [d]$, we say that a subset $P \subseteq \mathrm{Pos}(t)$ is a c-colored component if the following conditions are satisfied.

1. P is a connected subset of $\mathrm{Pos}(t)$.
2. $c \in t(p).B$ for every $p \in P$.
3. If P' is a connected subset of $\mathrm{Pos}(t)$, and $c \in t(p).B$ for every $p \in P'$, then $P' \subseteq P$.

Note that if P and P' are c-colored components, then either $P = P'$, or $P \cap P' = \emptyset$. Additionally, for each $p \in \mathrm{Pos}(t)$ and each $c \in t(p).B$, there exists a unique subset $P \subseteq \mathrm{Pos}(t)$ such that P is a c-colored component and $p \in P$. We denote this unique set by $P_{c,p}$.

Definition 1. *Let $t \in \mathrm{Terms}(\mathcal{B}_d)$. The graph $\mathcal{G}(t)$ associated with t is defined as follows.*

1. $V(\mathcal{G}(t)) = \{(c, P) : c \in [d],\ P \subseteq \mathrm{Pos}(t),\ P \text{ is a } c\text{-colored component}\}$.
2. $E(\mathcal{G}(t)) = \{e_p : p \in \mathrm{Pos}(t),\ t(p).b \neq \emptyset\}$.
3. $\mathrm{Inc}(\mathcal{G}(t)) = \{(e_p, (c, P_{c,p})) : c \in [d],\ e_p \in E(\mathcal{G}(t)),\ c \in T[p].b\}$.

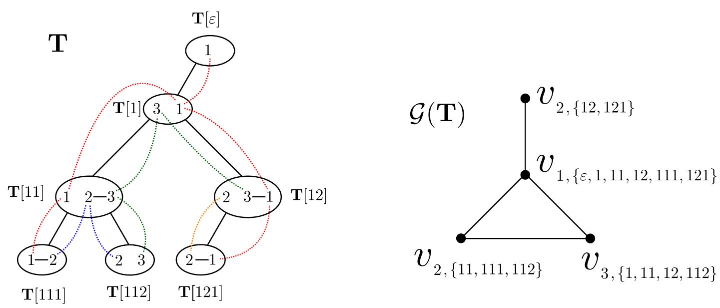

Fig. 1. A 3-concrete decomposition t and its corresponding graph $\mathcal{G}(t)$. The set $\{\varepsilon, 1, 11, 12, 111, 121\}$ is a 1-colored component. The sets $\{12, 121\}$ and $\{11, 111, 112\}$ are 2-colored components. The set $\{1, 11, 12, 112\}$ is a 3-colored component. The root of this last set is 1, and its leaves are 112 and 12.