Output-Linear Enumeration of Variable-Inclusion MAX for Extended Variable Automata

Valentina Silva ✉
Pontificia Universidad Católica de Chile

Cristian Riveros ✉
Pontificia Universidad Católica de Chile

Abstract

Algebraic frameworks for information extraction model a document as a finite string and pattern matches as *mappings* that bind variables to spans. While recent automata-based techniques can enumerate *all* mappings with delay linear in the size of each output, practical applications do not always require the full result: many mappings are redundant because they are *strictly extended* by others that capture additional variables without altering previously assigned spans. The *variable-inclusion skyline* (or MAX) operator removes this redundancy, retaining only maximal mappings.

We provide a evaluation scheme that integrates MAX *during* enumeration while preserving output-linear delay. Starting from a deterministic extended variable automaton (eVA) E, we construct a product automaton $E_{\max}$ whose state pairs keep, in lock-step, a candidate run and a bounded set of dominating witnesses. A mapping is output exactly when no witness can still reach acceptance. These results establish that skyline selection can be incorporated into automata-based text extraction without sacrificing enumeration performance.

2012 ACM Subject Classification *

Keywords and phrases Complex event recognition, constant-delay enumeration, skyline selection strategy, maximal mappings.

1 Introduction

**

2 Preliminaries

Documents, alphabet, and spans. We fix a finite alphabet Σ. A **document** is a finite string $d = a_1 \dots a_n \in \Sigma^*$. A **span** is a half-open interval $[i, j\rangle$ with $1 \le i \le j \le |d| + 1$, it denotes the substring $d[i, j\rangle = a_i \dots a_{j-1}$. The sets of all spans of d is written $Spans(d)$ [2].

Variables, markers and mappings. Let V be a finite set of **variables**. For each $x \in V$ we use two **variable markers** $[x$ open and $x\rangle$ close; $Markers_V = \{[x, x\rangle | x \in V\}$ [3]. A **mapping** or valuation is a partial function.

$$\mu : dom(\mu) \subseteq V \to Spans(d).$$

Two mappings are **compatible** when they agree on their common variables. Mappings constitute thr basic tuples produced by our operators.

Document spanners. A document spanner P associates to every string d a finite set of mappings over some variable set $V = SVars(P)$ [2]. Intuitively, a spanner "extracts" all matches of a pattern as span relations.

Extended Variable Automata (eVA). REmatch compiles each REQL query to an extended variable-set automaton

$$E = (Q, q_0, F, \delta),$$

whose transitions are quadruples (q, a, S, q') with $a \in \Sigma \cup \{\#\}$ and a (possibly empty) set of markers S. While reading the $i-th$ symbol, the automaton outputs the pair (S, i); if $S = \varnothing$ nothing is produced. A run sequence

$$q_0 \xrightarrow{b_0/S_0} q_1 \xrightarrow{b_1/S_1} \dots \xrightarrow{b_n/S_n} q_{n+1}$$

that alternates variable transitions and letter transitions and respects marker nesting. The mapping defined by a valid accepting run is obtained by pairing every $[x$ with the corresponding $x\rangle$. Determinisation via a subset construction yields a deterministic eVA guaranteeing at most one accepting run per output sequence, a key invariant for output-linear enumeration [2].

Determinisation of extended VA. an eVA $E = (Q, q_0, F, \delta)$ is deterministic when

$$\delta : Q \times (\Sigma \cup (2^{Markers_V} \{\varnothing\})) \to Q$$

is a partial function: for every state q and pail (a, S) there is at most one outgoing transition (q, a, S, q'). Determinism guarantees that, for any document d and any output sequence, at most one accepting run produces it – a key property to avoid duplicates during enumeration. Every eVA con be turned into an equivalent deterministic eVA via the classical powerset method.

Variable-inclusion order. let V be a finite set of variables, for two mappings μ, ν over the same document we write

$$\mu \preccurlyeq_{varinc} \nu$$

If for all $x \in V$, if $\mu(x)$ is defined, then $\nu(x)$ is defined and $\mu(x) = \nu(x)$.

MAX operator. given a spanner P we define

$$[\![MAX(P)]\!]_d = \{\mu \in P(d) | \text{there is no } \nu \in P(d) \text{ with } \mu \preccurlyeq_{varinc} \nu\}$$

That is, we keep only those mappings that are maximal under $\preccurlyeq_{varinc}$ [1].

3 Main results

Deterministic eVA. Let $E = (Q, q_0, F, \delta)$ be an eVA. Its pair-based subset construction yields the deterministic eVA

$$E_{\text{det}} = \big(Q_{\text{det}}, X_0, F_{\text{det}}, \Delta\big),$$

with

$$\begin{aligned} Q_{\text{det}} &= 2^Q, \\ X_0 &= \{q_0\}, \\ F_{\text{det}} &= \{X \subseteq Q \mid X \cap F \neq \varnothing\}, \\ \Delta(X, a, S) &= \{\, q' \mid \exists q \in X : \ (q, a, S, q') \in \delta\}. \end{aligned}$$

Selection strategy MAX. Given a deterministic extended variable automaton (eVA) $E_{\text{det}} = (2^Q, X_0, F_{\text{det}}, \Delta)$, we define a new eVA

$$E_{\max} = \big(Q_{\max}, (R_0, W_0), F_{\max}, \Delta_{\max}\big)$$

that accepts exactly the mappings that are maximal under variable inclusion, where $Q_{max} = 2^Q \times 2^Q$ and

$$Q_{\max} = \{ (R, W) \mid R, W \subseteq Q_{\det} \text{ and } R \cap W = \varnothing \}.$$

Initial state. $(R_0, W_0) = (X_0, \varnothing)$.

Transition function. R represents a set of "current" states of E_{det} and W represents the set of states of E_{det} having a run that dominates the current run under variable inclusion. For $(R, W) \in Q_{\max}$, letter $a \in \Sigma \cup \{\#\}$, and marker set S:

$$\Delta_{\max}\big((R, W), a, S\big) = (R', W')$$

where

$$R' = \Delta(R, a, S) \backslash W' \quad \text{and} \quad W' = \begin{cases} \Delta(W, a, S) \cup \bigcup\limits_{S \neq \varnothing} \Delta(R, a, S') & \text{if } S' = \varnothing, \\ \Delta(W, a, S) \cup \bigcup\limits_{S \subset S'} \Delta(R, a, S') & \text{if } S \neq \varnothing. \end{cases}$$

Final states.

$$F_{\max} = \Big\{ (R, W) \in Q_{\max} \;\Big|\; R \cap F_{\det} \neq \varnothing \text{ and } W \cap F_{\det} = \varnothing \Big\}.$$

▶ **Proposition 1.** *Let $E_{\det} = (2^Q, X_0, F_{\det}, \Delta)$ be the deterministic eVA obtained from an eVA E, and let $E_{\max} = (Q_{\max}, (R_0, W_0), F_{\max}, \Delta_{\max})$ be its MAX–product automaton as defined above. Then for every document d,*

$$[\![E_{\max}]\!]_d = \Big\{ \mu \in [\![E_{\det}]\!]_d \;\Big|\; \nexists\, \nu \in [\![E_{\det}]\!]_d : \mu \prec_{\text{varInc}} \nu \Big\} = \textsc{Max}\big([\![E_{\det}]\!]_d\big).$$

That is, $E_{\max}$ accepts exactly the maximal mappings under variable–inclusion.

4 Conclusions

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

References

1. Marco Bucchi, Alejandro Grez, Andrés Quintana, Cristian Riveros, and Stijn Vansummeren. CORE: a complex event recognition engine. *VLDB*, 15(9):1951–1964, 2022.
2. Ronald Fagin, Benny Kimelfeld, Frederick Reiss, and Wang-Chiew Tan. Document spanners: A formal approach to information extraction. *Journal of the ACM*, 62(2):12:1–12:51, 2015. `doi:10.1145/2699430`.
3. Fernando Florenzano, Cristian Riveros, Martín Ugarte, Stijn Vansummeren, and Domagoj Vrgoč. Efficient enumeration algorithms for regular document spanners. *ACM Transactions on Database Systems*, 45(1):3:1–3:42, 2020. `doi:10.1145/3351451`.

A Proofs of Section 3

A.1 Proof of Proposition 1 1

Proof. We show two inclusions.

(1) $[\![E_{\max}]\!]_d \subseteq \textsc{Max}([\![E_{\det}]\!]_d)$

If $E_{\max}$ accepts via a run $\rho_{\max}$ producing mapping μ, then μ is maximal under $\preceq_{\text{varInc}}$.

Indeed, acceptance in $E_{\max}$ means that in the final state $(R, W) \in F_{\max}$ we have $R \cap F_{\det} \neq \varnothing$ (so $\rho_{\max}$ yields μ) and $W \cap F_{\det} = \varnothing$. The fact $W \cap F_{\det} = \varnothing$ guarantees that no "witness" run that dominates the reference run ever reached an accepting configuration. But any mapping $\nu \succ_{\text{varInc}} \mu$ in $[\![E_{\det}]\!]_d$ would correspond to some such dominating run in the subset–construction, and hence would force $W \cap F_{\det} \neq \varnothing$. Contradiction. Therefore μ cannot be strictly extended, i.e. it is maximal.

(2) $\textsc{Max}([\![E_{\det}]\!]_d) \subseteq [\![E_{\max}]\!]_d$

If μ is maximal in $[\![E_{\det}]\!]_d$, E_{max} accepts μ. Let ρ be the unique accepting run of $E_{\det}$ that produces μ; denote by $X_0 \xrightarrow{(a_1,S_1)} X_1 \dots X_n$ the sequence of subset–states visited by ρ. We simulate ρ in $E_{\max}$ and show inductively that after processing the first k input positions $(0 \le k \le n)$ the witness component W_k contains *no* final states:

$$W_k \cap F_{\det} = \varnothing, \qquad X_k \in R_k.$$

- *Base case.* Initially $(R, W) = (X_0, \varnothing)$; clearly $W \cap F_{\det} = \varnothing$.
- *Inductive step.* Assume the invariant for k. When reading (a_{k+1}, S_{k+1}) the reference component follows ρ to X_{k+1}. The update rule adds to W (a) successors of the existing witnesses; (b) any successor of X_k that emits a *strict superset* of S_{k+1}. If any state inserted under (b) could reach $F_{\det}$, then the corresponding run would realise a mapping $\nu \succ_{\text{varInc}} \mu$, contradicting the maximality of μ. By induction the invariant holds for $k + 1$.

  .

After the last symbol we have $R_n \cap F_{\det} \neq \varnothing$ (contains X_n) and $W_n \cap F_{\det} = \varnothing$, so the final composite state is accepting. Hence $E_{\max}$ outputs μ.

Combining (1) and (2) yields the desired equality $[\![E_{\max}]\!]_d = \textsc{Max}([\![E_{\det}]\!]_d)$. ◀