Constant-delay Enumeration for Lorem Ipsum

Fernando Riveros ✉
Pontificia Universidad Católica de Chile

Cristian Riveros ✉
Pontificia Universidad Católica de Chile

Abstract

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.

2012 ACM Subject Classification Theory of computation → Database theory

Keywords and phrases Streams, query evaluation, enumeration algorithms.

1 Introduction

Write an introduction here.

Cristian: Here a comment from Cristian.

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.

2 Preliminaries

Sets and intervals. Given a set A, we denote by 2^A the *powerset* of A. We denote by $\mathbb{N}$ the natural numbers. Given $n, m \in \mathbb{N}$ with $n \leq m$, we denote by $[n]$ the set $\{1, \ldots, n\}$ and by $[n..m]$ the interval $\{n, n+1, \ldots, m\}$ over $\mathbb{N}$.

Events and streams. We fix a set $\mathbf{T}$ of *event types*, a set $\mathbf{A}$ of *attributes names*, and a set $\mathbf{D}$ of *data values* (e.g., integers, floats, strings). An *event* e is a partial mapping $e : \mathbf{A} \to \mathbf{D}$ that maps attributes names in $\mathbf{A}$ to data values in $\mathbf{D}$. We denote att(e) the domain of e, called the attributes of e, and we assume that att(e) is finite. We denote by $e(A)$ the data value of the attribute $A \in \mathbf{A}$ whenever $A \in att(e)$. Further, each event e has a type in $\mathbf{T}$ denoted by type(e). We write $\mathbf{E}$ to denote the set of all events over event types $\mathbf{T}$, attributes names $\mathbf{A}$, and data values $\mathbf{D}$. A *stream* is an (arbitrary long) sequence $\bar{S} = e_1 e_2 \ldots e_n$ of events where $|S| = n$ is the length of the stream.

Complex events. Fix a finite set $\mathbf{X}$ of variables and assume that $\mathbf{T} \subseteq \mathbf{X}$, where $\mathbf{T}$ is the set of event types as defined earlier, this is to say that all event types are a variable. Let $\bar{S}$ be a stream of length n. A complex event of $\bar{S}$ is a triple (i, j, μ) where $i, j \in [n]$, $i \leq j$, and $\mu : \mathbf{X} \to 2^{[i..j]}$ is a function with finite domain. Intuitively, i and j marks the beginning and end of the interval where the complex event happens, and μ stores the events in the interval $[i..j]$ that fired the complex event. In the following, we will usually use C to denote a complex event (i, j, μ) of $\bar{S}$ and omit $\bar{S}$ if the stream is clear from the context. We will use $\text{interval}(C)$, $\text{start}(C)$, and $\text{end}(C)$ to denote the interval $[i..j]$, the start i, and the end j

$$
\begin{aligned}
[\![R]\!](\bar{S}) &= \{\, (i,i,\mu) \mid \text{type}(e_i) = R \wedge \mu(R) = \{i\} \wedge \forall X \neq R.\ \mu(X) = \varnothing \,\} \\
[\![\varphi \text{ AS } X]\!](\bar{S}) &= \{\, C \mid \exists C' \in [\![\varphi]\!](\bar{S}).\ \text{interval}(C) = \text{interval}(C') \wedge C(X) = \textstyle\bigcup_Y C'(Y) \\
&\qquad \wedge \forall Z \neq X.\ C(Z) = C'(Z) \,\} \\
[\![\varphi \text{ FILTER } X[P]]\!](\bar{S}) &= \{\, C \mid C \in [\![\varphi]\!](\bar{S}) \wedge C(X) \vDash P \,\} \\
[\![\varphi_1 \text{ OR } \varphi_2]\!](\bar{S}) &= [\![\varphi_1]\!](\bar{S}) \cup [\![\varphi_2]\!](\bar{S}) \\
[\![\varphi_1 \text{ AND } \varphi_2]\!](\bar{S}) &= [\![\varphi_1]\!](\bar{S}) \cap [\![\varphi_2]\!](\bar{S}) \\
[\![\varphi_1 ; \varphi_2]\!](\bar{S}) &= \{\, C_1 \cup C_2 \mid C_1 \in [\![\varphi_1]\!](\bar{S}) \wedge C_2 \in [\![\varphi_2]\!](\bar{S}) \wedge \text{end}(C_1) < \text{start}(C_2) \,\} \\
[\![\varphi_1 : \varphi_2]\!](\bar{S}) &= \{\, C_1 \cup C_2 \mid C_1 \in [\![\varphi_1]\!](\bar{S}) \wedge C_2 \in [\![\varphi_2]\!](\bar{S}) \wedge \text{end}(C_1) + 1 = \text{start}(C_2) \,\} \\
[\![\varphi^+]\!](\bar{S}) &= [\![\varphi]\!](\bar{S}) \cup [\![\varphi ; \varphi^+]\!](\bar{S}) \\
[\![\varphi^\oplus]\!](\bar{S}) &= [\![\varphi]\!](\bar{S}) \cup [\![\varphi : \varphi^\oplus]\!](\bar{S}) \\
[\![\pi_L(\varphi)]\!](\bar{S}) &= \{\, \pi_L(C) \mid C \in [\![\varphi]\!](\bar{S}) \,\}
\end{aligned}
$$

Figure 1 Figure 1: The semantics of CEL formulas defined over a stream $\bar{S} = e_1 e_2 \ldots e_n$ where each e_i is an event.

of C, respectively. Further, by some abuse of notation we will also use C(X) for $X \in \mathbf{X}$ to denote the set $\mu(X)$ of C.

The following operations on complex events will be useful throughout the paper. We define the union of complex events C_1 and C_2, denoted by $C_1 \cup C_2$, as the complex event C' such that start(C') = min{start(C_1),start(C_2)}, end(C') = max{end(C_1), end(C_2)}, and C'(X) = $C_1(X) \cup C_2(X)$ for every $X \in \mathbf{X}$. Further, we define the *projection over* L of a complex event C, denoted by π_L(C), as the complex event C' such that interval(C') = interval(C) and C'(X) = C(X) whenever $X \in L$, and C'(X) = $\varnothing$, otherwise. Finally, we denote by $(i, j, \mu_\varnothing)$ the complex event with trivial mapping $\mu_\varnothing$ such that $\mu_\varnothing$(X) = $\varnothing$ for every $X \in \mathbf{X}$.

Predicate of events. A *predicate* is a possibly infinite set $\mathbf{P}$ of events. We say that an event e satisfies predicate P, denoted $e \vDash P$, if, and only if, $e \in P$. We generalize this notation from events to a set of events E such that $E \vDash P$ if, and only if, $e \vDash P$ for every $e \in E$. We assume a fixed set of predicates $\mathbf{P}$. Further, we assume that there is a basic set of predicates P_basic $\subseteq P$ and $\mathbf{P}$ is the closure of P_basic under intersection and negation (i.e., $P_1 \cap P_2 \in \mathbf{P}$ and $\mathbf{E}$ $P \in \mathbf{P}$ for every $P, P_1, P_2 \in \mathbf{P}$) where $\mathbf{E}$ is a predicate in $\mathbf{P}$, that we usually denote by true.

Complex event logic. In this work, we use the Complex Event Logic (CEL) introduced in [21] and implemented in CORE [11] as our basic query language for CER. The syntax of a CEL formula φ is given by the grammar:

$$\varphi := R \mid \varphi \text{ AS } X \mid \varphi \text{ FILTER } X[P] \mid \varphi \text{ OR } \varphi \mid \varphi \text{ AND } \varphi \mid \varphi ; \varphi \mid \varphi : \varphi \mid \varphi+ \mid \varphi\oplus \mid \pi_L(\varphi)$$

where $R \in \mathbf{T}$ is an event type, $X \in \mathbf{X}$ is a variable, $P \in \mathbf{P}$ is a predicate, and $L \subseteq \mathbf{X}$ is a set of variables. We define the semantics of a CEL formula φ over a stream $\bar{S}$, recursively, as a set of complex events over $\bar{S}$ In Figure 1, we define the semantics of each CEL operator like in [11, 21].

3 Main results

In this section we introduce an extension to the semantics of CEL, namely we introduce a new operator using [allen interval algebra] *overlap*. We then extend the formal computational model for evaluating CEL formulas and prove its correctness. We start by recalling the notion of a CEA to later extend the proof. **Complex Event Automata**. A *Complex Event Automata* (CEA) is a tuple $\mathcal{A} = (Q,\mathbf{P},\mathbf{X},\Delta,q_0,F)$ where Q is a finite set of states, $\mathbf{P}$ is the set of predicates, $\mathbf{X}$ is a finite set of variables, $\Delta \subseteq Q \times \mathbf{P} \times 2^{\mathbf{X}} \times Q$ is a finite relation (called the transition relation), $q_0 \in Q$ is the inital state, and F is the set of final states. A run ρ of $\mathcal{A}$ over the stream $\bar{S} = e_1 e_2 \dots e_n$ from position i to j is a sequence:

$$\rho := p_i \xrightarrow{P_i/L_i} p_{i+1} \xrightarrow{P_{i+1}/L_{i+1}} p_{i+2} \xrightarrow{P_{i+2}/L_{i+2}} \dots \xrightarrow{P_j/L_j} p_{j+1}$$

where $p_i = q_0, (p_k, P_k, L_k, p_{k+1}) \in \Delta$, and $e_k \vDash P_k$ for all $k \in [i..j]$. We say that the run is accepting if $p_{j+1} \in F$. A run ρ from positions i to j like above defines the complex event $C_\rho = (i, j, \mu_\rho)$ such that $\mu_\rho(X) = k \in [i..j] \mid X \in L_k$ for every $X \in \mathbf{X}$. Note that the starting and ending positions i, j of the run define the interval of the complex event, and the labels $L_k \subseteq \mathbf{X}$ define the mapping μ_ρ of C_ρ. We define the set of all complex events of $\mathcal{A}$ over $\bar{S}$ as:

$$[\![\mathcal{A}]\!](\bar{S}) = \{C_\rho \mid \rho \text{ is an accepting run of } \mathcal{A} \text{ over } \bar{S}\}$$

We present then the overlap operator for CEL as with the following definition:

$$[\![\varphi_1 \ {:}o \ \varphi_2]\!](\bar{S}) = \{C_1 \cup C_2 \mid C_1 \in [\![\varphi_1]\!](\bar{S}) \wedge C_2 \in [\![\varphi_2]\!](\bar{S}) \\ \wedge\, start(C_1) \leq start(C_2) \leq end(C_1) \leq end(C_2)\}$$

We also know from [11,22] the following theorem:

▶ **Theorem 1** (CEA and CEL equivalence)**.** *For every CEL formula φ there exists a CEA $\mathcal{A}_\varphi$ such that $[\![\varphi]\!](\bar{S}) = [\![\mathcal{A}_\varphi]\!](\bar{S})$ for every stream $\bar{S}$*

To maintain the correctness of it true, we extend the induction proof [11,22] by proving the following property: There exists a $\mathcal{A}_{:o}$ be a CEA as defined previously. Let φ_1 and φ_2 formulas in CEL. Then

$$[\![\varphi_1 \ {:}o \ \varphi_2]\!](\bar{S}) = [\![\mathcal{A}]\!](\bar{S})$$

Lets assume then that there exists an automaton that satisfies the previous property for φ_1 and φ_2, therefore we know there exists $\mathcal{A}_{\varphi_1} = (Q_1, \mathbf{P}_1, \mathbf{X}_1, \Delta_1, q_0, F_1)$ and $\mathcal{A}_{\varphi_2} = (Q_2, \mathbf{P}_2, \mathbf{X}_2, \Delta_2, p_0, F_2)$ Then the construction for the overlap operator is as folllows:
let $\mathcal{A}_{:o}$ be a CEA such that $\mathcal{A}_{:o} = (Q_1 \uplus Q_2 \uplus Q_1 \times Q_2, \mathbf{P}_1 \cup \mathbf{P}_2, \mathbf{X}_1 \cup \mathbf{X}_2, \Delta_{:o} \uplus \Delta_1 \uplus \Delta_2, q_0, F_2)$ where $\Delta_{:o}$ is defined as:
$\Delta_{:o} = \{(q, \mathrm{TRUE}, \varnothing, (q, p_0)) \mid q \in Q_1, (q, p_0) \in Q_1 \times Q_2\} \cup$
$\{((q,p), P_1 \wedge P_2, X_1 \cup X_2, (q', p')) \mid (q, P_1, X_1, q') \in \Delta_1, (p, P_2, X_2, p') \in \Delta_2, (q,p), (q',p') \in Q_1 \times Q_2\} \cup$
$\{((q,p), \mathrm{TRUE}, \varnothing, p) \mid q \in F_1, p \in Q_2\}$
Intuitively, given an stream S we capture the eventes given φ_1, and at some point (the overlap) we start capturing the events for φ_2 too.

We also recall that the base construction of a CEA $\mathcal{A}_R$ with $R \in \mathbf{T}$ an event type, is as follows: $\mathcal{A}_\varphi = (\{q_1, q_2\}, \Delta_\varphi, \{q_1\}, \{q_2\})$ with $\Delta_\varphi = \{(q_1, R, \mu(R), q_2), (q_1, \mathrm{TRUE}, \varnothing, q_1)\}$.

Because of this we note that in $\Delta_{:o}$ the transitions that represent the overlapped section mark correctly when P_1 and P_2 are triggered simultaneously by an event, and they too mark correctly when only one of the predicates is triggered. This conclusion comes from the fact that when we define $\{((q,p), P_1 \wedge P_2, X_1 \cup X_2, (q',p'))\}$ where $(q, P_1, X_1, q') \in \Delta_1, (p, P_2, X_2, p') \in \Delta_2$ the transitions in both Δ_1 and Δ_2 include the case $(q,\text{TRUE}, \varnothing, q)$, therefore when only one of the predicates is triggered there always exists a transition in $\Delta_{:o}$ defined as $((q,p), P_1 \cap \text{TRUE}, X_1 \cup \varnothing, (q',p))$ where $(q, P_1, X_1, q') \in \Delta_1$, $(p,\text{TRUE}, \varnothing, p) \in \Delta_2$ and $(q,p),(q',p) \in Q_1 \times Q_2$.

The proof is by double containment.
T.P. $[\![\varphi_1 \text{ :o } \varphi_2]\!](\bar{S}) \subseteq [\![\mathcal{A}_{:o}]\!](\bar{S})$

Let $C_1 \cup C_2 \in [\![\varphi_1 \text{ :o } \varphi_2]\!](\bar{S})$ where $C_i \in [\![\varphi_i]\!](\bar{S}) = [\![\mathcal{A}_{\varphi_i}]\!](\bar{S})$ with $i \in \{1,2\}$. From this we extend that there must exists a run on both $\mathcal{A}_{\varphi_1}$ and $\mathcal{A}_{\varphi_2}$ that accept C_1 and C_2 respectively. This is:

$$\rho_1 : q_0 \xrightarrow{P_0/X_0} q_1 \xrightarrow{P_1/X_1} \dots \xrightarrow{P_{n-1}/X_{n-1}} q_n$$

$$\rho_2 : p_0 \xrightarrow{P'_0/X'_0} p_1 \xrightarrow{P'_1/X'_1} \dots \xrightarrow{P'_{m-1}/X'_{m-1}} p_m$$

With $q_n \in F_1$ and $p_m \in F_2$. By the previous construction of $\mathcal{A}_{:o}$ we can start building a run $\rho_{:o}$ as follows:

$$\rho_{:o} : q_0 \xrightarrow{P_0/X_0} q_1 \xrightarrow{P_1/X_1} \dots \xrightarrow{P_{i-1}/X_{i-1}} q_i$$

with $i \leq n$. By definition we know that $start(C_1) \leq start(C_2)$ therefore we can extend the run as such:

$$\rho_{:o} : \dots q_i \xrightarrow{\text{TRUE}/\varnothing} (q_i, p_0)$$

And then by construction:

$$\rho_{:o} : \dots (q_i, p_0) \xrightarrow{P_i \wedge P'_0/X_i \cup X'_0} (q_{i+1}, p_1) \xrightarrow{P_{i+1} \wedge P'_1/X_{i+1} \cup X'_1} \dots \xrightarrow{P_{n-1} \wedge P'_{j-1}/X_{n-1} \cup X'_{j-1}} (q_n, p_j)$$

With $j \leq m$ and $q_n \in F_1$. By definition we know that $end(C_1) \leq end(C_2)$ therefore we can extend the run once more:

$$\rho_{:o} : \dots (q_n, p_j) \xrightarrow{\text{TRUE}/\varnothing} p_j$$

Finally then, by construction:

$$\rho_{:o} : \dots p_j \xrightarrow{P'_j/X'_j} p_{j+1} \xrightarrow{P'_{j+1}/X'_{j+1}} \dots \xrightarrow{P'_{n-1}/X'_{n-1}} p_n$$

Because $p_n \in F_2$, then $\rho_{:o}$ is a run of $\mathcal{A}_{:o}$ that accepts $C_1 \cup C_2$ over a stream $\bar{S}$. Then $[\![\varphi_1 \text{ :o } \varphi_2]\!](\bar{S}) \subseteq [\![\mathcal{A}_{:o}]\!](\bar{S})$.

T.P. $[\![\mathcal{A}_{:o}]\!](\bar{S}) \subseteq [\![\varphi_1 \text{ :o } \varphi_2]\!](\bar{S})$
Let $\rho_{:o}$ be a run of $\mathcal{A}_{:o}$ that accepts C over $\bar{S}$:

$$\rho_{:o} : q_0 \xrightarrow{P_0/X_0} \dots \xrightarrow{\text{TRUE}/\varnothing} (q_i, p_0) \xrightarrow{P_i \wedge P'_0/X_i \cup X'_0} \dots \rightarrow (q_n, p_{j+1}) \xrightarrow{\text{TRUE}/\varnothing} \dots \xrightarrow{P'_{n-1}/X'_{n-1}} p_m$$

By construction we can define the runs ρ_1 and ρ_2 of $\mathcal{A}_1$ and $\mathcal{A}_2$ as follows:

$$\rho_1 : q_0 \xrightarrow{P_0/X_0} q_1 \xrightarrow{P_1/X_1} \dots \xrightarrow{P_{n-1}/X_{n-1}} q_n$$

$$\rho_2 : p_0 \xrightarrow{P'_0/X'_0} p_1 \xrightarrow{P'_1/X'_1} \dots \xrightarrow{P'_{m-1}/X'_{m-1}} p_m$$

Where the complex event accepted by this runs we denote by C_1 and C_2. By construction its easy to see that $start(C_1) \leq start(C_2) \leq end(C_1) \leq end(C_2)$ and $C_1 \in [\![\varphi_1]\!](\bar{S}) \wedge C_2 \in [\![\varphi_2]\!](\bar{S})$. We now prove that $C = C_1 \cup C_2$.

Once again, by construction it holds true that $\text{start}(C) = \min\{\text{start}(C_1), \text{start}(C_2)\}$, $\text{end}(C) = \max\{\text{end}(C_1), \text{end}(C_2)\}$, we are left to prove that $C(X) = C_1(X) \cup C_2(X)$ for every $X \in \mathbf{X}$.

Let interval(C) $= [s \dots r]$ with $0 \leq s$ and $r \leq m$, let $C(X) = k \in [s \dots r] \mid X \in L_k$ for all $X \in \mathbf{X}$. For k to belong to C(X) then the predicate P_k must be triggered at some point during the run, this can happen on one of three segments of the run: when the transition are Δ_1, Δ_2 or $\Delta_{:o}$.

If the predicate was triggered by a transition in Δ_1 then $k \in C_1(X)$ and $k \in C(X)$. Similarly if it was a transition in Δ_2 then $k \in C_2(X)$ and $k \in C(X)$.

If the predicate was triggered by a transition in $\Delta_{:o}$ (i.e. $((q,p), P_1 \wedge P_2, X_1 \cup X_2, (q', p'))$) then, by what was said before, there are two scenarios:

- both $P_1 \neq$ TRUE and $P_2 \neq$ TRUE, and both $X_1 \neq \varnothing$ and $X_2 \neq \varnothing$: therefore $k \in C_1(X), k \in C_2(X)$ and $k \in C(X)$.
- P_1 = TRUE and $X_1 = \varnothing$ or P_2 = TRUE and $X_2 = \varnothing$: therefore $k \in C_2(X)$ and $k \in C(X)$ or $k \in C_1(X)$ and $k \in C(X)$.

From this we deduce that $k \in C_1(X)$ if and only if $k \in C(X)$, and $k \in C_2(X)$ if and only if $k \in C(X)$. Therefore, $C = C_1 \cup C_2$.

Then $[\![\mathcal{A}_{:o}]\!](\bar{S}) \subseteq [\![\varphi_1 \text{ :o } \varphi_2]\!](\bar{S})$.

Finally $[\![\mathcal{A}_{:o}]\!](\bar{S}) = [\![\varphi_1 \text{ :o } \varphi_2]\!](\bar{S})$.

□

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. Guillaume Bagan, Arnaud Durand, and Etienne Grandjean. On acyclic conjunctive queries and constant delay enumeration. In *CSL*, pages 208–222, 2007.
2. Marco Bucchi, Alejandro Grez, Andrés Quintana, Cristian Riveros, and Stijn Vansummeren. CORE: a complex event recognition engine. *VLDB*, 15(9):1951–1964, 2022.
3. Arnaud Durand and Etienne Grandjean. First-order queries on structures of bounded degree are computable with constant delay. *ACM Trans. Comput. Log.*, 8(4):21, 2007.

A Proofs from Section 2

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.

B Proofs of Section 3

B.1 Proof of Lemma ??

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.

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.

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.

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.

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.

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.

B.2 Proof of Theorem 1

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

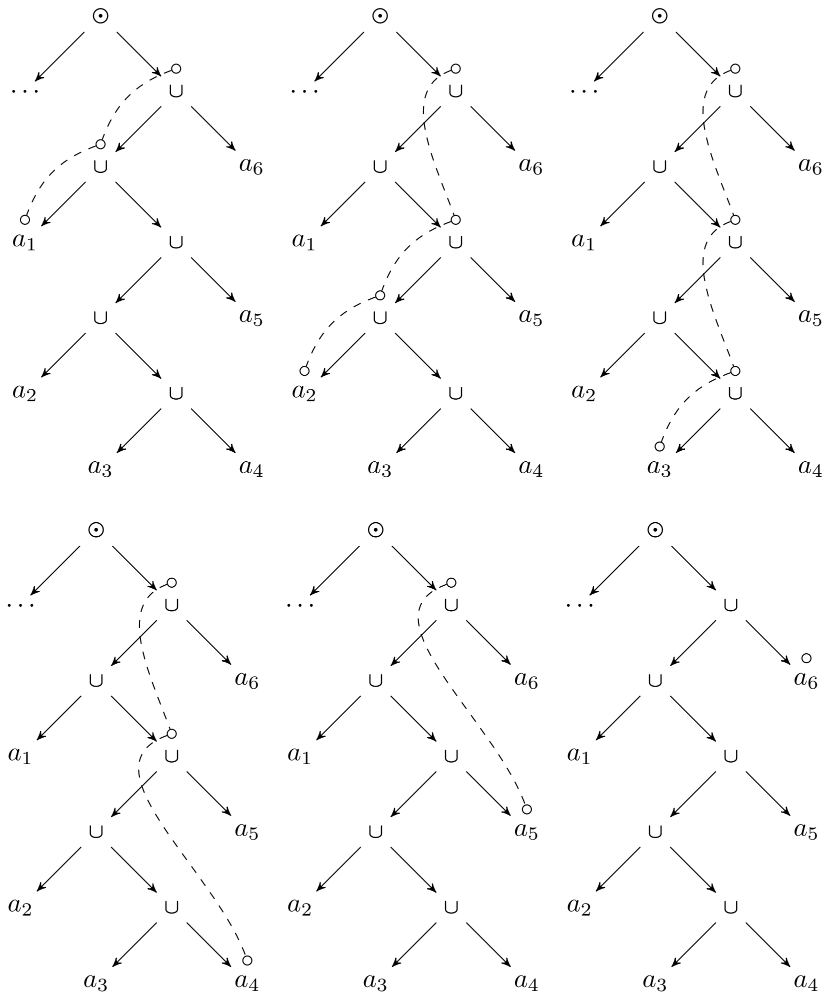

Figure 2 An example iteration of `trav` and `move`. The sequences of nodes joined by dashed lines represent a stack St, where the first one was obtained after calling `trav` over the topmost union node, and the following five are obtained by repeated applications of `move`(St).

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.

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. 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.

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

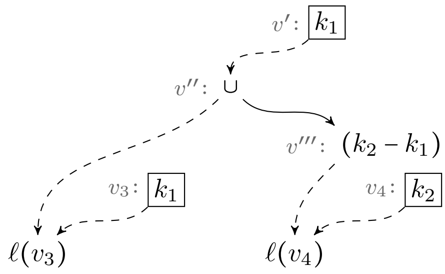

Figure 3 Gadget used in Theorem 1.

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.

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.

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.

B.3 Proof of Proposition ??

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.

▷ Claim 2. Fix $k \in \mathbb{N}$. Let $\mathcal{C}_k$ be the class of all duplicate-free and k-bounded D that satisfy the ϵ condition. Then one can solve the problem $\texttt{Enum}[\mathcal{C}_k]$ with output-linear delay and without preprocessing (i.e. constant preprocessing time).

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.

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.

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

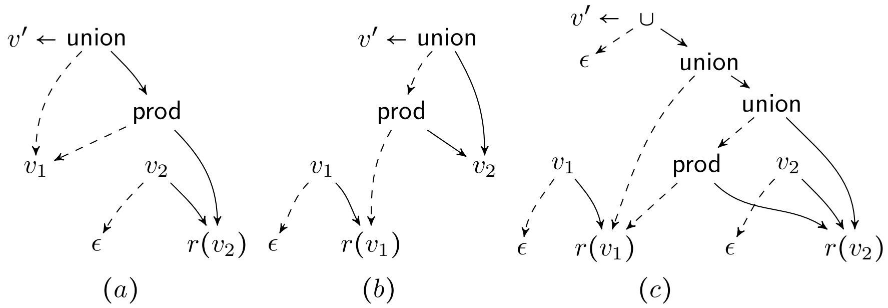

Figure 4 Gadgets for product as defined for an $\mathcal{D}$ with the ϵ-node.

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.

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.