ON THE AMORTIZED COST OF AN ODOMETER

V. Berthé (LIRMM/CNRS), C. Frougny (LIAFA/Univ. Paris 8),
M. Rigo (Univ. Liège), J. Sakarovitch (ENST/CNRS)

Work in progress...
So far, different aspects of odometers have been studied:
Combinatorial, Metrical, Topological, Dynamics, Sequential properties, \ldots

G. Barat, T. Downarowicz, C. Frougny, P. Grabner, P. Liardet, R. Tichy, A. M. Vershik, \ldots

**Our main question**

What is the *cost* / *complexity* in average for computing the odometer (i.e., successor map) on *finite words*, e.g. on integer representations?

\[
\begin{align*}
n & \longrightarrow \text{rep}(n) \in \Sigma^* \\
n + 1 & \longrightarrow \text{rep}(n + 1) \in \Sigma^* 
\end{align*}
\]
WHERE DOES IT COME FROM?

**Words’05**

E. Barcucci, R. Pinzani, M. Poneti, *Exhaustive generation of some regular languages by using numeration systems.*

For numeration systems built on some linear recurrent sequences of order 2, the “amortized cost” for computing $\text{rep}(n+1)$ from $\text{rep}(n)$ is bounded by a constant (CAT).

**J. Sakarovitch, ELTS. de théorie des automates’03**

For any rational set $R$ of $A^*$, the odometer on $R$ is a synchronized function.

i.e., letter-to-letter (left or right) finite transducer with a terminal function appending values of the form $(u, \varepsilon)$ or $(\varepsilon, v)$
More than synchronized functions, we will often assume that we have a (right) **sequential** transducer to do the computation.

A transducer $T$ is sequential if

- $T$ has a unique initial state,
- the underlying input automaton is deterministic.
More than synchronized functions, we will often assume that we have a (right) **sequential** transducer to do the computation.

A transducer $T$ is sequential if

- $T$ has a unique initial state,
- the underlying input automaton is **deterministic**.
Usual binary system

A (TRIVIAL) SEQUENTIAL FUNCTION

\[
\begin{align*}
&1/0 \\
\rightarrow & & 1/0 \\
&0/1 \\
\rightarrow & & 0/1 \\
&0/0 \\
\rightarrow & & 0/0 \\
&1/1
\end{align*}
\]

10100111
10101000
**DEFINITION (COST)**

We define the **cost** for computing $\text{rep}(n + 1)$ from $\text{rep}(n)$ as

- the position up to where the carry propagates, or
- the length of the path lying in the “transient part”,
- for an integer base system, the number of changed digits.
ALTERNATIVE DEFINITION (COST)

Another interpretation for the cost in the lexicographic tree:
- half of the distance between \( \text{rep}(n) \) and \( \text{rep}(n+1) \)
- distance to the common ancestor of \( \text{rep}(n) \) and \( \text{rep}(n+1) \)
Alternative definition (Cost)

Another interpretation for the cost in the lexicographic tree:

- Half of the distance between \( \text{rep}(n) \) and \( \text{rep}(n + 1) \)
- Distance to the common ancestor of \( \text{rep}(n) \) and \( \text{rep}(n + 1) \)
So, cost can be expressed mainly on words

\[ uav \longrightarrow ubv', \quad a \neq b, \quad |v| = |v'| \]

\[
\text{cost} = |av|
\]

Let us introduce a different notion (computational aspects)

**Definition (Complexity)**

The (algorithmic) complexity for computing rep\((n + 1)\) from rep\((n)\) is the minimum number of operations required to perform this computation (in the sense of a Turing machine).
Consider a numeration system such that the odometer can be realized by a letter-to-letter (right) sequential transducer. In that case, the cost is equal to the (algorithmic) complexity. Indeed, it is not possible to do less computations, the Turing machine at least has to read the digits up to where the carry propagates

```
# 1 0 0 1 0 # #
```

“cost $\leq$ complexity”
\[ X^2 - 3X + 1, \beta = \frac{3 + \sqrt{5}}{2}, d_\beta(1) = 21^\omega, (U_n)_{n \geq 0} = 1, 3, 8, 21, \ldots \]

\[ \text{rep}(\mathbb{N}) = \{ \varepsilon, 1, 2, 10, 11, 12, 20, 21, 100, 101, 102, \ldots \} \]

forbidden factors : \(21 \star 2\)

\[100111111 \rightarrow 100111112 \text{ but } 102111111 \rightarrow 110000000\]
\[ X^2 - 3X + 1, \quad \beta = \frac{3 + \sqrt{5}}{2}, \quad d_\beta(1) = 21^\omega, \quad (U_n)_{n \geq 0} = 1, 3, 8, 21, \ldots \]
\[ \text{rep}(\mathbb{N}) = \{ \varepsilon, 1, 2, 10, 11, 12, 20, 21, 100, 101, 102, \ldots \} \]
\[ \text{forbidden factors : 2} \cdot 1 \]

100111111 → 1001111112 but 102111111 → 110000000
\[ X^2 - 3X + 1, \beta = \frac{3 + \sqrt{5}}{2}, d_\beta(1) = 21^\omega, (U_n)_{n \geq 0} = 1, 3, 8, 21, \ldots \]
rep(\mathbb{N}) = \{\varepsilon, 1, 2, 10, 11, 12, 20, 21, 100, 101, 102, \ldots\}
forbidden factors : 2 1* 2
100111111 → 100111112 but 102111111 → 110000000
All this work on cost/complexity can be done in a general setting.

**Definition**

An abstract numeration system is a triple $S = (L, A, <)$ where $L$ is an infinite (rational) language over a totally ordered alphabet $(A, <)$.

The representation of $n \in \mathbb{N}$ is the $(n + 1)$-st word in the genealogically (i.e., radix) ordered language $L$.

**Example**

$L = \{(ab), (ac)\}^*$, $a < b < c$

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$\varepsilon$</td>
<td>$ab$</td>
<td>$ac$</td>
<td>$abab$</td>
<td>$abac$</td>
<td>$acab$</td>
<td>$acac$</td>
<td>$ababab$</td>
<td>...</td>
</tr>
</tbody>
</table>
In base $k$, $k^n$ words from $\varepsilon$ to $(k-1) \cdots (k-1)$,

$$
\frac{k^n + k^{n-1} + \cdots + 1}{k^n} = \frac{k - k^{-n}}{k - 1} \to \frac{k}{k - 1}, \text{ as } n \to \infty
$$

**Definition (Amortized Cost)**

$$
\lim_{n \to \infty} \left( \sum_{w \in \mathcal{L}, |w| \leq n} \text{cost}(w) \right) / \#\{w \in \mathcal{L} : |w| \leq n\}
$$

Same for amortized complexity
**Example continues... Amortized cost / complexity**

In base $k$, $k^n$ words from $\varepsilon$ to $(k-1) \cdots (k-1)$,

$$\frac{k^n + k^{n-1} + \cdots + 1}{k^n} = \frac{k - k^{-n}}{k - 1} \to \frac{k}{k - 1}, \text{ as } n \to \infty$$

**Definition (Amortized cost)**

$$\lim_{n \to \infty} \left( \frac{\sum_{w \in \mathcal{L}, |w| \leq n} \text{cost}(w)}{\#\{w \in \mathcal{L} : |w| \leq n\}} \right)$$

Same for amortized complexity
**First Exercise...**

For Fibonacci system...

\[
\begin{array}{cccccccc}
\varepsilon & \rightarrow & 1 & \rightarrow & 10 & \rightarrow & 100 & \rightarrow & 101 \rightarrow & 1000 & \rightarrow & 1010 & \rightarrow \\
1 & 2 & 3 & 1 & 4 & 2 & 5
\end{array}
\]

amortized cost = amortized complexity \[\frac{\tau}{\tau - 1} \approx 2.618\]
FIRST EXERCISE...

For Fibonacci system...

\[ \varepsilon \rightarrow 1 \rightarrow 10 \rightarrow 100 \rightarrow 101 \rightarrow 1000 \rightarrow 1010 \rightarrow \]

\[ 1 \quad 2 \quad 3 \quad 1 \quad 4 \quad 2 \quad 5 \]

amortized cost = amortized complexity \[ \rightarrow \frac{\tau}{\tau - 1} \approx 2.618 \]
**Theorem**

Let $L$ be a rational language having $\mathcal{M}$ as trim minimal automaton.

If the adjacency matrix $M$ of $\mathcal{M}$ is primitive with $\beta > 1$ as dominating Perron eigenvalue and if all states of $\mathcal{M}$ are final, then the amortized cost of the odometer on $L$ is $\frac{\beta}{\beta - 1}$.

**Remark**

- If the corresponding transducer is right sequential, then this is exactly the amortized (algorithmic) complexity.
- Otherwise, we get information on the average position up to where some change can occur. (More?)

**Remark**

All states final means $L$ is prefix closed.
Perron Theory

Let $M$ be a $d \times d$ primitive matrix having $\beta > 1$ as dominating eigenvalue. The following holds

$$\forall i, j \in \{0, \ldots, d - 1\}, \exists c_{ij} > 0 : (M^n)_{ij} = c_{ij} \beta^n + o(\beta^n).$$

If $x$ (resp. $y$) is a left $1 \times d$ (resp. right $d \times 1$) eigenvector of $M$ of eigenvalue $\beta$ such that $x.y = 1$ then $\forall 0 \leq i, j < d$,

$$c_{ij} = y_i x_j, \quad i.e., \quad \lim_{n \to \infty} \frac{M^n}{\beta^n} = y.x.$$
If $w = \text{pas}$ is such that

- $q_0.p = q_j$,
- $a \neq \max A_{q_j}$
- $s \in \max(L_{q_0.p_a})$

Fix $q_j \in Q$

$p_{\text{as}} \rightarrow p_{\text{bt}}$, $|s| = |t|$

Then sum over $Q$...
Let $\beta > 1$ be a Parry number. The amortized cost of the odometer for the canonical linear numeration system associated with $\beta$ is $\frac{\beta}{(\beta - 1)}$.

Same remark: cost = complexity when assuming that the odometer is realized with a right sequential transducer.

For such $\beta$-numeration systems ($\beta$ being a Parry number), we have

- a right sequential transducer in the finite type,
- but NOT in the sofic case.
simple Parry number

\[ q_1 \rightarrow q_2 \rightarrow \cdots \rightarrow q_m \]

non-simple case

\[ q_1 \rightarrow q_2 \rightarrow \cdots \rightarrow q_N \]

\[ q_{N+1} \rightarrow q_{N+2} \rightarrow \cdots \rightarrow q_{N+p} \]

\[ q_{N+p} \rightarrow q_{N+p+1} \rightarrow \cdots \rightarrow q_{N+1} \]
RESULT

Let $S = (L, A, <)$ be an abstract numeration system built on a rational language whose trim minimal automaton $M$ is primitive and has only final states. If $\beta$ is the dominating eigenvalue of $M$ then the amortized cost of the odometer for $S$ is $\frac{\beta}{(\beta - 1)}$.

Same remark: cost = complexity when assuming that the odometer is realized with a right sequential transducer.

NEXT STEP, EASY TO HANDLE

Consider several primitive strongly connected components...
Let’s have a look at the lexicographic tree

**Fibonacci Words of Length 5**

```
1
10
100
1000
10000
10001
10010
10100
10101
```

This diagram represents the Fibonacci words of length 5, where each node is labeled with a Fibonacci word. The tree structure illustrates the lexicographic order of these words.
Let’s have a look at the lexicographic tree

**Fibonacci Words of Length 5**

```
10000   10001   10010   10100   10101
1000 1001 1010
100 101
1 10
```
Let's have a look at the lexicographic tree

**Fibonacci Words of Length 5**

- 1
- 10
- 100
- 1000
- 10000

- 1001
- 10010
- 1010
- 10100
- 10101
- 10101
Let’s have a look at the lexicographic tree

**Fibonacci words of length 5**

```
10000   10001   10010   10100   10101
1000 1001 1010
100 101
1
```

![Lexicographic tree for Fibonacci words of length 5](chart.png)
Let's have a look at the lexicographic tree

**Fibonacci Words of Length 5**
Let’s have a look at the lexicographic tree

**Fibonacci Words of Length 5**

```
10000   10001   10010   10100   10101
1000 1001 1010
100 101
1
```
Consequently, the total cost for all words of length \( n \) is

\[
C_n := \# \{\text{edges in } T_n\} + 1 = \# \{\text{leaves in } T_n\} = \#(L \cap \Sigma^{\leq n})
\]

“Nice” hypothesis :

\begin{itemize}
\item \( L \) is a prefix closed language (\( uv \in L \Rightarrow u \in L \))
\item Any branch in the tree is infinite
\end{itemize}

**Remark**

If \( u_L : n \mapsto \#(L \cap \Sigma^n) \) has a “nice asymptotic behavior”, then the amortized cost can be computed...

\[
\lim_{n \to \infty} \frac{\sum_{i=0}^{n} C_i}{\#(L \cap \Sigma^{\leq n})} = \lim_{n \to \infty} \frac{\sum_{i=0}^{n} \sum_{k=0}^{i} u_L(k)}{\sum_{i=0}^{n} u_L(i)}
\]
Can we compute the amortized complexity if there is no sequential transducer behind?

If $L$ is rational, when can the odometer be computed with a right sequential transducer? (local automaton)
$p/q$-base (S. Akiyama, C. Frougny, J. Sakarovitch)

$p > q \geq 1$ coprime integers,

$$N = \sum_{i=0}^{k} \frac{a_i}{q} \left(\frac{p}{q}\right)^i, \quad 0 \leq a_i < p$$
$p/q$-base (S. Akiyama, C. Frougny, J. Sakarovitch)

$p > q \geq 1$ coprime integers,

$$N = \sum_{i=0}^{k} \frac{a_i}{q} \left(\frac{p}{q}\right)^i, \quad 0 \leq a_i < p$$
The language of numeration is “highly” non-rational: any two sub-trees of the lexicographic tree are non-isomorphi
c but it is easy to build a digit-to-digit right sequential transducer
that realizes the odometer

**PROPOSITION**

For the base $p/q$ system, $p > q \geq 1$, the amortized cost (resp. complexity) is

$$\frac{p}{q} - 1$$
**Example of language with zero entropy**

$a^*b^*$ is a rational, prefix closed language (and any branch in the lexicographic tree is infinite)

$$u(n) = \#(a^*b^* \cap \{a, b\}^n) = n + 1,$$

therefore

$$\lim_{n \to \infty} \frac{\sum_{i=0}^{n} \#(L \cap \Sigma \leq i)}{\#(L \cap \Sigma \leq n)} = \lim_{n \to \infty} \frac{\frac{1}{6}(n + 1)(n + 2)(n + 3)}{\frac{1}{2}(n + 1)(n + 2)} = +\infty.$$