On Entropic Measures of Computations:

Entropic Weight of Domain Partitions

Anatol Slissenko

Laboratory for Algorithmics, Complexity and Logic (LACL)
Université Paris-Est-Créteil
FRANCE

Intuitively comprehensible: during its work an algorithm

- processes the information contained in its input;
- and approaches the result.

One can try to look at computations as trajectories, and either

– to seek for some geometry in the set of these trajectories, OR

– to try to evaluate what happen with the information on these trajectories.

Unfortunately, there is no rigorous (e.g., mathematical) theory of information that reflects our intuition.

(The information theory is a theory of entropy, that is a measure of uncertainty, not of information).

From now on we consider the work of algorithms on inputs of length n, and for better intuition we suppose that this set is 'big', say, of size exponential in n.

Geometry starts with metric.
Each computation is a sequence of events, i.e., executions of commands .
We call such sequence a *trace*, and
denote by $tr(X)$ the trace generated by input X.

What metric or something similar can we define in the set $\{tr(X)\}_X$?

If we try to define it on the basis of Kolmogorov entropy we arrive at a measure $\mathfrak{K}(\alpha,\beta) = K(\alpha|\beta) + K(\beta|\alpha)$ that is not quite a metric, strictly speaking, but can be seen as measure of metric flavor.

But it gives 'distances' of order n that can hardly be perceived as satisfactory.

We can try to measure a distance on a basis of some similarity of events and something like symmetric difference between traces. We illustrate it in a wider context.

We start with examples.

Example: sum σ of bits of a string of length n modulo 2, i.e. XOR.

Input: A string x over an alphabet $\mathbb{B} = \{0,1\}$. The length $n = |x|$ of the input is its size; $x(i)$ is the ith bit.

Output: $\sigma(x) = \sum_{i=1}^{n} x(i) \mod 2$.

Algorithm:

1: $i := 0;\ s := 0;$ %Initialization

2: if $i < n$ then $i := i + 1;\ s := s + x(i);$ goto 2

3: else $\sigma := s;$ halt % case $i \geq n$

All traces of this algorithm are symbolically similar:

$i := 0;\ s := 0;\ i < n;\ i := i + 1;\ s := s + x(i);\ i < n; \ldots, i \geq n, ;\ \sigma := s$

but the values of s and of $x(i)$ depend on the input.

Eliminate all internal (i.e., updated by σ) functions and get literals trace ([...] is the value in the appropriate trace):

$$0 = 0,\ 0 = 0,\ 0 < n,\ 1 = 1,\ x(1) = [x(1)],\ 1 < n, \ldots,$$
$$n \geq n,\ x(1) + \cdots + x(n) = [x(1)] + \cdots + [x(n)],$$

where $[x(1)] + \cdots + [x(n)]$ is a constant term or the result of its evaluation (depending on applications).

In this literals trace we eliminate trivially true literals and keep only those that depend on inputs indeed:

$0{=}0$, $0{=}0$, $0{<}n$, $1{=}1$, $x(1) = [x(1)]$, $1{<}n$, . . . ,

$n \geq n$, $x(1) + \cdots + x(n) = [x(1)] + \cdots + [x(n)]$

and get a weeded literals trace

$$x(1) = v_1, \quad x(1) + x(2) = v_2, \ldots, x(1) + \cdots + x(n) = v_n$$

If we define similarity as equality of trace literals with evaluated right-hand side then $x(1) + x(2) = 0 + 1$ and $x(1) + x(2) = 1 + 0$ are similar; this is weak similarity.

More productive is strong similarity :

trace literals are equal if they are equal with non evaluated constant terms in the right-hand side.

In the example just above: $x(1) + x(2) = 0 + 1$ and $x(1) + x(2) = 1 + 0$ are not similar.

These considerations also shows other ways to define similarity.

Come back to measuring distance between traces of σ.

Take weak similarity.

Then the traces are distinguished only by v_i in

$$x(1) = v_1, \quad x(1) + x(2) = v_2, \ldots, x(1) + \cdots + x(n) = v_n$$

This a trace can be identified with a Boolean vector from $\mathbb{B}^n$.

If we measure a distance as symmetric difference divided by 2, we are in Boolean cube $\mathbb{B}^n$ with Hamming metric.

This vision does not help to understand the geometry of computation.

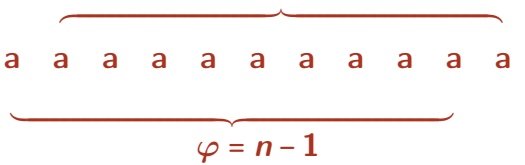
Not to speak about the fact that the number of traces is bounded by the number of inputs.

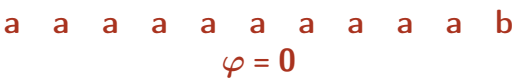
Example: maxPS: maximal prefix-suffix

Input: A string w over an alphabet $\mathcal{A}$, $\alpha = |\mathcal{A}| \geq 2$.
The length $n = |w|$ of the input is its size.

Output: the longest prefix $\varphi(w)$ of w that is different of w
and is also a suffix of w.

E.g.:





We consider two algorithms: $\varphi_0(w)$ and $\varphi_1(w)$.

Algorithm $\varphi_0(w)$ tries all possible values of $\varphi(w)$ starting with the maximal, that is with $(n-1)$, comparing $w(i) = w(i+h)$.

Its complexity is quadratic, and a worst-case input is $a \ldots ab$, $a \neq b$.

Algorithm $\varphi_1(w)$ recursively computes $\varphi(w)$.

Its complexity is linear. We consider it for the same input $a \ldots ab$, $a \neq b$ that is in some way worst-case.

For both algorithms the weeded traces consist of literals

$$w(i) = w(j) \text{ and } w(i) \neq w(j).$$

Some generalities about algorithms.

We consider en algorithm $\mathfrak{A}$ that computes a one-component function F over inputs of size n.

The inputs constitute the domain $dm(F)$, and $rn(F) = F(dm(F))$ is its range.

Everything is finite, the size of inputs is fixed: n.

The work of $\mathfrak{A}$ over $dm(F)$ is represented as a set of traces.

A trace $tr(X)$ is a sequence of commands executed by $\mathfrak{A}$ for an input X.

An execution of a command in a trace $tr(X)$ at instant t is an (atomic) event $tr(X, t)$: a guard or an update.

To be productive in the general setting we assume that $dm(F)$ is 'big' (of exponential size in terms of n) and the size of events and their values is $\mathcal{O}(\log n)$). These constraints are quite practical.

Traditionally, the syntax is the program of $\mathfrak{A}$, and the semantics is the set of runs or the set of traces.

This semantics is 'logical': how the program rewrites states.

From the set-theoretic viewpoint of mathematics,
the semantics is the graph of F.

How a trace approaches the graph?

How to relate the events (the syntax) to the 'ultimate' semantics,
i.e., to the graph of F?

How to measure the approximation of the graph of F by events?

This is the question!

My current choice:

The approximation is measured in terms of probabilistic measure.

The events are related to the graph of F by this or that notion of similarity.

(Here similarity is an equivalence relation.)

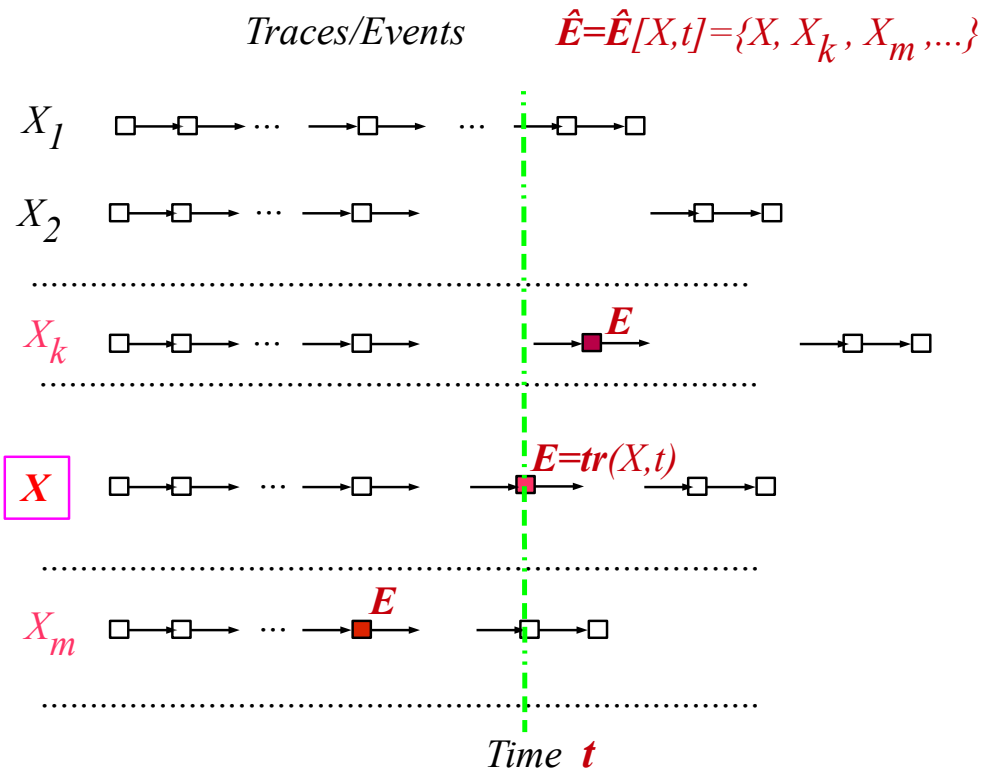
Suppose that some similarity $\sim$ of events is chosen.

Notice that the similarity may use not only events themselves but also all or a part of its history.

A class of similar events is related to a subset of $dm(f)$.

To an event $E = tr(X, t)$ we relate a set $\widehat{E} = \widehat{E}[X, t]$ as follows:

$$\widehat{E}[X, t] = \{X' \in dm(f) : \exists t'.\, E \sim tr(X', t')\}$$


Traces/Events
$\hat{E}=\hat{E}[X,t]=\{X, X_k, X_m, ...\}$
X_1
X_2
X_k
E
X
E=tr(X,t)
E
X_m
Time t

For the fixed similarity $\sim$ of events we attribute to each event an ordered partition of $\widehat{E}$

Let $M = |rn(F)|$, and the values of F are ordered: $rn(F) = (F_1, \ldots, F_M)$.

Denote: $\widehat{F}_k =_{df} F^{-1}(F_k)$, $1 \le k \le M$.

For each event E we define an ordered partition

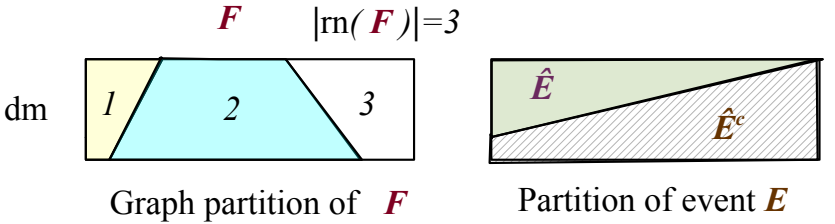
$$\boxed{\pi(E) = \pi(\widehat{E}) =_{df} (\widehat{E} \cap \widehat{F}_1,\ \widehat{E} \cap \widehat{F}_2,\ \ldots,\ \widehat{E} \cap \widehat{F}_M).}$$

In particular,

$$\Pi_F =_{df} \pi(dm(F)) = (\widehat{F}_1,\ \widehat{F}_2,\ \ldots,\ \widehat{F}_M)$$

$$\Pi_{F_k} =_{df} \pi(\widehat{F}_k) = (\varnothing,\ \ldots, \varnothing,\ \widehat{F}_k,\ \varnothing, \ldots, \varnothing) \text{ (the graph of } F)$$

So the graph of F is the set $gr(F) =_{df} \{\Pi_{F_k}\}_k$.


F
|rn(F)|=3
dm
1
2
3
Graph partition of F
Ê
Ê^c
Partition of event E
1
2
3
Partition to measure entropic weight

Denote by $\mathfrak{R}^{\sim}(t)$ (or simply by $\mathfrak{R}(t)$ if $\sim$ is clear from the context)

the set of all points (partitions) $\pi(tr(X,\tau))$, where $\tau \geq t$ and $X \in dm(F)$,

plus the graph $gr(F)$ of F.

We call $\mathfrak{R}^{\sim}(t)$ a space of events at instant t.

Its usual volume is $|\mathfrak{R}^{\sim}(t)|$ (the number of points);

logarithm of the volume corresponds to epsilon-entropy.

This volume depends on $\sim$.

Example of σ for 'weak' similarity the volume is polynomial,

but for 'strong' similarity it is exponential.

It is more productive to consider the volume weighted by the entropic weight defined just below.

How to choose measure and similarity?

The measure is chosen on the basis of Principle of Maximal Uncertainty :

For $\mathfrak{A}$ all the results (outputs) have equal probabilities.

Measure. $\boxed{P(F^{-1}(v)) = \frac{1}{|rn_n(F)|}}$ for any $v \in rn(F)$, and

P is uniform on $F^{-1}(v)$.

Motivation: imagine that $\mathfrak{A}$ plays against an adversary that chooses any input to ensure the maximal uncertainty for $\mathfrak{A}$. In this case all outputs of $rn(F)$ are equiprobable.

For our examples: $P(\sigma^{-1}(0)) = P(\sigma^{-1}(1)) = \frac{1}{2}$ (and $|\sigma^{-1}(0)| = |\sigma^{-1}(1)| = 2^{n-1}$)

$P(\varphi^{-1}(k)) = \frac{1}{n}$ (here the sizes of $\varphi^{-1}(k)$ are very different)

We can define a metric between partitions $\pi(E)$, e.g., like

$$d((A_1, \ldots, A_M), (B_1, \ldots, B_M)) = \sum_{1 \le i \le M} P(A_i \triangle B_i),$$

where $\triangle$ is symmetric difference of sets.

However, it is not clear whether the detailed structure of this (or other) metric might clarify the question of convergence of an algorithm to its result.

(Moreover, it is hard to calculate distances even for MaxPS. Only for σ it is relatively easy: for the weak similarity introduced below, the distances take values $\frac{1}{4}, \frac{1}{2}, 1$.)

Whether this or that metric may be useful for information convergence is not clear.

Entropic weight of event E (in fact, that of $\pi(E)$):

$$\mathcal{D}(E) = \mathcal{D}(\widehat{E}) = \mathcal{D}(\pi(E)) = -\sum_k P(\widehat{E} \cap \widehat{F}_k) \log \frac{P(\widehat{E} \cap \widehat{F}_k)}{P(\widehat{E})}$$

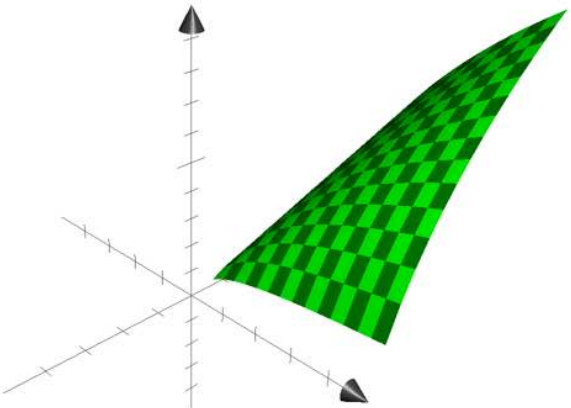
Properties of entropic weight:

(D1) $\mathcal{D}(dm(F)) = \mathcal{D}(\Pi_F) = \log M$ (maximal uncertainty)

(D2) $\mathcal{D}(\widehat{F}_k) = 0$ (maximal certainty)

(D3) $\mathcal{D}(E) = 0$ for $E \subseteq \widehat{F}_k$ for all k (certainty)

(D4) $\mathcal{D}$is monotone: $S_0 \subset S_1 \to \mathcal{D}(S_0) \leq \mathcal{D}(S_1)$



Graph of entropic weight of two variables
$z = -x \log x - y \log y + (x + y) \log(x + y)$

As it was mentioned above not all events are relevant to processing the information in inputs.

We take as relevant the events in weeded traces as defined above.

In the general case it is hard to distinguish relevant and irrelevant events.

If we take an event (e.g., defining a loop counter) that is present in all traces then it gives partition

$$\Pi_{F=_{df}\pi(dm(F))} = (\widehat{F}_1, \widehat{F}_2, \ldots, \widehat{F}_M)$$

whose entropic weight is maximal i.e., $\log M$

So we look at the evolution of the entropic weight of events with time and decide what events are relevant to our analysis of the behavior of $\mathfrak{A}$.

Types of convergence mentioned above.

- Convergence along traces: $\mathcal{D}(tr(X,t))$ for $t \to \infty$

(here ∞ is the worst-case time complexity of $\mathfrak{A}$)

- Convergence in terms of properties of $\mathfrak{R}(t)$: e.g.,

$$|\mathcal{D}\mathfrak{R}(t)| = \sum_{\pi(E)\in\mathfrak{R}(t)} \mathcal{D}(\pi(E)) \text{ – weighted volume}$$

One can try other criteria, e.g.,

$$\max\{\mathcal{D}(E) : \pi(E) \in \mathfrak{R}(t)\} \quad \text{(worst-case over } \mathfrak{R}(t)\text{)}$$

What may be *similarity* of events in general, is not quite clear for me.

Weak similarity for σ.

This similarity is based on symbolic form of events and the values of the involved functions.

Two such events $x(1) + \cdots + x(i) = v$ and $x'(1) + \cdots + x'(i') = v'$ are similar if

$$\boxed{i = i' \text{ and } v = v'.}$$

For event Φ we have $\widehat{E} = \{x : \Phi\}$ and $\pi(E) = (\Phi \cap (\sigma = 0), \Phi \cap (\sigma = 1))$.

Hence, $\mathcal{D}(E) = \frac{1}{2}$, e.g., is constant, and

so there is no convergence along traces.

This is an example of Laurent Bienvenu.

Look at the space $\mathfrak{R}(t)$.

After t steps there are $\dfrac{2t}{3}$ executed essential events that calculate σ_i,

so the number of remaining events of this type is $\dfrac{2(3n-t)}{3}$,

where $3n+\mathcal{O}(1)$ is the complexity of the algorithm.

Taking into account that $t=3i$,
the volume of $\mathfrak{R}(t)$ is of the form $\mathcal{O}((3n-t))$, or $\boxed{\mathcal{O}(n-i)}$.

So we see a convergence in terms of
(weighted or not weighted) volume of $\mathfrak{R}(t)$.

Strong similarity for σ.

Two essential events $x(1) + \cdots + x(i) = v$ and $x'(1) + \cdots + x'(i') = v'$ are strongly similar if all their preceding events are respectively weakly similar, i.e., $i = i'$ and

$$x(1) = x'(1),\ x(1) + x(2) = x'(1) + x'(2), \ldots, x(1) + \cdots + x(i) = x'(1) + \cdots + x'(i').$$

In this case the size of $\mathfrak{R}(i)$ is exponential, and

$$P(\sigma_i = a) = \frac{2^{n-i}}{2^n} = \frac{1}{2^i}, \quad P((\sigma_i = a) \cap (\sigma = 0)) = \frac{1}{2^{i+1}}, \quad \mathcal{D}(\sigma_i = a) = \frac{1}{2^i}$$

For weighted (by $\mathcal{D}$) volume we have the same expression as for the weak similarity:

$$\boxed{\mathcal{O}(n-i)}$$

Besides this there is convergence along all traces:

$$\mathcal{D}(tr(x,t)) = \frac{1}{2^k}, \text{ where } k = \frac{t-2}{3}.$$

Consider convergence of φ_m along traces, that is, to my mind, is more instructive that the convergence in terms of the volume of $\mathfrak{R}$.
Order $rn(\varphi) = \{0, 1, \ldots, n-1\}$.

Similarity: $w(i) = w(j)$ and $w(i') = w(j')$ are similar if $i = i'$ and $j = j'$.

Consider the trace for a worst-case input $w = a \ldots ab$, $a \neq b$.

Notice that estimation of $|\varphi^{-1}(k)|$ for $k < \frac{n}{2}$ is an open problem.

Take φ_0. After $n-1$ comparisons $w(i) = w(i+1)$ algorithm φ_0 arrives at $w(n-1) \neq w(n)$.
The entropic weight of this event is 0, and thus, algorithm 'knows' the result $\varphi(w) = 0$.

Algorithm φ_0 converges to this event not slower than $\frac{n-i}{n} \log n$, where i is the number of comparison $w(i) = w(i+1)$.

However, it continues to work. How evolves the entropic weight of the events that follows?

The event E next to $w(n-1) \neq w(n)$ is $w(1) \neq w(3)$.
Let $S = \widehat{E}$.

What is entropic weight of $w(1) \neq w(3)$, i.e., $\mathcal{D}(S)$?

What values of $\varphi(w)$ are excluded by this event?

In order to evluate $\mathcal{D}(S) = -\sum_{0 \le k \le (n-1)} P(S \cap \widehat{\varphi}_k) \log \frac{P(S \cap \widehat{\varphi}_k)}{P(S)}$

we should estimate

$$P(S) = \sum_{k=0}^{n-1} P(S \cap \widehat{\varphi}_k) = \sum_{k=0}^{n-1} \frac{|S \cap \widehat{\varphi}_k|}{n \cdot |\widehat{\varphi}_k|},$$

but here we stumble upon open combinatorial problems.

With an considerable effort I can show $\mathcal{D}(S) \ge \frac{\log n}{4\alpha} - c(\alpha)$,

where $c(\alpha)$ is a small positive constant.

After the next $(n-2)$ comparisons φ_0 arrives at event $E_1 =_{df} w(n-2) \neq w(n)$,

whose entropic weight becomes 'very small': $\mathcal{D}(E_1) \le O\left(\alpha^{-\frac{n}{2}}\right)$.

This 'non-smooth' behavior continues that show that φ_0 does not use the obtained information effectively.

Look again at $\mathcal{D}(S) = -\sum_{0 \le k \le (n-1)} P(S \cap \widehat{\varphi}_k) \log \frac{P(S \cap \widehat{\varphi}_k)}{P(S)}$.

We see that after each event $S =_{df} w(n-h) \neq w(n)$ we have $S \cap \widehat{\varphi}_k = \varnothing$ for $k = n-1, n-2, \ldots, n-h$ that diminishes $\mathcal{D}(S)$.

What is the price (in the number of comparisons) to pay for elimination of one value of φ?

$n-1$ comparisons to eliminate $\varphi = n-1$
$n-2$ comparisons to eliminate $\varphi = n-2$ etc.
Thus about $\frac{n}{2}$ on the average.
Whether we can do better?

Yes, φ_1 after the event $w(n-1) \neq w(n)$ (with 0 entropic weight)

it spends one comparison to eliminate one value of φ.

Remaks.

- Similarity and trace literals for other algorithms

Trace literals of the standard wave algorithm for the shortest path are inequalities and equalities of sums of weights of edges.
For similarity (that is always context dependent) only edges involved are important not their values.

- The number of partitions, is it sufficient?

For problems of the level $NP \cup coNP$ the number of partitions is exponential with respect to the cardinality of domain.
So very likely there are enough partitions for the analysis of algorithms.

As for problems of higher classes, their inputs do not give explicitly the structures sufficient to define the problem. If we try to develop such an input into a structure whose substructures explicitly define the problem, the input becomes bigger.

Conclusion

Concepts developed:
trace event, trace literal, measure,
event determined partition,
entropic weight

Current Open Questions.

- Extend the approach onto algorithmic problems
- Find methods for entropy estimations
  (either work out or bypass tough combinatorics)
- Look for other ways to define measures like entropic weight

Find the beginning of everything

and you will understand a lot.

Where is the beginning of that end that terminates the beginning?

(Koz'ma Prutkov)

The End