Estimating Information Amount under Uncertainty: Algorithmic Solvability and Computational Complexity

Vladik Kreinovich
Department of Computer Science
University of Texas at El Paso
El Paso, TX 79968, USA
vladik@cs.utep.edu
http://www.cs.utep.edu/vladik

interval computations website
http://www.cs.utep.edu/interval-comp

1. Types of Uncertainty: In Brief

- *Problem:* measurement result (estimate) $\widetilde{x}$ differs from the actual value x.
- *Probabilistic uncertainty:* we know which values of $\Delta x = \widetilde{x} - x$ are possible; we also know the frequency of each value, i.e., we know $F(t) \stackrel{\text{def}}{=} Prob(x \le t)$.
- *Interval uncertainty:* we only know the upper bound Δ on $|\Delta x|$; then, $x \in [\widetilde{x} - \Delta, \widetilde{x} + \Delta]$.
- *p-boxes:* for every t, we only know the interval $[\underline{F}(t), \overline{F}(t)]$ containing $F(t)$.
- *Fuzzy uncertainty:* we may also have expert estimates that provide better bounds Δx and on $F(t)$ with limited confidence.
- A nested family of intervals corresponding to different levels of certainty forms a *fuzzy number.*

2. Need to Compare Different Types of Uncertainty

- *Problem.* Often, there is a need to compare different types of uncertainty.
- *Example:* we have two sensors:
  - one with a smaller bound on a systematic (interval) component of the measurement error,
  - the other with the smaller bound on the standard deviation of the random component of the measurement error.
- *Question:* if we can only afford one of these sensors, which one should we buy?
- *Question:* which of the two sensors brings us more information about the measured signal?
- *Problem:* to gauge the amount of information.

3. Traditional Amount of Information: Brief Reminder

- *Shannon's idea:* (average) number of "yes"-"no" (binary) questions that we need to ask to determine the object.
- *Fact:* after q binary questions, we have 2^q possible results.
- *Discrete case:* if we have n alternatives, we need q questions, where $2^q \geq n$, i.e., $q \sim \log_2(n)$.
- *Discrete probability distribution:* $q = -\sum p_i \cdot \log_2(p_i)$.
- *Continuous case – definition:* number of questions to find an object with a given accuracy ε.
- *Interval uncertainty:* if $x \in [a, b]$, then $q \sim S - \log_2(\varepsilon)$, with $S = \log_2(b - a)$.
- *Probabilistic uncertainty:* $S = -\int \rho(x) \cdot \log_2 \rho(x)\, dx$.

4. How to Extend These Formulas to p-Boxes etc.

- *Problem:* extend the formulas for information to more general uncertainty.
- *Axiomatic approach (Klir et al.) – idea:*
  - find properties of information;
  - look for generalizations that satisfy as many of these properties as possible.
- *Problem:* sometimes, there are several possible generalizations.
- Which generalization should we choose?
- *Natural idea:* define information as the worst-case average number of questions.

5. Shannon's Derivation: Reminder

- *Situation:* we know the probabilities $p_1, \ldots, p_n$ of different alternatives.
- We repeat the selection N times.
- Let N_i be number of times when we get A_i.
- For big N, the value N_i is $\approx$ normally distributed with average $a = p_i \cdot N$ and $\sigma = \sqrt{p_i \cdot (1 - p_i) \cdot N}$.
- With certainty depending on k_0, we conclude that $N_i \in [a - k_0 \cdot \sigma, a + k_0 \cdot \sigma]$.
- Let $N_{\rm cons}(N)$ be the number of situations for which N_i is within these intervals.
- Then, for N repetitions, we need $q(N) = \log_2(N_{\rm cons})$ questions.
- Per repetition, we need $S = q(N)/N$ questions.

6. Shannon's Derivation (cont-d)

- *Shannon's theorem:* $S \to -\sum p_i \cdot \log_2(p_i)$.
- *Proof:* $N_{\rm cons} \sim$

$$\frac{N!}{N_1!(N-N_1)!}\cdot\frac{(N-N_1)!}{N_2!(N-N_1-N_2)!}\cdot\ldots = \frac{N!}{N_1!N_2!\ldots N_n!}$$

where $k! \sim (k/e)^k$. So,

$$N_{\rm cons} \sim \frac{\left(\frac{N}{e}\right)^N}{\left(\frac{N_1}{e}\right)^{N_1}\cdot\ldots\cdot\left(\frac{N_n}{e}\right)^{N_n}}$$

- Since $\sum N_i = N$, terms e^N and e^{N_i} cancel each other.
- Substituting $N_i = N \cdot f_i$ and taking logarithms, we get

$$\log_2(N_{\rm cons}) \approx -N\cdot f_1\cdot\log_2(f_1) - \ldots - N\cdot f_n \log_2(f_n).$$

7. Case of a Continuous Probability Distribution

- Once an *approximate* value r is determined, possible values of x form an interval $[r-\varepsilon, r+\varepsilon]$ of width 2ε.
- So, we divide the real line into intervals $[x_i, x_{i+1}]$ of width 2ε and find the interval that contains x.
- The average number of questions is
$$S = -\sum p_i \cdot \log_2(p_i),$$
where the prob. p_i that $x \in [x_i, x_{i+1}]$ is $p_i \approx 2\varepsilon \cdot \rho(x_i)$.
- So, for small ε, we have
$$S = -\sum \rho(x_i) \cdot \log_2(\rho(x_i)) \cdot 2\varepsilon - \sum \rho(x_i) \cdot 2\varepsilon \cdot \log_2(2\varepsilon).$$
- The first sum in this expression is the integral sum for the integral $S(\rho) \stackrel{\text{def}}{=} -\int \rho(x) \cdot \log_2(\rho(x))\, dx$, so
$$S \approx -\int \rho(x) \cdot \log_2(\rho(x))\, dx - \log_2(2\varepsilon).$$

8. Partial Information about Probability Distribution

- *Ideal case:* complete information about the probabilities $p = (p_1, \ldots, p_n)$ of different alternatives.
- *In practice:* often, we only have *partial* information about these probabilities.
- *In other words:* we have a set P of possible values of p.
- *Convexity of P:*
  - if it is possible to have $p \in P$ and $p' \in P$,
  - then it is also possible that we have p with some probability α and p' with the probability $1 - \alpha$.
- *Definition.* By the *entropy* $S(P)$ of a probabilistic knowledge P, we mean $S(P) \stackrel{\text{def}}{=} \max\limits_{p \in P} S(p)$.
- *Proposition.* When $N \to \infty$, the average number of questions tends to the $S(P)$.

9. Problem with this Definition

- *Problem:* the entropy is the same for two different cases:
  - when the distribution is uniform $p_1 = \ldots = p_n = 1$; and
  - when we have no information about the probabilities.
- *Why this is a problem:* in the second case, we have more uncertainty.
- *Natural solution (Klir):* instead of $S(P) = \max S(\rho)$, return the interval $\mathbf{S}(P) = [\min S(\rho), \max S(\rho)]$.
- *Problems with this solution:*
  - maximum of a convex function is easy to compute, minimum is not;
  - for a p-box, the minimum is often 0.

10. Discrete Case: Reminder and Outline

- *Uncertainty*: usually, there are several (n) different states which are consistent with our knowledge.
- *Question:* how much information we need to gain to determine the actual state of the world?
- *Natural measure:* average number S of "yes"-"no" questions that we need to ask.
- *Probabilistic case:* sometimes, we know the probabilities $p_1, \ldots, p_n$ of different states.
- *Problem:* often, we only know intervals $\mathbf{p}_i = [\underline{p}_i, \overline{p}_i]$.
- *Compute:* the range $\mathbf{S} = [\underline{S}, \overline{S}]$ of $S = -\sum p_i \cdot \log_2(p_i)$.
- *Result 1:* we can efficiently compute $\overline{S}$.
- *Result 2:* computing $\underline{S}$ is, in general, NP-hard.
- *Result 3:* we can efficiently compute $\underline{S}$ in many practically important situations.

11. Computing $\overline{S}$: Analysis of the Problem

- *Main idea:* if we change $p_j \to p_j + \Delta$ and $p_k \to p_k - \Delta$, then
$$\Delta S = \left(\frac{\partial S}{\partial x_j} - \frac{\partial S}{\partial x_k}\right) \cdot \Delta + o(\Delta) =$$
$$(-\log_2(p_j) + \log_2(p_k)) \cdot \Delta + o(\Delta) \le 0.$$

- *First conclusion:*
  - if $p_j \in (\underline{p}_j, \overline{p}_j)$ and $p_k \in (\underline{p}_k, \overline{p}_k)$, then $\Delta S \le 0$ for all small Δ
  - hence $-\log_2(p_j) + \log_2(p_k) = 0$ and $p_j = p_k = p$.

- *Second conclusion:* if $p_j = \underline{p}_j$ and $p_k > \underline{p}_k$, then $\Delta S \le 0$ for all small $\Delta > 0$ hence
$$p_k \le p_j.$$

- *Third conclusion:* if $p_j = \overline{p}_j$ and $p_k < \overline{p}_k$, then $\Delta S \le 0$ for all small $\Delta < 0$ hence $p_j \le p_k$.

12. Towards an Efficient Algorithm for Computing $\overline{S}$

- *Conclusions – reminder:*
  - if $p_j \in (\underline{p}_j, \overline{p}_j)$ and $p_k \in (\underline{p}_k, \overline{p}_k)$, then $p_j = p_k = p$;
  - if $p_j = \underline{p}_j$ and $p_k > \underline{p}_k$, then $p_k \le p_j$;
  - if $p_j = \overline{p}_j$ and $p_k < \overline{p}_k$, then $p_j \le p_k$.
- *Result of the analysis:* once we know the location of p in comparison to $\underline{p}_i$ and $\overline{p}_i$, then:
  - when $\overline{p}_i < p$, we have $p_i = \overline{p}_i$;
  - when $p < \underline{p}_i$, we have $p_i = \underline{p}_i$;
  - when $\underline{p}_i \le p \le \overline{p}_i$, we have $p_i = p$.
- *First stage:* sort $\underline{p}_i$ and $\overline{p}_i$.
- *Second stage:* for each of $2n + 2$ locations of p in this sorting, we perform $O(n)$ steps to find corr. p_i and S.
- *Computation time:* $(2n + 2) \cdot O(n) = O(n^2)$.

13. Quadratic-Time Algorithm for Computing $\overline{S}$

- Sort $\underline{p}_i$, $\overline{p}_i$: $p_{(0)} = 0 < p_{(1)} < \ldots < p_{(m)} < p_{(m+1)} = 1$.
- For every k from 0 to $m-1$, compute

$$M_k = -\sum_{i:\overline{p}_i \le p_{(k)}} \overline{p}_i \cdot \log_2(\overline{p}_i) - \sum_{j:\underline{p}_j \ge p_{(k+1)}} \underline{p}_j \cdot \log_2(\underline{p}_j);$$

$$P_k = \sum_{i:\overline{p}_i \le p_{(k)}} \overline{p}_i + \sum_{j:\underline{p}_j \ge p_{(k+1)}} \underline{p}_j;$$

$$n_k = \#\{i : \overline{p}_i \le p_{(k)} \vee \underline{p}_i \ge p_{(k+1)}\}.$$

- If $n_k = n$, we take $S_k = M_k$.
- If $n_k < n$, then we compute $p = \dfrac{1 - P_k}{n - n_k}$. If $p \in [p_{(k)}, p_{(k+1)}]$, then we compute

$$S_k = M_k - (n - n_k) \cdot p \cdot \log_2(p).$$

- Finally, we return the largest of these values S_k as the desired bound $\overline{S}$.

14. How to Reduce the Computation Time of Computing $\overline{S}$ to $O(n \cdot \log(n))$

- *Observation:* after sorting, we can mark, for each $p_{(j)}$, which endpoints coincide with this $p_{(j)}$.
- *Initial computation* M_0, P_0, and n_0 takes $O(n)$ steps.
- *Iteration:* from M_k to M_{k+1} (P_k to P_{k+1}), we only update the values corresponding to the zone's endpoints.
- Overall, for all the updates, we thus take as much time as there are updated values p_i.
- Each endpoint in this arrangement changes only once.
- *Conclusion:* overall, we use a linear number of steps $(2n)$ to update all the values M_k, P_k, and n_k.
- *Resulting computation time:*

$$O(n \cdot \log_2(n)) + O(n) + O(n) = O(n \cdot \log_2(n)).$$

15. $O(n \cdot \log(n))$ Time Algorithm for Computing $\overline{S}$

- Sort $\underline{p}_i$, $\overline{p}_i$: $0 = p_{(0)} < p_{(1)} < \ldots < p_{(m)} < p_{(m+1)} = 1$.
- Form sets $A_k^- \stackrel{\text{def}}{=} \{i : \underline{p}_i = p_{(k)}\}$, $A_k^+ \stackrel{\text{def}}{=} \{i : \overline{p}_i = p_{(k)}\}$.
- Compute $M_0 = -\sum\limits_{i=1}^n \underline{p}_i \cdot \log_2(\underline{p}_i)$, $P_0 = \sum\limits_{i=1}^n \underline{p}_i$, $n_0 = n$.
- For $k = 1, \ldots, m$, compute

$$M_{k+1} = M_k + \sum_{j \in A_{k+1}^-} \underline{p}_j \cdot \log_2(\underline{p}_j) - \sum_{j \in A_{k+1}^+} \overline{p}_j \cdot \log_2(\overline{p}_j);$$

$$P_{k+1} = P_k - \sum_{j \in A_{k+1}^-} \underline{p}_j + \sum_{j \in A_{k+1}^+} \overline{p}_j; \quad n_{k+1} = n_k - |A_{k+1}^-| + |A_{k+1}^+|.$$

- If $n_k < n$, we compute $p = \dfrac{1 - P_k}{n - n_k}$; if $p \in [p_{(k)}, p_{(k+1)}]$, we compute $S_k = M_k - (n - n_k) \cdot p \cdot \log_2(p)$.
- The largest of these S_k is $\overline{S}$.

16. Towards Linear-Time Algorithm for Computing $\overline{S}$

- *Problem:* sorting requires $O(n\log(n))$ steps.
- *Idea:* use linear-time algorithm for computing median.
- *At each iteration*, we have:

$$J^- = \{\underline{p}_i : \text{ we know that } p_i \neq \underline{p}_i \text{ for optimal } p\} \cup$$
$$\{\overline{p}_i : \text{ we know that } p_j = \overline{p}_i \text{ for optimal } p\};$$
$$J^+ = \{\underline{p}_i : \text{ we know that } p_i = \underline{p}_i \text{ for optimal } p\} \cup$$
$$\{\overline{p}_i : \text{ we know that } p_j \neq \overline{p}_i \text{ for optimal } p\},$$

  and the set J of undecided endpoints.
- In the beginning, $J^- = J^+ = \emptyset$.
- At each iteration we also update the values $N^- = |J^-|$, $N^+ = |J^+|$, $E^- = \sum\limits_{\overline{p}_j \in J^-} \overline{p}_j$, and $E^+ = \sum\limits_{\underline{p}_i \in J^+} \underline{p}_i$.
- Initially, $N^- = N^+ = E^- = E^+ = 0$.

17. Iterations of the New Algorithm

- Compute the median m of the undecided set J.
- Divide J into two sets

$$Q^- = \{p \in J : p \le m\}, \quad Q^+ = \{p \in J : p > m\}.$$

- Compute $m^+ = \min\{p : p \in Q^+\}$.
- Compute $e^- = E^- + \sum\limits_{\overline{p}_j \in Q^-} \overline{p}_j$, $e^+ = E^+ + \sum\limits_{\underline{p}_i \in Q^+} \underline{p}_i$,

$$n^- = N^- + |\{\overline{p}_j \in Q^-\}|, \quad n^+ = N^+ + |\{\underline{p}_i \in Q^+\}|,$$

and $r = \dfrac{1 - e^- - e^+}{n - n^- - n^+}$.

- *We want:* to make sure that $p \in [p_{(k)}, p_{(k+1)}]$.
- *We can check:* whether we are to the left or to the same of this k.

18. Iterations of the New Algorithm (cont-d)

- $r < m$: $J^- \to J^- \cup Q^-$, $E^- \to e^-$, $J \to Q^+$, $N^- \to n^-$.
- If $r > m^+$: $J^+ \to J^+ \cup Q^+$, $E^+ \to e^+$, $J \to P^-$, and $N^+ \to n^+$.
- If $m \le r \le m^+$, set $J \to \emptyset$.
- *Stopping criterion:* $|J| \le 1$.
- *Final stage:* compute $S(p_1, \ldots, p_n)$, where:
  - $p_j = \overline{p}_j$ when $\overline{p}_j \in J^-$,
  - $p_i = \underline{p}_i$ when $\underline{p}_i \in J^+$,
  - $p_i = r$ for $i \in J$.
- *Observation:* each iteration takes time $C \cdot |J|$, and halves the set J.
- *Conclusion:* algorithm takes time

$$C \cdot (2n + n + (n/2) + \ldots) = C \cdot 4n = O(n).$$

19. Computing $\underline{S}$ Is, In General, NP-Hard

- *Algorithms:* several algorithms for computing $\underline{S}$ are known (Abellan et al.)
- *Problem:* in the worst case, these algorithms require time that grows exponentially with n.
- *Explanation:* we prove that computing $\underline{S}$ in general case is, indeed, NP-hard.
- *Main idea of the proof:* we reduce a known NP-hard problem to the problem of computing $\underline{S}$.
- *Reduction:* we use the *subset* problem (known to be NP-hard:)
  - given n positive integers $s_1, \ldots, s_n$,
  - check whether there exist signs $\eta_i \in \{-1, +1\}$ for which the signed sum $\sum\limits_{i=1}^{n} \eta_i \cdot s_i$ is equal to 0.

20. Effective Algorithm for Computing $\underline{S}$ When Intervals Are Not Contained in Each Other

- *Important case of interval uncertainty:* we know all p_i with the same accuracy Δ.
- *Definition:* intervals $[\underline{p}_i, \overline{p}_i]$ satisfy the *subset property* if $[\underline{p}_i, \overline{p}_i] \not\subset (\underline{p}_j, \overline{p}_j)$ for all i and j.
- *Consequences:*
  - when we sort the intervals in lexicographic order, then $\underline{p}_i$ and $\overline{p}_i$ are also sorted;
  - it is sufficient to consider monotonic optimal tuples $p_1, \ldots, p_n$, for which $p_i \le p_{i+1}$ for all i;
  - in the optimal tuple, at most one p_i is inside the corresponding interval
- *Result:* an $O(n \cdot \log(n))$ algorithm that computes $\underline{S}$ for all cases when the subset property holds.

21. Algorithm for Computing $\underline{S}$

- Sort $\mathbf{p}_i$ in lexicographic order: $\mathbf{p}_1 \le_{\text{lex}} \ldots \le_{\text{lex}} \mathbf{p}_n$.
- *Objective:* compute $P_i = \sum\limits_{j:j<i} \underline{p}_j + \sum\limits_{m:m>i} \overline{p}_m$,

$$M_i = -\sum_{j:j<i} \underline{p}_j \cdot \log_2(\underline{p}_j) - \sum_{m:m>i} \overline{p}_m \cdot \log_2(\overline{p}_m).$$

- Compute $M_1 = -\sum\limits_{j=2}^{n} \overline{p}_j \cdot \log_2(\overline{p}_j)$ and $P_1 = \sum\limits_{j=2}^{n} \overline{p}_j$.
- For i from 2 to n, compute $P_i = P_{i-1} + \underline{p}_{i-1} - \overline{p}_i$ and

$$M_i = M_{i-1} - \underline{p}_{i-1} \cdot \log_2(\underline{p}_{i-1}) + \overline{p}_i \cdot \log_2(\overline{p}_i).$$

- Compute $p_i = \dfrac{1 - P_i}{n-1}$.
- If $p_i \in [\underline{p}_i, \overline{p}_i]$, compute $S_i = M_i - p_i \cdot \log_2(p_i)$.
- Return the smallest of these values S_i as $\underline{S}$.
- There is also a linear-time version of this algorithm.

22. Continuous Case: p-Box

- *Situation:* we know that

$$F(x) \in \mathbf{F}(x) = [F_0(x) - \Delta F(x), F_0(x) + \Delta F(x)],$$

where $F_0(x)$ is smooth, with $\rho_0(x) \stackrel{\text{def}}{=} F_0'(x)$.

- *Problem:* find the range $[\underline{S}, \overline{S}] = \{S_\varepsilon(F) : F \in \mathbf{F}\}$.
- *Known result:* asymptotically,

$$\overline{S} \sim -\int \rho_0(x) \cdot \log_2(\rho_0(x))\, dx - \log_2(2\varepsilon).$$

- *New result:*

$$\underline{S} \sim -\int \rho_0(x) \cdot \log_2(\max(2\Delta F(x), 2\varepsilon \cdot \rho_0(x)))\, dx.$$

- *Comment:* when $\varepsilon \to 0$, $\overline{S} \to \infty$ but $\underline{S}$ remains finite:

$$\underline{S} \to -\int \rho_0(x) \cdot \log_2(2\Delta F(x))\, dx.$$

23. Case of a p-Box: Idea of the Proof

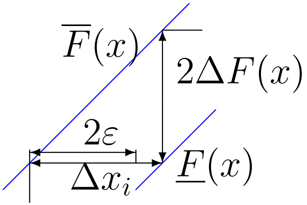
- *Result – reminder:*

$$\underline{S} \sim -\int \rho_0(x) \cdot \log_2(\max(2\Delta F(x), 2\varepsilon \cdot \rho_0(x)))\, dx.$$

- *Idea:* $p_i \approx \rho_0(x_i) \cdot \Delta x_i$, hence

$$-\sum p_i \cdot \log_2(p_i) \approx -\int \rho_0(x) \cdot \log(\rho_0(x) \cdot \Delta x)\, dx.$$

- Here, $\Delta x_i = \max\left(\frac{2\Delta F(x)}{\rho_0(x)}, 2\varepsilon\right)$:



24. Alternative Approach: Idea

- *Previous situation:*
  - we do not know the object,
  - we want to find out how many "yes"-"no" questions we need to find the object x.
- *New situation:*
  - in addition to not knowing the object x,
  - we also do not know the exact probability distribution $\rho(x)$.
- *Solution:*
  - in addition to finding out how many binary questions we need to find x,
  - also find out how many "yes"-"no" questions we need to find the exact probability distribution $\rho(x)$.

25. Alternative Approach: Formulas

- *Objective:* find cdf $F(x)$.
- *Details:* fix two accuracy values:
  - accuracy ε with which we approximate probabilities;
  - accuracy δ with which we approximate x,
- *Case of a p-box – situation:* for every x_0, we have the interval $[\underline{F}(x_0), \overline{F}(x_0)]$. To
- To find $F(x_0)$, we need $\log_2(\overline{F}(x_0) - \underline{F}(x_0)) - \log_2(2\delta)$ questions.
- *Conclusion:* the overall number of questions for all x is

$$\sim \int \log_2(\overline{F}(x) - \underline{F}(x))\,dx = \int \log_2(2\Delta F(x)).$$

26. Adding Fuzzy Uncertainty

- *Crisp case:* we have a (crisp) set P of possible probability distributions (e.g., a p-box).
- In this case, we have information $I(P)$.
- *Fuzzy case:* we have a fuzzy set $\mathcal{P}$ of possible probability distributions.
- *In other words:* we have a family of nested crisp sets $\mathcal{P}(\alpha)$ – α-cuts of the given fuzzy set.
- *Solution:* we define $I(\mathcal{P})$ as a fuzzy number whose α-cut is $I(\mathcal{P}(\alpha))$.
- *Alternative approach:* we can also interpret degree of possibility in probabilistic terms.
- *Alternative solution:* compute the corresponding information by using probability formulas (Ramer et al.).

27. Beyond Number of Questions: How to Measure Loss of Privacy

- *Problem with amount of information:* 1st bit of salary is crucial, last bit is useless.
- *Natural idea:* gauge the loss of privacy by the resulting worst-case financial loss.
- *Example:* the effect of a person's blood pressure x on this person's insurance payments:
  - let $f(x)$ be average medical expenses for a person with blood pressure x; let α be investment profit;
  - in case of privacy, the insurance payments are
  $$r = (1 + \alpha) \cdot E[f(x)];$$
  - if a person's blood pressure is revealed as x_0, with $f(x_0) > E[f(x)]$, then the payments are higher:
  $$r_0 = (1 + \alpha) \cdot f(x_0) > r = (1 + \alpha) \cdot E[f(x)].$$

28. Loss of Privacy: Definition and the Main Result

- *Situation:* we knew that $x \in [L, U]$, now we learned that $x \in [l, u] \subseteq [L, U]$.
- *We knew:* $P \in \mathcal{P} = \{\text{all distributions located on } [L, U]\}$.
- *We know:* $P \in \mathcal{Q} = \{\text{all distributions located on } [l, u]\}$.
- Let $M > 0$. The *amount of privacy* $A(\mathcal{P})$ is the largest value of $F(x_0) - \int \rho(x) \cdot F(x)\, dx$ over:
  - all possible values x_0,
  - all possible probability distributions $\rho \in \mathcal{P}$, and
  - all possible f-s $F(x)$ for which $|F'(x)| \le M$ for all x.
- *Result:* the *relative loss of privacy* $\dfrac{A(\mathcal{P}) - A(\mathcal{Q})}{A(\mathcal{P})}$ is equal to $1 - \dfrac{u - l}{U - L}$.
- *Example:* relative loss of privacy is $1/2$ for the 1st bit, $1/4$ for the 2nd bit, …, $1/2^k$ for the k-th bit, …

29. Acknowledgments

This work was supported in part by:

- by National Science Foundation grants HRD-0734825, EAR-0225670, and EIA-0080940,
- by Texas Department of Transportation grant No. 0-5453, and
- by the conference organizers.