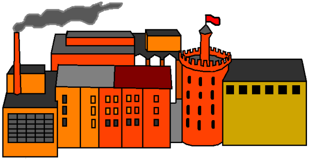
On the (Im)possibility of Privately Outsourcing Linear Programming

26.10.13

Linear programming

- Suppose a brewery produces **ale** and **beer.**
- It uses three type of resources: **corn**, **hops**, and **malt.**
- Each beverage requires particular amount of resources per barrel.

	Ale	**Beer**	**Limit**
Corn	5	15	480
Hops	4	4	160
Malt	35	20	1190
Profit	13	23	



How to maximize the profit having such resource limits?

[Robert G. Bland. The allocation of resources by linear programming. Scientific American, 244(6):108–119, June 1981.]

Linear programming (a bit more formally)

- Let x_1 denote the number of barrels of ale.
- Let x_2 denote the number of barrels of beer.

$$\begin{array}{rrcl} \textbf{maximize} & 13x_1 + 23x_2 & & \\ \textbf{subject to} & 5x_1 + 15x_2 & \leq & 480 \\ & 4x_1 + 4x_2 & \leq & 160 \\ & 35x_1 + 20x_2 & \leq & 1190 \\ & x_1 & \geq & 0 \\ & x_2 & \geq & 0 \end{array}$$

	Ale	**Beer**	**Limit**
Corn	5	15	480
Hops	4	4	160
Malt	35	20	1190
Profit	13	23	

Linear programming (formally)

The same task in a matrix form:

$$\textbf{maximize} \quad \begin{pmatrix} 13 \\ 23 \end{pmatrix}^{\mathrm{T}} \cdot \begin{pmatrix} x_1 \\ x_2 \end{pmatrix},$$

$$\textbf{subject to} \quad \begin{pmatrix} 5 & 15 \\ 4 & 4 \\ 35 & 20 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \leq \begin{pmatrix} 480 \\ 160 \\ 1190 \end{pmatrix}, \quad \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \geq \begin{pmatrix} 0 \\ 0 \end{pmatrix} \ .$$

where $\leq$ is defined coordinatewise.

Linear programming (formally)

The same task in a matrix form:

$$\textbf{maximize} \quad \begin{pmatrix} 13 \\ 23 \end{pmatrix}^{\mathrm{T}} \cdot \begin{pmatrix} x_1 \\ x_2 \end{pmatrix},$$

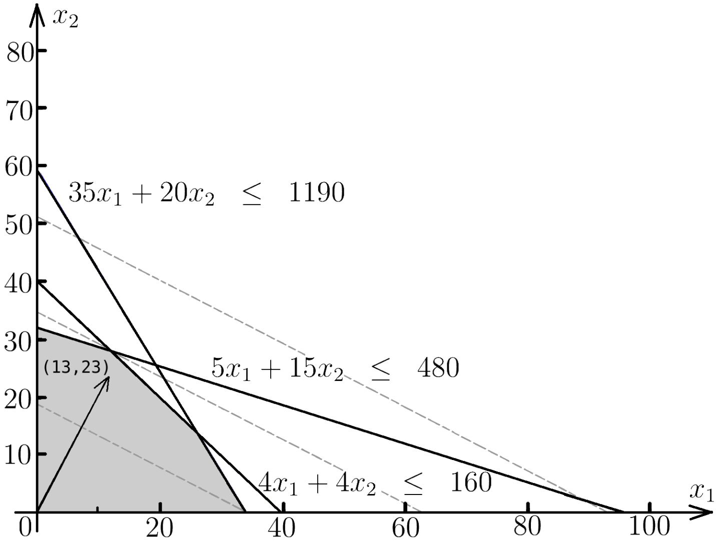
$$\textbf{subject to} \quad \begin{pmatrix} 5 & 15 \\ 4 & 4 \\ 35 & 20 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \leq \begin{pmatrix} 480 \\ 160 \\ 1190 \end{pmatrix}, \quad \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \geq \begin{pmatrix} 0 \\ 0 \end{pmatrix} .$$

where $\leq$ is defined coordinatewise.

Canonical form:

$$\text{maximize } \mathbf{c}^{\mathrm{T}} \cdot \mathbf{x}, \text{ subject to } \mathrm{A}\mathbf{x} \leq \mathbf{b}, \mathbf{x} \geq \mathbf{0}$$

Feasible region of a linear program



Privacy-preserving linear programming

Solve a linear programming task:

$$\text{maximize } \mathbf{c}^{\mathrm{T}} \cdot \mathbf{x}, \text{ subject to } A\mathbf{x} \leq \mathbf{b}, \mathbf{x} \geq \mathbf{0} \quad ,$$

Privacy-preserving linear programming

Solve a linear programming task:

$$\text{maximize } \mathbf{c}^{\mathrm{T}} \cdot \mathbf{x}, \text{ subject to } \mathrm{A}\mathbf{x} \leq \mathbf{b}, \mathbf{x} \geq \mathbf{0} \quad ,$$

where the quantities A, $\mathbf{b}$, $\mathbf{c}$ are distributed amongst several parties.

Privacy-preserving linear programming

Solve a linear programming task:

$$\text{maximize } \mathbf{c}^{\mathrm{T}} \cdot \mathbf{x}, \text{ subject to } A\mathbf{x} \leq \mathbf{b}, \mathbf{x} \geq \mathbf{0} \quad ,$$

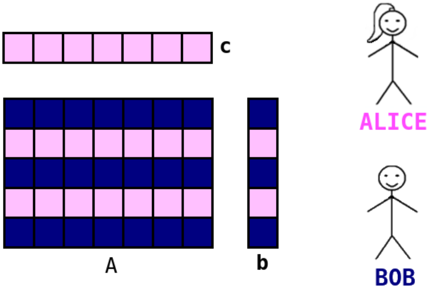
where the quantities A, **b**, **c** are distributed amongst several parties.

Privacy-preserving linear programming

Solve a linear programming task:

$$\text{maximize } \mathbf{c}^{\mathrm{T}} \cdot \mathbf{x}, \text{ subject to } A\mathbf{x} \leq \mathbf{b}, \mathbf{x} \geq \mathbf{0} \quad ,$$

where the quantities A, $\mathbf{b}$, $\mathbf{c}$ are distributed amongst several parties.



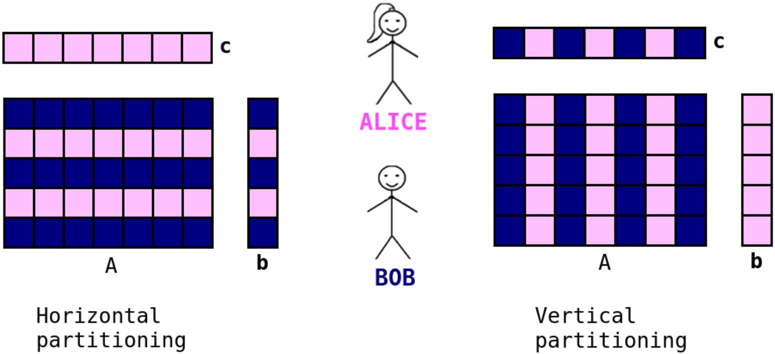
Horizontal
partitioning

Privacy-preserving linear programming

Solve a linear programming task:

$$\text{maximize } \mathbf{c}^{\mathrm{T}} \cdot \mathbf{x}, \text{ subject to } A\mathbf{x} \leq \mathbf{b}, \mathbf{x} \geq \mathbf{0} \; ,$$

where the quantities A, **b**, **c** are distributed amongst several parties.

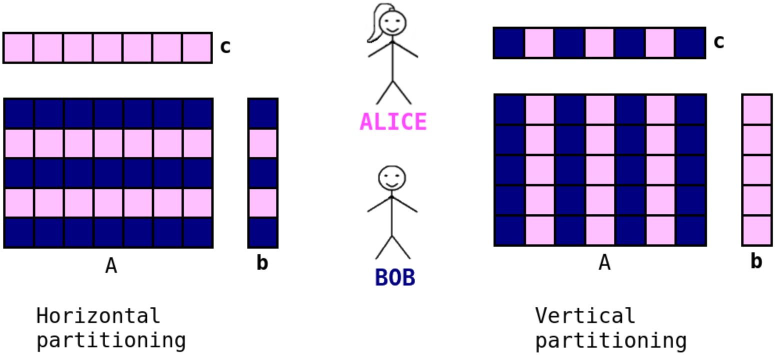


Privacy-preserving linear programming

Solve a linear programming task:

$$\text{maximize } \mathbf{c}^{\mathrm{T}} \cdot \mathbf{x}, \text{ subject to } A\mathbf{x} \leq \mathbf{b}, \mathbf{x} \geq \mathbf{0} \quad ,$$

where the quantities A, **b**, **c** are distributed amongst several parties.



No information about A, **b**, **c** should be leaked in the computational process.

Two main approaches

Two main approaches

1. **Straightforward:** implement directly a linear programming solving algorithm by computing the basic operations in a cryptographic way.

Two main approaches

1. **Straightforward:** implement directly a linear programming solving algorithm by computing the basic operations in a cryptographic way.

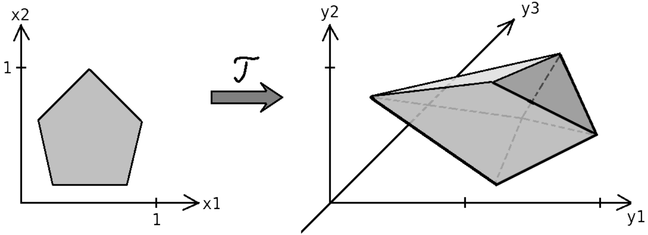
   Always possible, but too inefficient.

Two main approaches

1. **Straightforward:** implement directly a linear programming solving algorithm by computing the basic operations in a cryptographic way.

   Always possible, but too inefficient.

2. **Transformation-based:** transform the program to another linear program so that it may be solved offline without leaking information about the initial program.

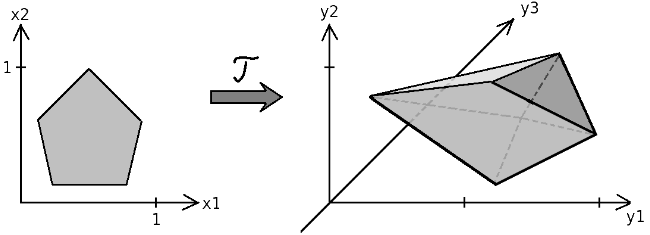


Two main approaches

1. **Straightforward:** implement directly a linear programming solving algorithm by computing the basic operations in a cryptographic way.

   Always possible, but too inefficient.

2. **Transformation-based:** transform the program to another linear program so that it may be solved offline without leaking information about the initial program.

   Much more efficient.

Acceptable security

Definition

A protocol achieves acceptable security if the only thing that the adversary can do is to reduce all the possible values of the secret data to some domain with the following properties:

1. The number of values in this domain is infinite, or the number of values in this domain is so large that a brute-force attack is computationally infeasible.
2. The range of the domain (the difference between the upper and the lower bounds) is acceptable for the application.

[Du & Zhan, New Security Paradigms Workshop 2002]

Problems of the acceptable security definition

- Non-standard and cannot therefore be integrated into complex protocols.

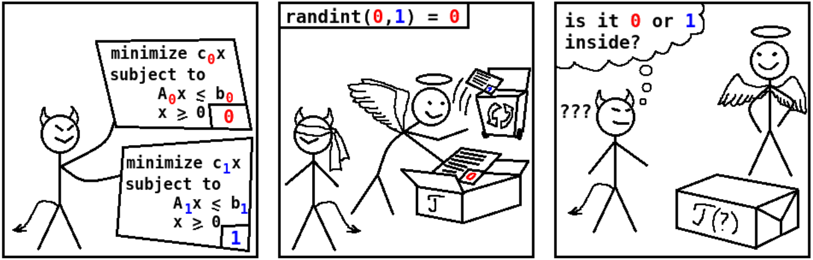
Problems of the acceptable security definition

- Non-standard and cannot therefore be integrated into complex protocols.
- Makes the scheme too dependent on the initial sharing of A,**b**,**c**.

Problems of the acceptable security definition

- Non-standard and cannot therefore be integrated into complex protocols.
- Makes the scheme too dependent on the initial sharing of A,**b**,**c**.
- Too weak. Some attacks have been found against the schemes that were assumed to be secure under this definition.

Indistinguishability-based security definition



The adversary creates two instances of linear programming tasks.

The environment applies transformation $\mathcal{T}$ to one of these two tasks, chosen randomly.

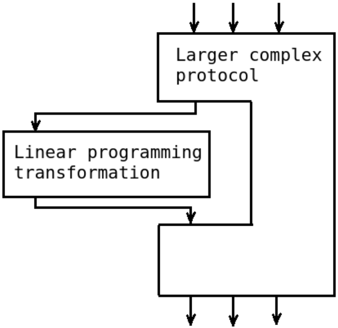
The adversary sees the transformation result and attempts to guess what it has been.

Why this definition is good

- Makes the linear program independent on the initial sharing.

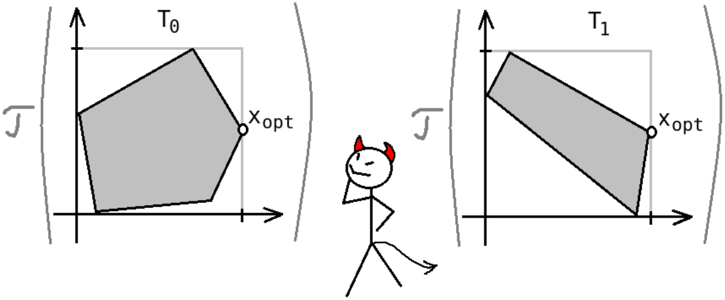
Why this definition is good

- Makes the linear program independent on the initial sharing.
- Is sufficiently standard to be integrated into more complex protocols.



Acceptable Side Information

- It is reasonable to weaken the security definition so that only LP tasks with certain properties are indistinguishable after the transformation:
  - have the same bounding box;
  - have the same feasible solution.

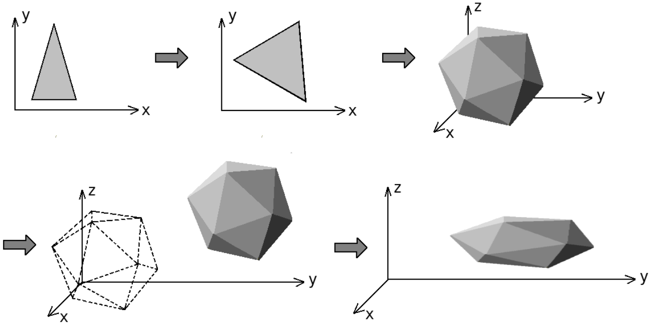


Affine transformations

- The transformation-based methods map a linear program to another linear program.

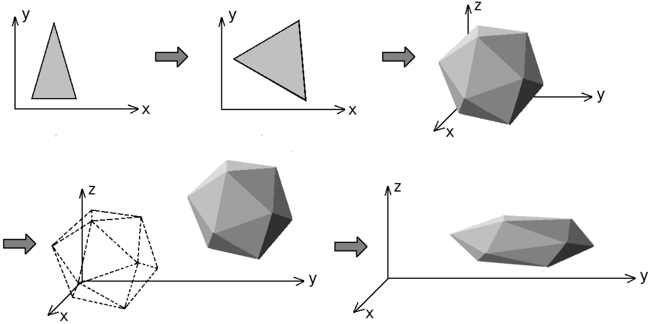
Affine transformations

- The transformation-based methods map a linear program to another linear program.
- The known transformations used in the related work belong to the class of affine transformations.



Affine transformations

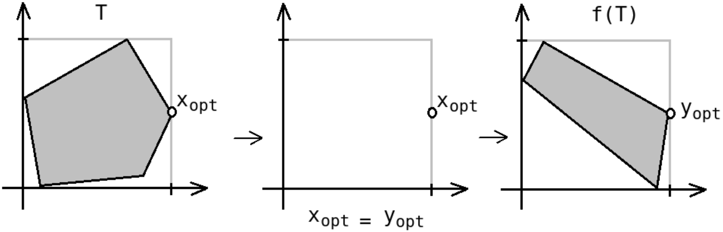
- The transformation-based methods map a linear program to another linear program.
- The known transformations used in the related work belong to the class of affine transformations.



- We will show that this approach may quite unlikely be successful.

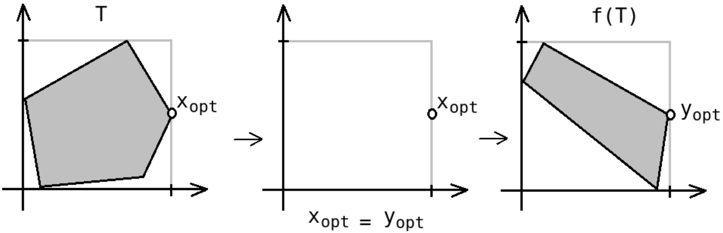
Perfect Secrecy

- A transformation with perfect secrecy is definitely possible.



Perfect Secrecy

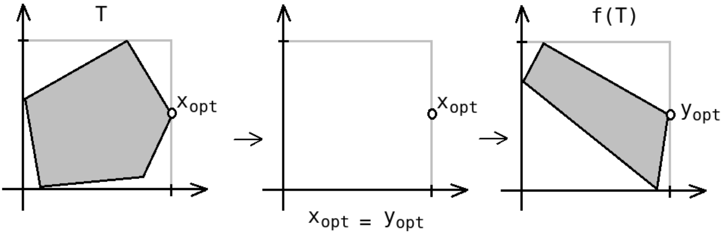
- A transformation with perfect secrecy is definitely possible.



- The problem is that the transformation should be no more complex than solving the linear program itself.

Perfect Secrecy

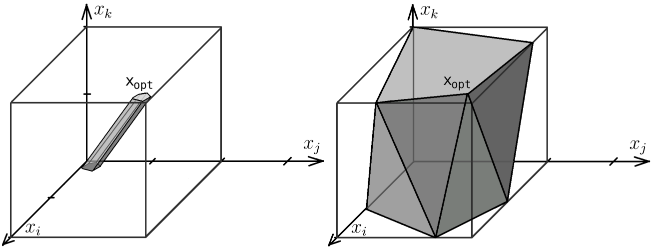
- A transformation with perfect secrecy is definitely possible.



- The problem is that the transformation should be no more complex than solving the linear program itself.
- In the case of affine functions such that y_{opt} is continuous with respect to x_{opt}, a perfectly secure transformation allows to find optimal solutions in a large class of linear programs solving just one instance.

A Requirement of Perfect Secrecy

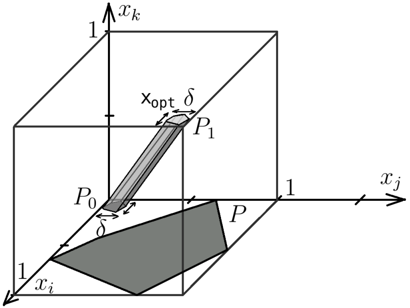
- According to our definition, the following programs have to be indistinguishable.



- Hence the distribution of distances between the hyperplanes of a transformed program should not depend on the distances between the hyperplanes of the initial program.

Preprocessing

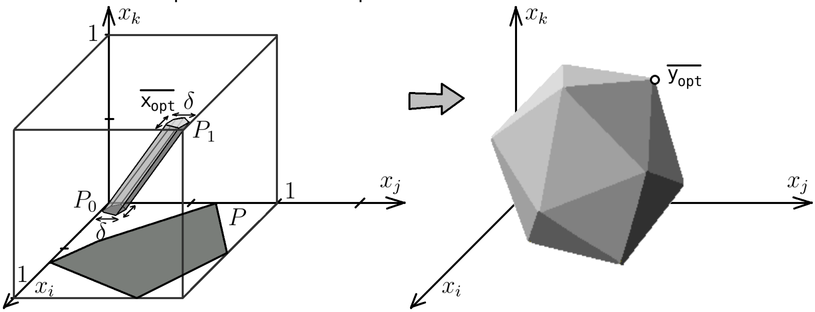
- An arbitrary $n-1$ dimensional polyhedron with $m-2$ facets can be scaled to a bounding box of size at most δ and then extended to an n-dimensional m-facet hyperprism as follows:



- We are interested in the optimal solution $\mathrm{x_{opt}}$ that is closer to the point $(1, 1, \ldots, 1)$.

Preprocesing

- Let $\overline{\mathsf{x}_{\mathsf{opt}}}$ be a known solution to some LP with parameters $n-1$, $m-2$ modified in this way. Let its transformed solution be $\overline{\mathsf{y}_{\mathsf{opt}}}$. Suppose $\overline{\mathsf{y}_{\mathsf{opt}}}$ is known.



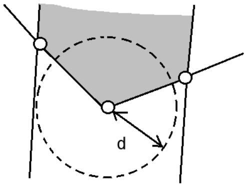
- We show how to find an optimal solution for an arbitrary LP with parameters $n-1$, $m-2$.

No Perfect Secrecy

- First, scale the LP to δ and form a hyperprism as before. Let x_{opt} be the optimal solution. Clearly, $\|\overline{x_{opt}} - x_{opt}\| < \delta$.
- Due to continuity

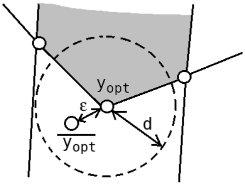
$$\forall \varepsilon > 0\ \exists \delta > 0 : \|\overline{x_{opt}} - x_{opt}\| < \delta \implies \|\overline{y_{opt}} - y_{opt}\| < \varepsilon$$

- Due to perfect secrecy, for a certain *d* that does not depend on δ, any vertex of the transformed program is located at the distance at least *d* from the hyperplanes that do not contain this vertex.



No Perfect Secrecy

- If we take $\varepsilon < d/2$, then there is exactly one vertex at the distance at most ε from $\overline{\mathrm{y}_{\mathrm{opt}}}$, and this is the $\mathrm{y}_{\mathrm{opt}}$.



- Hence it suffices to find the intersection of the bounding hyperplanes that are at the distance of at most ε from the $\overline{\mathrm{y}_{\mathrm{opt}}}$.
- This is much easier than solving the linear programming task itself.

Requirements of Computational Security

- Some assumptions similar to the finite fields could be defined over real numbers.

Requirements of Computational Security

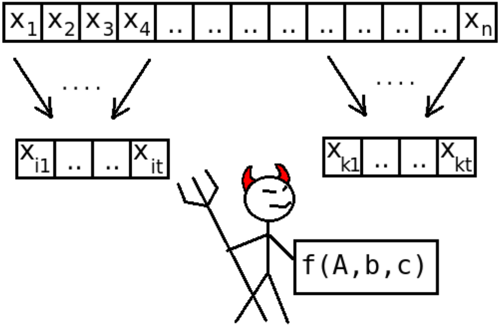
- Some assumptions similar to the finite fields could be defined over real numbers.
- We have tried different means of hiding:
  - Adding more columns (and hence more variables)
  - Adding more rows (and hence more constraints)
  - Splitting the variables

Requirements of Computational Security

- Some assumptions similar to the finite fields could be defined over real numbers.
- We have tried different means of hiding:
  - Adding more columns (and hence more variables)
  - Adding more rows (and hence more constraints)
  - Splitting the variables
- In all experiments, we have failed for the same reason: different types of variables behave in different ways.

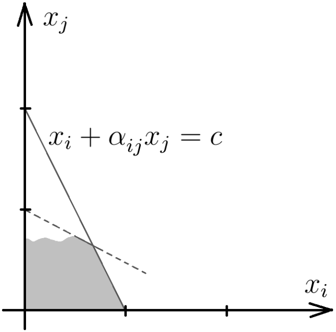
Requirements of Computational Security

- Hence we empirically state the requirement:
- Any set of t variables (where t is a security parameter) should look the same for the adversary who has access to the transformed linear program.



2-symmetric transformations

- In order to achive security for any t, we need to achieve it for at least $t = 2$.
- In a 2-dimensional projection, computing the angle between the bounds and the axes is easy.



Hence we require that for each pair of variables (x_i, x_j), it must hold that $x_i + \alpha x_j = c$.

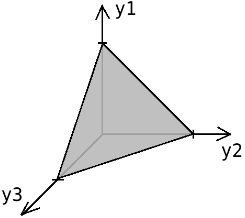
No natural way to achieve it

- Since all the angles should be the same, we get the following system:

$$\begin{cases} \alpha x_1 + x_2 + a_{123}x_3 + \ldots + a_{12(n-1)}x_{n-1} + a_{12n}x_n = c \\ x_1 + \alpha x_2 + a_{213}x_3 + \ldots + a_{21(n-1)}x_{n-1} + a_{21n}x_n = c \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ a_{n(n-1)1}x_1 + a_{n(n-1)2}x_2 + \ldots + \alpha x_{n-1} + x_n = c \\ a_{(n-1)n1}x_1 + a_{(n-1)n2}x_2 + \ldots + x_{n-1} + \alpha x_n = c \end{cases}$$

- Solving it, we get that the polyhedron is a simplex.

$$\sum_{i=1}^{n} y_i = c$$



- Such a transformation has too low degree of freedom to encode something reasonable.

Conclusion

- The current approaches towards privacy-preserving outsourcing or multiparty linear programming are unlikely to be successful.
- Success in this direction requires some radically new ideas violating our rather generous assumptions.
- Alternatively, it may be fruitful to optimize privacy-preserving implementations of LP solving algorithms in order to have universal privacy-preserving optimization methods for large classes of tasks.

THE END