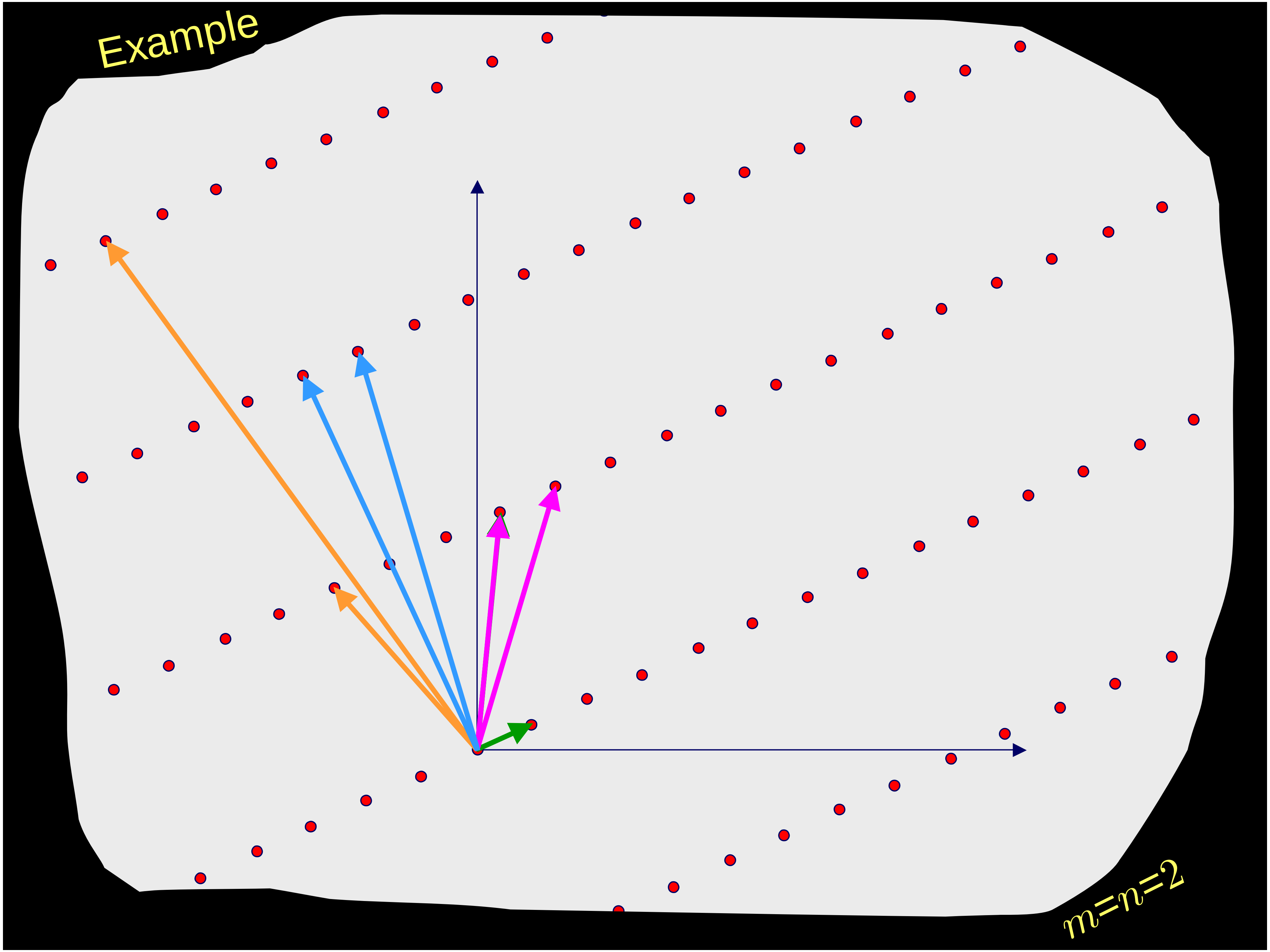
Lattices in Cryptography #1

Lattice

- Discrete subgroup of $\mathbb{R}^n$
- Linear combinations with integer coefficients of vectors in $\mathbb{R}^n$:

$$L = \{x_1\vec{v}_1 + x_2\vec{v}_2 + \cdots + x_m\vec{v}_n \ : \ x_1, x_2, \ldots, x_m \in \mathbb{Z}\}$$

- Such a set of vectors generates the lattice. If it is linearly independent, it forms a basis.
- Every lattice has infinitely many bases (except...)



Example
$m=n=2$

Lattice problems

- Shortest vector problem (SVP):

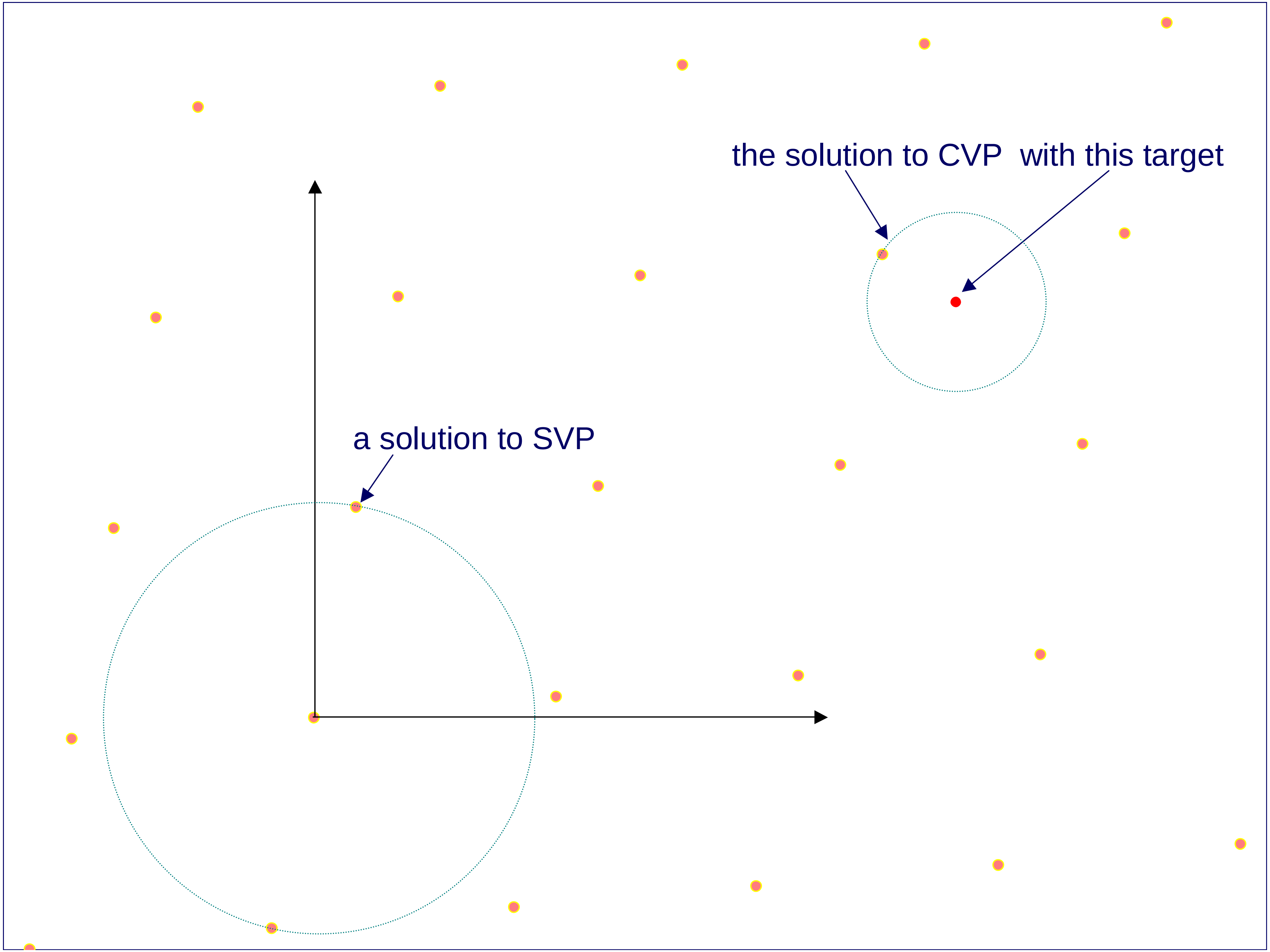
  Given a basis for a lattice L, find a shortest nonzero element in L under a given norm (usually l_2).

$$\vec{u} \quad : \quad \vec{u} \in L \setminus \{\vec{0}\}\,,\ \forall \vec{v} \in L : \|\vec{u}\| \leq \|\vec{v}\|$$

- Closest vector problem (CVP):

  Given a basis for a lattice $L \subseteq \mathbb{R}^n$ and a target vector $\vec{t} \subseteq \mathbb{R}^n$, find the lattice vector closest to $\vec{t}$.

$$\vec{u} \quad : \quad \vec{u} \in L\,,\ \forall \vec{v} \in L : \|\vec{u} - \vec{t}\| \leq \|\vec{v} - \vec{t}\|$$


the solution to CVP
with this target
a solution to SVP

Complexity of SVP, CVP

Finding a vector that is at most γ times longer than the shortest vector.

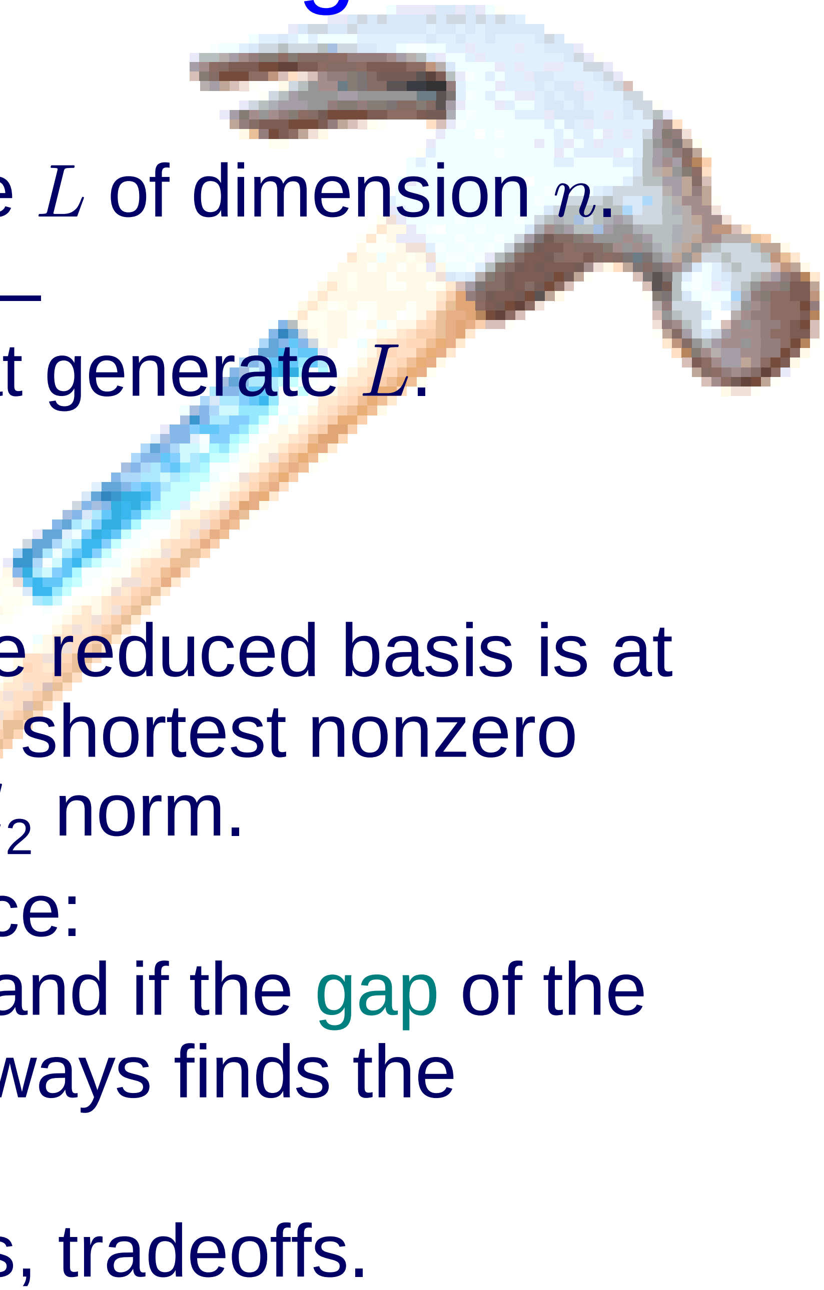
- Approximating SVP for the l_p norm within factor $\gamma=2^{1/p}$ is NP-hard
  (with randomized reductions or some assumptioms)
  but is unlikely to be NP-hard for $\gamma= n^{1/2}$.
- Approximating CVP within polylogarithmic factor $\gamma= \log^c n$ is NP-hard (for any l_p norm).

Finding a vector that is at most γ times farther from the target than the closest vector.

- All known algorithms have exponential approximation ratios or run in exponential time.

The LLL lattice reduction algorithm

[Lenstra,Lenstra,Lovász 1982]



- Input: a basis for a lattice L of dimension n.
- Output: a reduced basis – a set of short vectors that generate L.
- Runs in polynomial time.
- Proven performance: The shortest vector in the reduced basis is at most $2^{n/2}$ longer than the shortest nonzero lattice vector, under the l_2 norm.
- Experimental performance: For reasonably small n, and if the gap of the lattice is large, almost always finds the shortest vector.
- Many variants: speedups, tradeoffs.

Solving low-density knapsacks

- Consider the following knapsack problem: given s,a_1,a_2,…,a_n find x_1,x_2,…,x_n such that $\sum_i x_i a_i = s$.
- Density of the knapsack problem: $d = n/m$ where $m = \max\{\log_2 a_i\}$.
- Random knapsacks with d<0.9408 can be efficiently reduced to SVP. [Coster, Joux et al., 1991]
- Will show: breaking random knapsacks with d<$1/n$ by reduction to SVP.

[Lagarias, Odlyzko 1983][Frieze 19876]

Low-density knapsacks – the lattice

$w = n 2^{n/2}$

$$\left.\begin{array}{c|c|c|c|c|c}
ws & 0 & 0 & 0 & \cdots & 0 \\ \hline
-wa_1 & 1 & 0 & 0 & \cdots & 0 \\ \hline
-wa_2 & 0 & 1 & 0 & \cdots & 0 \\ \hline
-wa_3 & 0 & 0 & 1 & \cdots & 0 \\ \hline
\vdots & \vdots & \vdots & \vdots & \ddots & 0 \\ \hline
-wa_n & 0 & 0 & 0 & \cdots & 1
\end{array}\right\} \text{vectors generating the lattice } L$$

Algorithm:

1. Use LLL to find the shortest vector in L.
2. Rejoice.

For the solution vector:

$$\begin{pmatrix} 1 \\ x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{pmatrix}^T \times \begin{pmatrix} ws & 0 & 0 & 0 & \cdots & 0 \\ -wa_1 & 1 & 0 & 0 & \cdots & 0 \\ -wa_2 & 0 & 1 & 0 & \cdots & 0 \\ -wa_3 & 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & 0 \\ -wa_n & 0 & 0 & 0 & \cdots & 1 \end{pmatrix}$$

$$= \begin{pmatrix} w0 & x_1 & x_2 & x_3 & \cdots & x_n \end{pmatrix}$$

$$\text{norm}_2 \leq \sqrt{n}$$

Bad vectors, case #1:

$$\begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_n \end{bmatrix}^T \times \begin{bmatrix} ws & 0 & 0 & 0 & & \\ -wa_1 & 1 & 0 & 0 & & \\ -wa_2 & 0 & 1 & 0 & & \\ -wa_3 & 0 & 0 & 1 & & \\ \vdots & \vdots & \vdots & \vdots & & \\ -wa_n & 0 & 0 & 0 & \cdots & 1 \end{bmatrix}$$

$$= \begin{bmatrix} \underbrace{w(\cdots)}_{\neq 0} & y_1 & y_2 & y_3 & \cdots & y_n \end{bmatrix}$$

$\text{norm}_2 \geq w = n\sqrt{2^n}$

LLL finds a vector that is at most $2^{n/2}$ times longer than the shortest lattice vector, whose length here is at most $n^{1/2}$. Thus, LLL will never return this type of bad vectors.

Bad vectors, case #2:

$$\begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_n \end{pmatrix}^T \times \begin{pmatrix} ws & 0 & 0 & 0 & & \\ -wa_1 & 1 & 0 & & & \\ -wa_2 & 0 & 1 & & & \\ -wa_3 & 0 & 0 & & & \\ \vdots & \vdots & \vdots & \vdots & & \\ -wa_n & 0 & 0 & 0 & \cdots & 1 \end{pmatrix} = \begin{pmatrix} 0 & y_1 & y_2 & y_3 & \cdots & y_n \end{pmatrix}$$

$$\neq k \cdot \begin{pmatrix} 0 & x_1 & x_2 & x_3 & \cdots & x_n \end{pmatrix} \quad (\forall k \in \mathbb{Z})$$

A multiple of $(x_1,\ldots,x_n)$ is OK.

However, we may get a short vector with coefficients outside {0,1} that does not correspond to a solution to the knapsack.

Bad vectors, case #2 (cont.)

Fix a arbitrary nonzero solution vector $\vec{x} = (x_1, \ldots, x_n) \in \{0,1\}^n$.

Definition 1. A vector $\vec{y} \in \mathbb{Z}^n$ is <u>bad</u> for knapsack weights $\vec{a} = (a_1, \ldots, a_n)$ if:

(1) $(0, \vec{y}) \in L_{\vec{a}}$
(2) $\|\vec{y}\| \leq \sqrt{n2^n}$
(3) $\forall k \in \mathbb{Z} : \ \vec{y} \neq k\vec{x}$

Goal: bound the probability that a random knapsack has <u>any</u> bad vector.

Bad vectors, case #2 (cont.)

Fix a arbitrary nonzero solution vector $\vec{x} = (x_1, \ldots, x_n) \in \{0,1\}^n$.

Definition 1. A vector $\vec{y} \in \mathbb{Z}^n$ is <u>bad</u> for knapsack weights $\vec{a} = (a_1, \ldots, a_n)$ if:

(1) $(0, \vec{y}) \in L_{\vec{a}}$
(2) $\|\vec{y}\| \leq \sqrt{n2^n}$
(3) $\forall k \in \mathbb{Z} : \ \vec{y} \neq k\vec{x}$

Lemma 1. *If $\vec{y}$ is bad for knapsack weights $\vec{a}$ then there exists $y_0 \in Z$ such that*

(4) $|y_0| \leq 2\sqrt{n2^n}$
(5) $\sum_{i=1}^n y_i a_i = y_0 \sum_{i=1}^n x_i a_i$

Proof. Let $s = \sum_{i=1}^n x_i a_i$ and let $y_0 = \sum_{i=1}^n y_i a_i / s$. By (1), $y_0 \in \mathbb{Z}$. (5) holds since both sides equal $y_0 s$. We have

$$|s y_0| = \left| \sum_{i=1}^n y_i a_i \right| \leq \left| \sum_{i=1}^n \|\vec{y}\| \, a_i \right| = \|\vec{y}\| \sum_{i=1}^n a_i \overset{(*)}{\leq} \|\vec{y}\| \, 2s$$

where $(*)$ holds because w.l.o.g., $s \geq \frac{1}{2} \sum_{i=1}^n a_i$. Thus $|y_0| \leq 2 \|\vec{y}\| \overset{(2)}{\leq} \sqrt{n2^n}$. □

Bad vectors, case #2 (cont.)

Fix a arbitrary nonzero solution vector $\vec{x} = (x_1, \ldots, x_n) \in \{0,1\}^n$.

Definition 1. A vector $\vec{y} \in \mathbb{Z}^n$ is <u>bad</u> for knapsack weights $\vec{a} = (a_1, \ldots, a_n)$ if:

(1) $(0, \vec{y}) \in L_{\vec{a}}$
(2) $\|\vec{y}\| \leq \sqrt{n2^n}$
(3) $\forall k \in \mathbb{Z}: \ \vec{y} \neq k\vec{x}$

Lemma 1. *If $\vec{y}$ is bad for knapsack weights $\vec{a}$ then there exists $y_0 \in Z$ such that*

(4) $|y_0| \leq 2\sqrt{n2^n}$
(5) $\sum_{i=1}^n y_i a_i = y_0 \sum_{i=1}^n x_i a_i$

Lemma 2. *For any fixed $y_0 \in Z$ and $\vec{y} \in \mathbb{Z}^n$ fulfilling (3), if $\vec{a}$ is drawn randomly from $\{0, \ldots, b\}^n$ then the probability that $\vec{y}$ and y_0 fulfill (5) is at most $1/b$.*

Proof. Let $z_i = y_i - x_i a_i$. Then (5) is equivalent to $\sum_{i=1}^n z_i a_i = 0$. By (3), there exists some nonzero z_j. $\Pr_{\vec{a}}[(5)] = \Pr_{\vec{a}}\left[z_i a_i = -\sum_{i \neq j} z_i a_i\right] \leq 1/\text{b}$ by independence. $\square$

⋮ Fix a arbitrary nonzero solution vector $\vec{x} = (x_1, \ldots, x_n) \in \{0,1\}^n$.

Definition 1. A vector $\vec{y} \in \mathbb{Z}^n$ is <u>bad</u> for knapsack weights $\vec{a} = (a_1, \ldots, a_n)$ if:

(1) $(0, \vec{y}) \in L_{\vec{a}}$
(2) $\|\vec{y}\| \leq \sqrt{n2^n}$
(3) $\forall k \in \mathbb{Z} : \vec{y} \neq k\vec{x}$

Corollary. *If the knapsack weights $\vec{a}$ are drawn from $\{0, \ldots, 2^{n^2}\}$ then the probability that there exists a bad vector for $\vec{a}$ is negligible.*

Lemma 1. *If $\vec{y}$ is bad for knapsack weights $\vec{a}$ then there exists $y_0 \in Z$ such that*

(4) $|y_0| \leq 2\sqrt{n2^n}$
(5) $\sum_{i=1}^n y_i a_i = y_0 \sum_{i=1}^n x_i a_i$

Lemma 2. *For any fixed $y_0 \in Z$ and $\vec{y} \in \mathbb{Z}^n$ fulfilling (3), if $\vec{a}$ is drawn randomly from $\{0, \ldots, b\}^n$ then the probability that $\vec{y}$ and y_0 fulfill (5) is at most $1/b$.*

Lemma 3. *If $\vec{a}$ is drawn randomly from $\{0, \ldots, b\}^n$, the probability that there exists a bad $\vec{y}$ for $\vec{a}$ is at most $2^{(1/2+o(1))n^2}/b$.*

Proof. There are $\left(4\sqrt{n2^n}+1\right)$ choices of y_0 that fulfill (4) and at most $\left(2\sqrt{n2^n}+1\right)^n$ choices of $\vec{y}$ that fulfill (2). Thus:

$$\begin{aligned}
\Pr_{\vec{a}}[\exists \text{bad } \vec{y}] &\leq \Pr_{\vec{a}}[\exists y, \vec{y}_0 : (2)(3)(4)(5)] \\
&\leq \left(2\sqrt{n2^n}+1\right)^n \left(4\sqrt{n2^n}+1\right) \underbrace{\max_{\vec{y}, y_0}\left\{\Pr_{\vec{a}}[(3)(5)]\right\}}_{\leq 1/b \text{ by Lemma 2}}
\end{aligned}$$

Low-density knapsacks - conclusion

- Even though the LLL algorithm provides only an exponential approximation, it can provably solve most knapsacks with density $d \leq n/log_2 2^{n^2} = 1/n$.
- In practice, LLL and variants thereof perform much better than the proven bounds, and can be used to solve knapsacks with much higher density.

Factoring using lattices [Schnorr 1993]

- To factorize a composite n with high probability, find "random" x,y such that $x^2 \equiv y^2 (\bmod n)$
- The Morrison-Brillhart recipe: find smooth numbers and combine their exponent vectors. In this case:
  - Consider the t primes smaller than B.

1. Find $2t+1$ pairs (u_i,v_i) such that both u_i and $(u_i - v_i n)$ are B-smooth:
$$u_i = \prod_{j=1}^{t} p_j^{a_{i,j}} \;, \qquad (u_i - v_i n) = \prod_{j=1}^{t} p_j^{b_{i,j}}$$
2. Find a subset S such that
$$\forall j : \sum_{i \in S} a_{i,j} \equiv 0 \;, \qquad \sum_{i \in S} b_{i,j} \equiv 0 \pmod 2$$
3. We get two squares over $\mathbb{Z}$. Extract their square roots:
$$\prod_{i \in S} u_i = x^2 \;, \qquad \prod_{i \in S} (u_i - v_i n) = y^2$$

Factoring using lattices – variant

1. Find only t+1 pairs (u_i,v_i) such that both u_i and u_i- $v_i n$ are B-smooth:

$$u_i = \prod_{j=1}^{t} p_j^{a_{i,j}}\ , \qquad |u_i - v_i n| = \prod_{j=1}^{t} p_j^{b_{i,j}}$$

2. Find a subset S such that

$$\forall j : \sum_{i \in S} (a_{i,j} + b_{i,j}) = 0 \pmod 2$$

3. Now:

$$y = \prod_j p_j^{\sum_{i\in S} a_{i,j}} = \prod_{i \in S} u_i$$

$$y' = \prod_j p_j^{\sum_{i\in S} b_{i,j}} = \prod_{i \in S} (u_i - v_i n) \equiv y \pmod n$$

$$x = \prod_j p_j^{\sum_{i\in S} (a_{i,j} + b_{i,j})/2} \equiv \sqrt{y \cdot y'} \pmod n$$

$$\Rightarrow \qquad \boxed{x^2 \equiv y \cdot y' \equiv y^2 \pmod n}$$

Closest-vector problem for factoring:

$$
\begin{bmatrix} e_1 \\ e_2 \\ e_3 \\ e_4 \\ \vdots \\ e_t \end{bmatrix}^T
\times
\begin{bmatrix}
\log p_1 & 0 & 0 & 0 & \cdots & 0 & w \log 2 \\
0 & \log p_2 & 0 & 0 & \cdots & 0 & w \log 3 \\
0 & 0 & \log p_3 & 0 & \cdots & 0 & w \log 5 \\
0 & 0 & 0 & \log p_4 & \cdots & 0 & w \log 7 \\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & 0 & 0 & \cdots & \log p_t & w \log p_t
\end{bmatrix}
$$

$$
= \begin{bmatrix} * & * & * & * & \cdots & * & * \end{bmatrix}
$$

$$
\approx \begin{bmatrix} 0 & 0 & 0 & 0 & \cdots & 0 & w \log n \end{bmatrix} \quad \vec{t}
$$

Factoring using lattices (cont.)

- How to find many pairs (u_i,v_i) such that both u_i and u_i- $v_i n$ are smooth over the first t primes?

- Find very good CVP solutions in l_1 norm, that is: |/////|$_1$ close to 0, ///// close to to $w$log$n$.
- Set $u = \prod_{e_j>0} p_j^{e_j}\ ,\ v = \prod_{e_j<0} p_j^{-e_j}$

$$
\begin{pmatrix} e_1 & e_2 & e_3 & e_4 & \cdots & e_t \end{pmatrix} \times
\begin{pmatrix}
\log p_1 & 0 & 0 & 0 & \cdots & 0 & w\log 2 \\
0 & \log p_2 & 0 & 0 & \cdots & 0 & w\log 3 \\
0 & 0 & \log p_3 & 0 & \cdots & 0 & w\log 5 \\
0 & 0 & 0 & \log p_4 & \cdots & 0 & w\log 7 \\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & 0 & 0 & \cdots & \log p_t & w\log p_t
\end{pmatrix}
$$

$$
= \begin{pmatrix} * & * & * & * & \cdots & * & * \end{pmatrix} \quad \vec{v}
$$

$$
\approx \begin{pmatrix} 0 & 0 & 0 & 0 & \cdots & 0 & w\log n \end{pmatrix} \quad \vec{t}
$$

- ε > |///// - $w$log$n$|

$$\Longrightarrow \varepsilon/w > \left|\sum_{i=1}^{t} e_i \log p_i - \log n\right|$$

$$= |\log(u/vN)| = \left|\log\left(1+\frac{u-vn}{vn}\right)\right| \approx \left|\frac{u-vn}{vn}\right|$$

$$\Longrightarrow |u-vn| < vn\varepsilon/w$$

- |/////|$_1$ is small, so |u-vn| is small $\Rightarrow$ likely to be smooth.

Factoring using lattices (cont.)

- (Verify that there are enough short vectors.)
- Using an efficient algorithm for the CVP problem in l_1 with sufficiently good approximation, we can factor integers.
- With known lattice algorithms: impractical.