Double Matrix Algorithm (Task 2.3)
Participants

- Paul Zimmermann (task leader)
- Cécile Pierrot
- Charles Bouillaguet
- Ambroise Fleury?
- Post-doc to be hired (LIP6)?
References

Cf git repo, biblio/dble_matrix
Slides from Thorsten Kleinjung at WCNT 2011.
“Mersenne factory” paper, 2014.
Slides from Emmanuel and Pierrick, 2015.
Antoine Joux ?
Abstract: “Most factorizations used a new double-product approach that led to additional savings in the matrix step.”

Page 14: “Details about the new filtering strategy will be provided once we have more experience with it.”
The Idea

Let $M$ be the matrix at the end of “purge” (singleton removal + “clique” algorithm).

Each row of $M$ consists of a relation, and each column correspond to an ideal.

The “merge” step (Structured Gaussian Elimination) combines rows to eliminate columns:

$$PM = M'$$

The linear algebra step computes (left) matrix-vector products $vM'$. Instead, we can compute $w = vP$ and then $wM$.

If the cumulated cost of $vP$ and $wM$ is less than that of $vM'$, we win!
Some figures

RSA-250: $M$ has 1.8G rows and columns with average weight 24. $M'$ has 405M rows, with average weight 252.

When doing “replay”, with the classical strategy, we do row combinations directly on $M$, with initial average weight of 24.

With the double matrix strategy, we do row combinations on $P$, which is initially the identity, with average weight 1.
Possible Subtasks

- rewrite “replay” to perform the row combinations on $P$, initialized to the identity matrix, to get an idea of the final average weight of $P$.
- rewrite “merge” to work on both $M'$ (initialized to $M$) and $P$ (initialized to 1). We need to construct $M'$ to know which ideals we can eliminate.
sage: M

[[0 1 0 0 0 1 0 0]
 [1 0 1 1 0 1 1 1]
 [1 0 1 0 1 0 0 0]
 [1 0 0 1 0 1 0 1]
 [1 1 0 1 1 1 0 0]
 [0 0 0 1 0 1 0 1]
 [0 1 0 1 1 0 0 1]
 [1 0 1 0 1 1 1 1]]

If we want to cancel column 6 (starting from 0) we add row 1 to row 7.
If we want to cancel column 6 (starting from 0) we add row 1 to row 7:

```sage
P1 = matrix(GF(2), 8, 8, 1); P1[7,1] = 1
sage: P1, P1*M
```

Row 1 and column 6 are now inactive. Now to cancel column 7 we add row 5 to rows 3 and 6.
To cancel column 7 we add row 5 to rows 3 and 6:

```python
sage: P2=matrix(GF(2),8,8,1); P2[3,5]=P2[6,5]=1
sage: P2*P1, P2*P1*M

\[
\begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\
\end{bmatrix},
\begin{bmatrix}
0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\
1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 \\
1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \\
0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\
\end{bmatrix}
\]

Rows 1, 5 and columns 6, 7 are now inactive.
At each step we need the current matrix $M'$ to identify which ideals we can merge, and the current $P$ to compute the cost of each merge:

- scan columns of $M'$ to identify those $j$ of weight $k \leq K$;
- for each such column $j$ of weight $k$, compute the cost of the merge in $P$;
- perform the merges with smallest cost by updating both $P$ and $M'$. 