NO NEWS ON MATRIX MULTIPLICATION

Manuel Kauers · Institute for Algebra · JKU

$$\begin{pmatrix} a_{1,1} & a_{1,2} \\ a_{2,1} & a_{2,2} \end{pmatrix} \begin{pmatrix} b_{1,1} & b_{1,2} \\ b_{2,1} & b_{2,2} \end{pmatrix} = \begin{pmatrix} c_{1,1} & c_{1,2} \\ c_{2,1} & c_{2,2} \end{pmatrix}$$

$$c_{1,1} = a_{1,1} \cdot b_{1,1} + a_{1,2} \cdot b_{2,1}$$
$$c_{1,2} = a_{1,1} \cdot b_{1,2} + a_{1,2} \cdot b_{2,2}$$
$$c_{2,1} = a_{2,1} \cdot b_{1,1} + a_{2,2} \cdot b_{2,1}$$
$$c_{2,2} = a_{2,1} \cdot b_{1,2} + a_{2,2} \cdot b_{2,2}$$

$$\begin{pmatrix} a_{1,1} & a_{1,2} \\ a_{2,1} & a_{2,2} \end{pmatrix} \begin{pmatrix} b_{1,1} & b_{1,2} \\ b_{2,1} & b_{2,2} \end{pmatrix} = \begin{pmatrix} c_{1,1} & c_{1,2} \\ c_{2,1} & c_{2,2} \end{pmatrix}$$

$$\begin{aligned} c_{1,1} &= M_1 + M_4 - M_5 + M_7 \\ c_{1,2} &= M_3 + M_5 \\ c_{2,1} &= M_2 + M_4 \\ c_{2,2} &= M_1 - M_2 + M_3 + M_6 \end{aligned}$$

$$\begin{pmatrix} a_{1,1} & a_{1,2} \\ a_{2,1} & a_{2,2} \end{pmatrix} \begin{pmatrix} b_{1,1} & b_{1,2} \\ b_{2,1} & b_{2,2} \end{pmatrix} = \begin{pmatrix} c_{1,1} & c_{1,2} \\ c_{2,1} & c_{2,2} \end{pmatrix}$$

... where

$$M_1 = (a_{1,1} + a_{2,2}) \cdot (b_{1,1} + b_{2,2})$$
$$M_2 = (a_{2,1} + a_{2,2}) \cdot b_{1,1}$$
$$M_3 = a_{1,1} \cdot (b_{1,2} - b_{2,2})$$
$$M_4 = a_{2,2} \cdot (b_{2,1} - b_{1,1})$$
$$M_5 = (a_{1,1} + a_{1,2}) \cdot b_{2,2}$$
$$M_6 = (a_{2,1} - a_{1,1}) \cdot (b_{1,1} + b_{1,2})$$
$$M_7 = (a_{1,2} - a_{2,2}) \cdot (b_{2,1} + b_{2,2})$$

$$\begin{pmatrix} a_{1,1} & a_{1,2} \\ a_{2,1} & a_{2,2} \end{pmatrix} \begin{pmatrix} b_{1,1} & b_{1,2} \\ b_{2,1} & b_{2,2} \end{pmatrix} = \begin{pmatrix} c_{1,1} & c_{1,2} \\ c_{2,1} & c_{2,2} \end{pmatrix}$$

- This scheme needs 7 multiplications instead of 8.
- Recursive application allows to multiply $n \times n$ matrices with $O(n^{\log_2 7})$ operations in the ground ring.
- Let ω be the smallest number so that $n \times n$ matrices can be multiplied using $O(n^{\omega})$ operations in the ground domain.
- Then $2 \leq \omega < 3$. What is the exact value?

- Strassen 1969: $\omega \leq \log_2 7 \leq 2.807$

- Strassen 1969: $\omega \le \log_2 7 \le 2.807$
- Pan 1978: $\omega \le 2.796$
- Bini et al. 1979 : $\omega \le 2.7799$
- Schönhage 1981: $\omega \le 2.522$
- Romani 1982: $\omega \le 2.517$
- Coppersmith/Winograd 1981: $\omega \le 2.496$
- Strassen 1986: $\omega \le 2.479$
- Coppersmith/Winograd 1990: $\omega \le 2.376$

- Strassen 1969: $\omega \leq \log_2 7 \leq 2.807$
- Pan 1978: $\omega \leq 2.796$
- Bini et al. 1979 : $\omega \leq 2.7799$
- Schönhage 1981: $\omega \leq 2.522$
- Romani 1982: $\omega \leq 2.517$
- Coppersmith/Winograd 1981: $\omega \leq 2.496$
- Strassen 1986: $\omega \leq 2.479$
- Coppersmith/Winograd 1990: $\omega \leq 2.376$
- Stothers 2010: $\omega \leq 2.374$
- Williams 2011: $\omega \leq 2.3728642$
- Le Gall 2014 : $\omega \leq 2.3728639$

- Only Strassen's algorithm beats the classical algorithm for reasonable problem sizes.

- Only Strassen's algorithm beats the classical algorithm for reasonable problem sizes.
- Want: a matrix multiplication algorithm that beats Strassen's algorithm for matrices of moderate size.

- Only Strassen's algorithm beats the classical algorithm for reasonable problem sizes.
- Want: a matrix multiplication algorithm that beats Strassen's algorithm for matrices of moderate size.
- Idea: instead of dividing the matrices into 2×2-block matrices, divide them into 3×3-block matrices.

- Only Strassen's algorithm beats the classical algorithm for reasonable problem sizes.
- Want: a matrix multiplication algorithm that beats Strassen's algorithm for matrices of moderate size.
- Idea: instead of dividing the matrices into 2×2-block matrices, divide them into 3×3-block matrices.
- Question: What's the minimal number of multiplications needed to multiply two 3×3 matrices?

- Only Strassen's algorithm beats the classical algorithm for reasonable problem sizes.
- Want: a matrix multiplication algorithm that beats Strassen's algorithm for matrices of moderate size.
- Idea: instead of dividing the matrices into 2×2-block matrices, divide them into 3×3-block matrices.
- Question: What's the minimal number of multiplications needed to multiply two 3×3 matrices?
- Answer: Nobody knows.

Question: What's the minimal number of multiplications needed to multiply two 3×3 matrices?

- naive algorithm: 27
- padd with zeros, use Strassen twice, cleanup: 25
- best known upper bound: 23 (Laderman 1976)
- best known lower bound: 19 (Bläser 2003)
- maximal number of multiplications allowed if we want to beat Strassen: 21 (because $\log_3 21 < \log_2 7 < \log_3 22$).

Using SAT solvers, Courtois et al. discovered a new scheme with 23 multiplication in 2011, but none with $\leq$22. Can we do better?

Using SAT solvers, Courtois et al. discovered a new scheme with 23 multiplication in 2011, but none with ≤ 22. Can we do better?

Idea:

- Find a better encoding
- Exploit symmetries of the problem
- Use more powerful SAT solvers
- Let it run on bigger computers
- Wait for a solution with fewer multiplications

Using SAT solvers, Courtois et al. discovered a new scheme with 23 multiplication in 2011, but none with ≤ 22. Can we do better?

Idea:

- Find a better encoding
- Exploit symmetries of the problem
- Use more powerful SAT solvers $\longrightarrow$ Biere's Treengeling
- Let it run on bigger computers $\longrightarrow$ SG Altix UV 1000
- Wait for a solution with fewer multiplications

Using SAT solvers, Courtois et al. discovered a new scheme with 23 multiplication in 2011, but none with $\leq$22. Can we do better?

Idea:

- Find a better encoding
- Exploit symmetries of the problem
- Use more powerful SAT solvers $\rightarrow$ Biere's Treengeling
- Let it run on bigger computers $\rightarrow$ SG Altix UV 1000
- Wait for a solution with fewer multiplications

We are still waiting.

Using SAT solvers, Courtois et al. discovered a new scheme with 23 multiplication in 2011, but none with ≤ 22. Can we do better?

Idea:

- Find a better encoding
- Exploit symmetries of the problem
- Use more powerful SAT solvers $\longrightarrow$ Biere's Treengeling
- Let it run on bigger computers $\longrightarrow$ SG Altix UV 1000
- Wait for a solution with fewer multiplications

We are still waiting.

(Possible other approach: use QBF instead of SAT.)

How to encode the search for a matrix multiplication scheme as a SAT problem?

How to encode the search for a matrix multiplication scheme as a SAT problem?

Make an ansatz

$$
\begin{aligned}
M_1 &= (\alpha_{1,1}^{(1)} a_{1,1} + \alpha_{1,2}^{(1)} a_{1,2} + \cdots)(\beta_{1,1}^{(1)} b_{1,1} + \cdots) \\
M_2 &= (\alpha_{1,1}^{(2)} a_{1,1} + \alpha_{1,2}^{(2)} a_{1,2} + \cdots)(\beta_{1,1}^{(2)} b_{1,1} + \cdots) \\
&\vdots \\
c_{1,1} &= \gamma_{1,1}^{(1)} M_1 + \gamma_{1,1}^{(2)} M_2 + \cdots \\
&\vdots
\end{aligned}
$$

How to encode the search for a matrix multiplication scheme as a SAT problem?

Make an ansatz

$$
\begin{aligned}
M_1 &= (\alpha_{1,1}^{(1)} a_{1,1} + \alpha_{1,2}^{(1)} a_{1,2} + \cdots)(\beta_{1,1}^{(1)} b_{1,1} + \cdots) \\
M_2 &= (\alpha_{1,1}^{(2)} a_{1,1} + \alpha_{1,2}^{(2)} a_{1,2} + \cdots)(\beta_{1,1}^{(2)} b_{1,1} + \cdots) \\
&\vdots \\
c_{1,1} &= \gamma_{1,1}^{(1)} M_1 + \gamma_{1,1}^{(2)} M_2 + \cdots \\
&\vdots
\end{aligned}
$$

Set $c_{i,j} = \sum_k a_{i,k} b_{k,j}$ for all i, j and compare coefficients.

How to encode the search for a matrix multiplication scheme as a SAT problem?

This gives the Brent equations (e.g., for 3×3 with 21 multiplications)

$$\forall\ i,j,k,l,m,n \in \{1,2,3\} : \sum_{q=1}^{21} \alpha_{i,j}^{(q)} \beta_{k,l}^{(q)} \gamma_{m,n}^{(q)} = \delta_{j,k}\delta_{i,m}\delta_{l,n}$$

How to encode the search for a matrix multiplication scheme as a SAT problem?

This gives the Brent equations (e.g., for 3×3 with 21 multiplications)

$$\forall\ i,j,k,l,m,n \in \{1,2,3\} : \sum_{q=1}^{21} \alpha_{i,j}^{(q)} \beta_{k,l}^{(q)} \gamma_{m,n}^{(q)} = \delta_{j,k}\delta_{i,m}\delta_{l,n}$$

- $3^6 = 729$ cubic equations
- $21 \cdot 9 \cdot 3 = 567$ variables

How to encode the search for a matrix multiplication scheme as a SAT problem?

This gives the Brent equations (e.g., for 3×3 with 21 multiplications)

$$\forall\ i,j,k,l,m,n \in \{1,2,3\} : \sum_{q=1}^{21} \alpha_{i,j}^{(q)} \beta_{k,l}^{(q)} \gamma_{m,n}^{(q)} = \delta_{j,k}\delta_{i,m}\delta_{l,n}$$

- $3^6 = 729$ cubic equations
- $21 \cdot 9 \cdot 3 = 567$ variables

Laderman claims that he solved this system by hand,
but he doesn't say exactly how.

How to encode the search for a matrix multiplication scheme as a SAT problem?

This gives the Brent equations (e.g., for 3×3 with 21 multiplications)

$$\forall\ i,j,k,l,m,n \in \{1,2,3\} : \sum_{q=1}^{21} \alpha_{i,j}^{(q)} \beta_{k,l}^{(q)} \gamma_{m,n}^{(q)} = \delta_{j,k}\delta_{i,m}\delta_{l,n}$$

Solve this system in $\mathbb{Z}_2$.

How to encode the search for a matrix multiplication scheme as a SAT problem?

This gives the Brent equations (e.g., for 3×3 with 21 multiplications)

$$\forall\ i,j,k,l,m,n \in \{1,2,3\} : \sum_{q=1}^{21} \alpha_{i,j}^{(q)} \beta_{k,l}^{(q)} \gamma_{m,n}^{(q)} = \delta_{j,k}\delta_{i,m}\delta_{l,n}$$

Solve this system in $\mathbb{Z}_2$.

Reading $\alpha_{i,j}^{(q)}$, $\beta_{k,l}^{(q)}$, $\gamma_{m,n}^{(q)}$ as boolean variables and $+$ as XOR, the problem becomes a SAT problem.

Problem: SAT solvers don't like XOR. They want CNF as input.

Problem: SAT solvers don't like XOR. They want CNF as input.

$$a + b = 1 \iff (\bar{a} \lor \bar{b}) \land (a \lor b)$$

Problem: SAT solvers don't like XOR. They want CNF as input.

$$a + b = 1 \iff (\bar{a} \vee \bar{b}) \wedge (a \vee b)$$

$$a + b + c = 1 \iff (\bar{a} \vee \bar{b} \vee c) \wedge (\bar{a} \vee \bar{c} \vee b) \wedge (\bar{b} \vee \bar{c} \vee a) \wedge (a \vee b \vee c)$$

Problem: SAT solvers don't like XOR. They want CNF as input.

$$
\begin{aligned}
a + b = 1 &\iff (\bar{a} \lor \bar{b}) \land (a \lor b) \\
a + b + c = 1 &\iff (\bar{a} \lor \bar{b} \lor c) \land (\bar{a} \lor \bar{c} \lor b) \\
&\quad \land (\bar{b} \lor \bar{c} \lor a) \land (a \lor b \lor c) \\
a + b + c + d = 1 &\iff (\bar{a} \lor \bar{b} \lor \bar{c} \lor \bar{d}) \land (\bar{a} \lor \bar{b} \lor c \lor d) \\
&\quad \land (\bar{a} \lor \bar{c} \lor b \lor d) \land (\bar{a} \lor \bar{d} \lor b \lor c) \\
&\quad \land (\bar{b} \lor \bar{c} \lor a \lor d) \land (\bar{b} \lor \bar{d} \lor a \lor c) \\
&\quad \land (\bar{c} \lor \bar{d} \lor a \lor b) \land (a \lor b \lor c \lor d).
\end{aligned}
$$

Problem: SAT solvers don't like XOR. They want CNF as input.

$$
\begin{aligned}
a + b = 1 &\iff (\bar{a} \vee \bar{b}) \wedge (a \vee b) \\
a + b + c = 1 &\iff (\bar{a} \vee \bar{b} \vee c) \wedge (\bar{a} \vee \bar{c} \vee b) \\
&\qquad \wedge (\bar{b} \vee \bar{c} \vee a) \wedge (a \vee b \vee c) \\
a + b + c + d = 1 &\iff (\bar{a} \vee \bar{b} \vee \bar{c} \vee \bar{d}) \wedge (\bar{a} \vee \bar{b} \vee c \vee d) \\
&\qquad \wedge (\bar{a} \vee \bar{c} \vee b \vee d) \wedge (\bar{a} \vee \bar{d} \vee b \vee c) \\
&\qquad \wedge (\bar{b} \vee \bar{c} \vee a \vee d) \wedge (\bar{b} \vee \bar{d} \vee a \vee c) \\
&\qquad \wedge (\bar{c} \vee \bar{d} \vee a \vee b) \wedge (a \vee b \vee c \vee d).
\end{aligned}
$$

Expanding 21-term sum into CNF like this gives a million clauses.

SAT people avoid this explosion by assigning new variables ("Tseitin variables") to subexpressions before converting to CNF:

SAT people avoid this explosion by assigning new variables ("Tseitin variables") to subexpressions before converting to CNF:

$$a + b + c + d + e + f + g + h + i = 0$$

SAT people avoid this explosion by assigning new variables ("Tseitin variables") to subexpressions before converting to CNF:

$$a + b + c + d + e + f + g + h + i = 0$$

$$\downarrow$$

$$a + b + c = T_1$$

$$d + e + f = T_2$$

$$g + h + i = T_3$$

$$T_1 + T_2 + T_3 = 0$$

SAT people avoid this explosion by assigning new variables ("Tseitin variables") to subexpressions before converting to CNF:

$$a + b + c + d + e + f + g + h + i = 0$$

$$\downarrow$$

$$a + b + c = T_1 \quad \to \text{CNF}$$

$$d + e + f = T_2 \quad \to \text{CNF}$$

$$g + h + i = T_3 \quad \to \text{CNF}$$

$$T_1 + T_2 + T_3 = 0 \quad \to \text{CNF}$$

SAT people avoid this explosion by assigning new variables ("Tseitin variables") to subexpressions before converting to CNF:

$$a + b + c + d + e + f + g + h + i = 0$$

$$\downarrow$$

$$a + b + c = T_1 \quad \to \text{CNF}$$

$$d + e + f = T_2 \quad \to \text{CNF}$$

$$g + h + i = T_3 \quad \to \text{CNF}$$

$$T_1 + T_2 + T_3 = 0 \quad \to \text{CNF}$$

This decreases the number (and length) of clauses at the cost of increasing the number of variables.

Breaking all the sums of length 21 into chunks of size 3 terms each, we end up with a CNF with 21861 variables and 116748 clauses.

Breaking all the sums of length 21 into chunks of size 3 terms each, we end up with a CNF with 21861 variables and 116748 clauses.

Next, we enrich this formula with some additional information that the solver might find helpful.

Breaking all the sums of length 21 into chunks of size 3 terms each, we end up with a CNF with 21861 variables and 116748 clauses.

Next, we enrich this formula with some additional information that the solver might find helpful. For example:

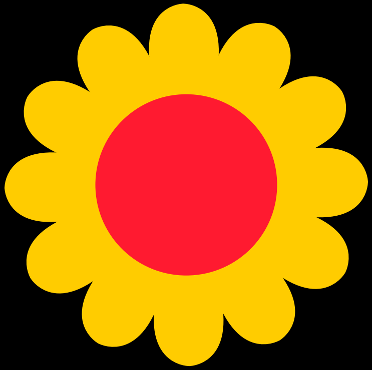
- For each i, j, at least one of $\alpha_{i,j}^{(1)}, \dots, \alpha_{i,j}^{(q)}$ must be nonzero. Likewise for β and γ.

Breaking all the sums of length 21 into chunks of size 3 terms each, we end up with a CNF with 21861 variables and 116748 clauses.

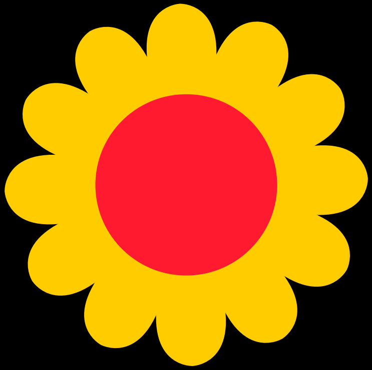
Next, we enrich this formula with some additional information that the solver might find helpful. For example:

- For each i, j, at least one of $\alpha_{i,j}^{(1)}, \ldots, \alpha_{i,j}^{(q)}$ must be nonzero. Likewise for β and γ.
- For each q, at least one of the $\alpha_{i,j}^{(q)}$ must be nonzero. Likewise for β and γ.

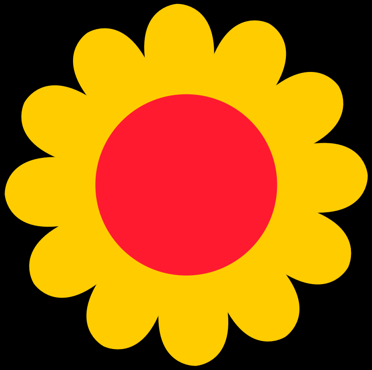
We should also exploit symmetries.



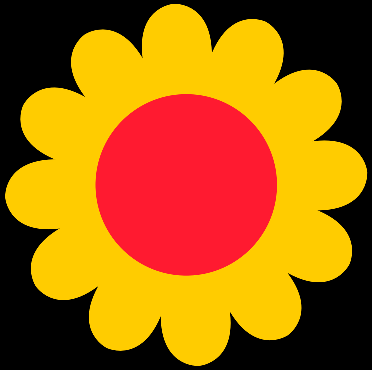
We should also exploit symmetries.



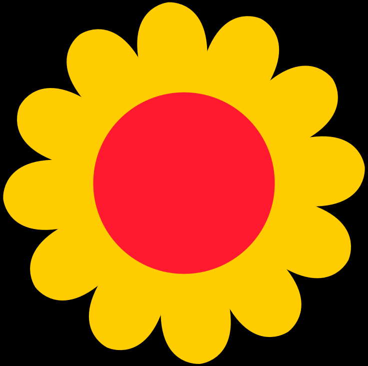
We should also exploit symmetries.



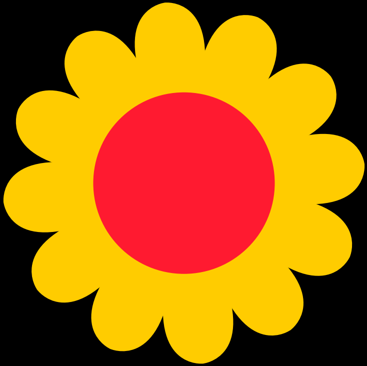
We should also exploit symmetries.



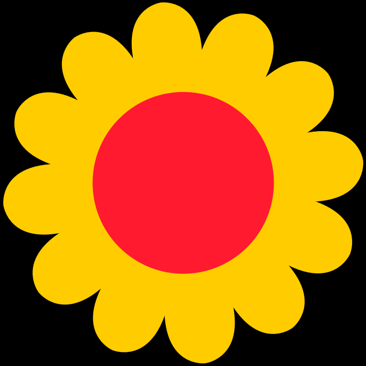
We should also exploit symmetries.



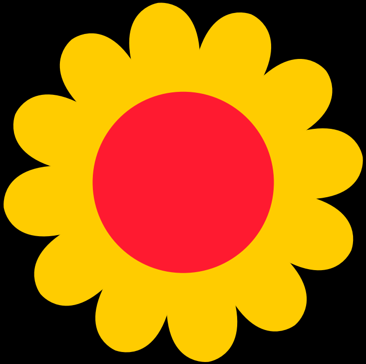
We should also exploit symmetries.



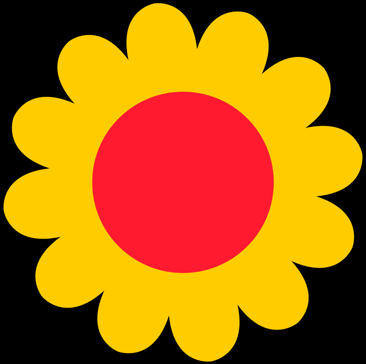
We should also exploit symmetries.



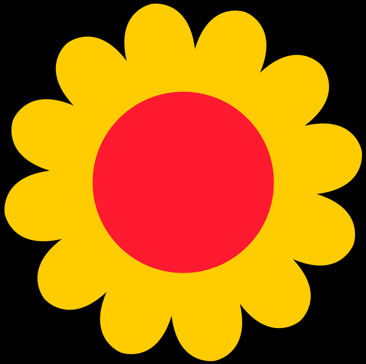
We should also exploit symmetries.



We should also exploit symmetries.

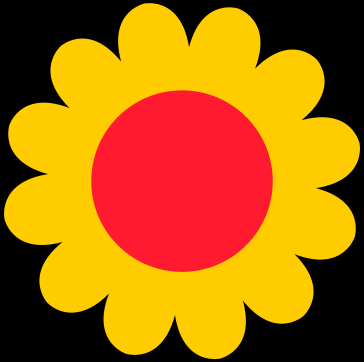


We should also exploit symmetries.

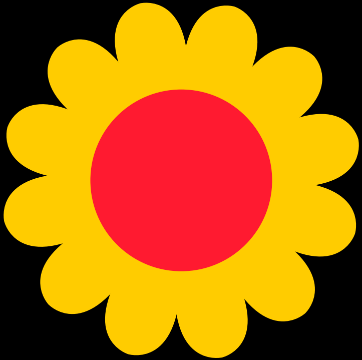


We should also exploit symmetries.

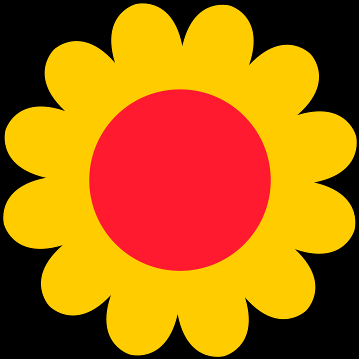
We should also exploit symmetries.



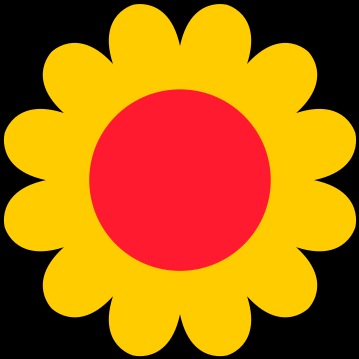
We should also exploit symmetries.



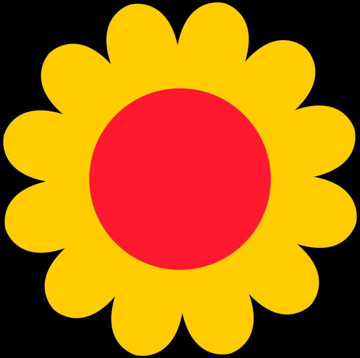
We should also exploit symmetries.



We should also exploit symmetries.

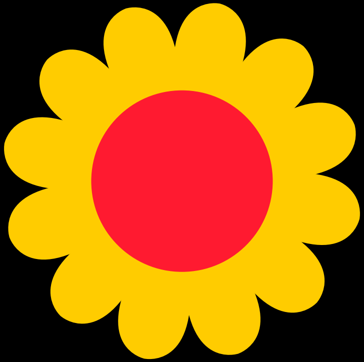


We should also exploit symmetries.



We should also exploit symmetries.

We should also exploit symmetries.



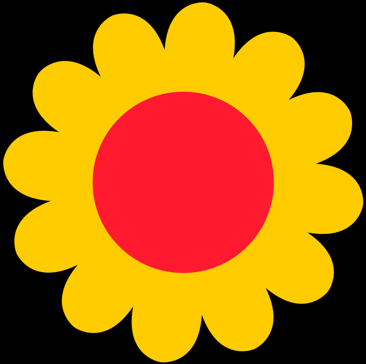
We should also exploit symmetries.

We should also exploit symmetries.

We should also exploit symmetries.

We should also exploit symmetries.

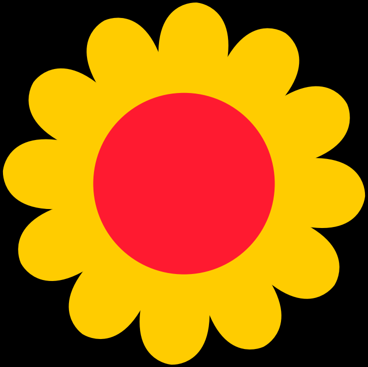
We should also exploit symmetries.



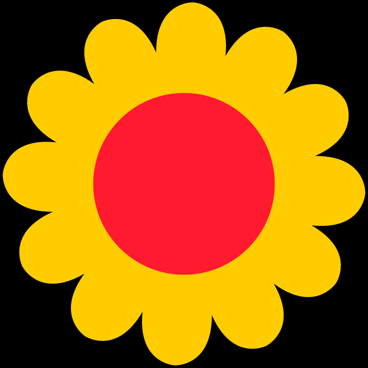
We should also exploit symmetries.

We should also exploit symmetries.

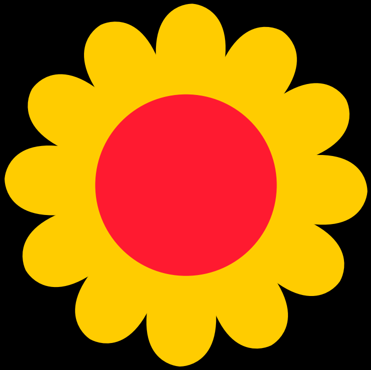
We should also exploit symmetries.



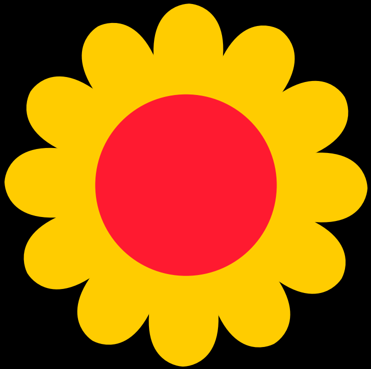
We should also exploit symmetries.



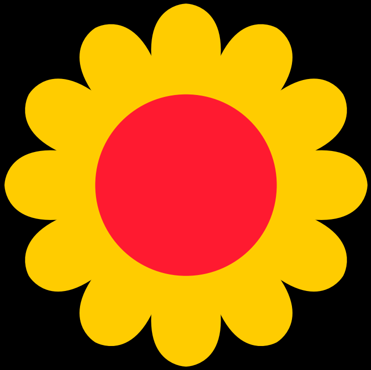
We should also exploit symmetries.



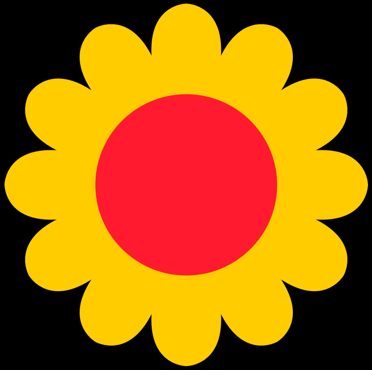
We should also exploit symmetries.



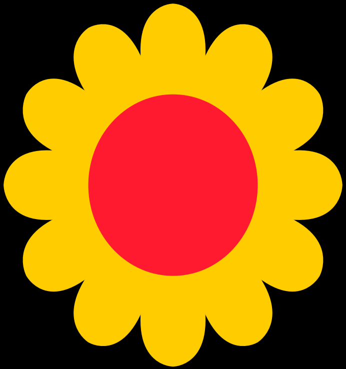
We should also exploit symmetries.



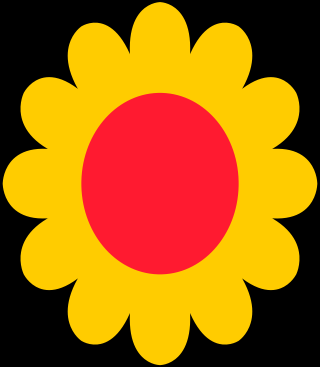
We should also exploit symmetries.



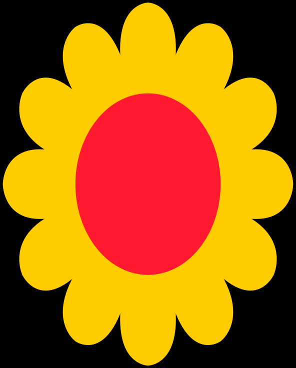
We should also exploit symmetries.



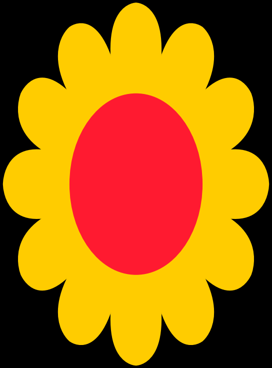
We should also exploit symmetries.



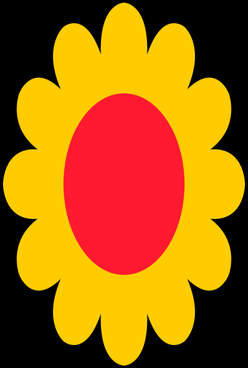
We should also exploit symmetries.



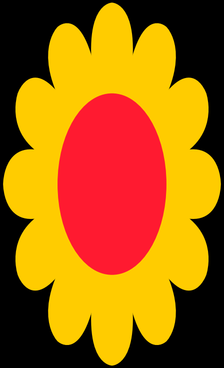
We should also exploit symmetries.



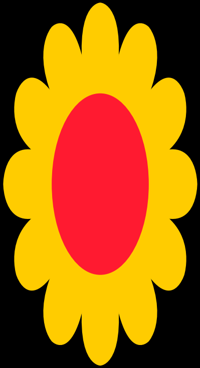
We should also exploit symmetries.



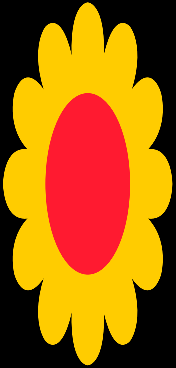
We should also exploit symmetries.



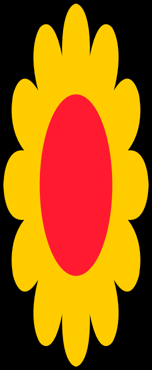
We should also exploit symmetries.



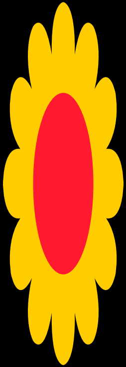
We should also exploit symmetries.



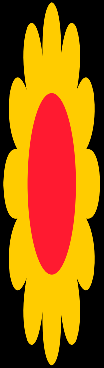
We should also exploit symmetries.



We should also exploit symmetries.



We should also exploit symmetries.



We should also exploit symmetries.

We should also exploit symmetries.

We should also exploit symmetries.

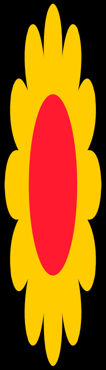
We should also exploit symmetries.

We should also exploit symmetries.

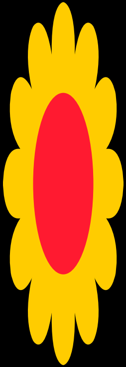
We should also exploit symmetries.

We should also exploit symmetries.

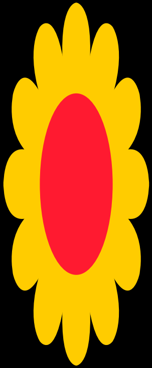
We should also exploit symmetries.



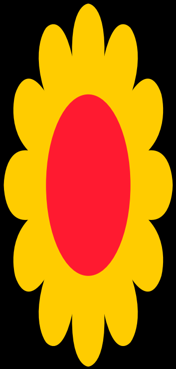
We should also exploit symmetries.



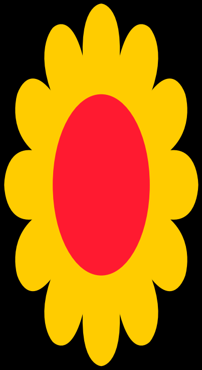
We should also exploit symmetries.



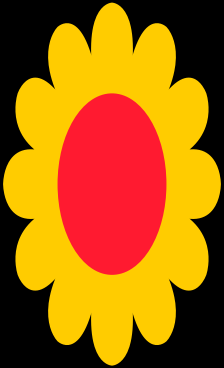
We should also exploit symmetries.



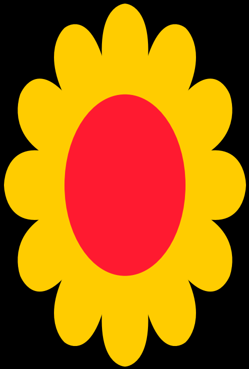
We should also exploit symmetries.



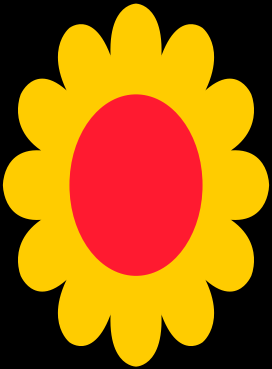
We should also exploit symmetries.



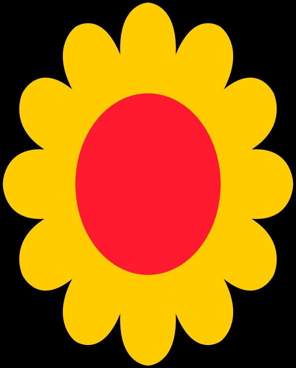
We should also exploit symmetries.



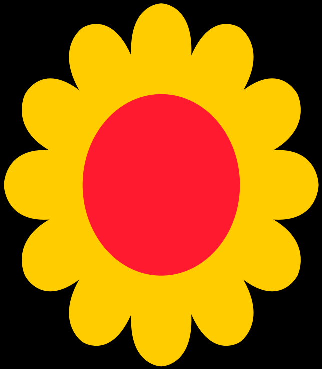
We should also exploit symmetries.



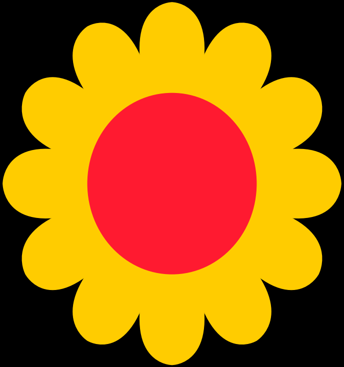
We should also exploit symmetries.



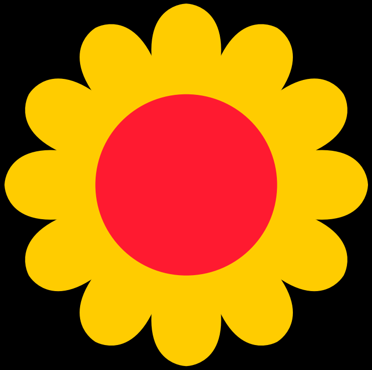
We should also exploit symmetries.



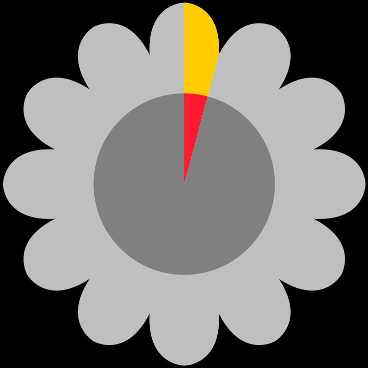
We should also exploit symmetries.



We should also exploit symmetries.

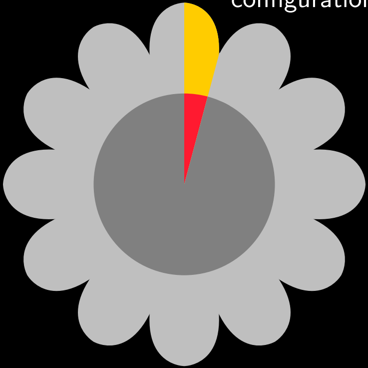


We should also exploit symmetries.



We should also exploit symmetries.

It suffices to inspect one configuration per orbit.



Matrix multiplication $AB = C$ enjoys several symmetries:

Matrix multiplication $AB = C$ enjoys several symmetries:

- $AUU^{-1}B = C$ for every invertible U

Matrix multiplication $AB = C$ enjoys several symmetries:

- $AUU^{-1}B = C$ for every invertible U
- $VAB = VC$ for every invertible V

Matrix multiplication $AB = C$ enjoys several symmetries:

- $AUU^{-1}B = C$ for every invertible U
- $VAB = VC$ for every invertible V
- $ABW = CW$ for every invertible W

Matrix multiplication $AB = C$ enjoys several symmetries:

- $AUU^{-1}B = C$ for every invertible U
- $VAB = VC$ for every invertible V
- $ABW = CW$ for every invertible W
- $B^\top A^\top = C^\top$

Matrix multiplication $AB = C$ enjoys several symmetries:

- $AUU^{-1}B = C$ for every invertible U
- $VAB = VC$ for every invertible V
- $ABW = CW$ for every invertible W
- $B^\top A^\top = C^\top$
- and one more that is a little more subtle

Matrix multiplication $AB = C$ enjoys several symmetries:

- $AUU^{-1}B = C$ for every invertible U
- $VAB = VC$ for every invertible V
- $ABW = CW$ for every invertible W
- $B^\top A^\top = C^\top$
- and one more that is a little more subtle

Altogether, the symmetry group is $S_3 \times \mathrm{GL}(n)^3$.

Matrix multiplication $AB = C$ enjoys several symmetries:

- $AUU^{-1}B = C$ for every invertible U
- $VAB = VC$ for every invertible V
- $ABW = CW$ for every invertible W
- $B^\top A^\top = C^\top$
- and one more that is a little more subtle

Altogether, the symmetry group is $S_3 \times \mathrm{GL}(n)^3$.

We have worked out a small set of clauses that has exactly one solution in each orbit under this group action.

Simpler example: For the group action

$$\mathrm{GL}(3)^2 \times \mathbb{Z}_2^{3\times 3} \to \mathbb{Z}_2^{3\times 3}, \qquad (U, V) \cdot A := U\,A\,V^{-1}$$

we have four orbits.

Simpler example: For the group action

$$\mathrm{GL}(3)^2 \times \mathbb{Z}_2^{3\times 3} \to \mathbb{Z}_2^{3\times 3}, \qquad (U, V) \cdot A := U\,A\,V^{-1}$$

we have four orbits. A choice of orbit representatives is

$$\begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \quad \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \quad \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \quad \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

Simpler example: For the group action

$$\mathrm{GL}(3)^2 \times \mathbb{Z}_2^{3\times 3} \to \mathbb{Z}_2^{3\times 3}, \qquad (U, V) \cdot A := U\,A\,V^{-1}$$

we have four orbits. A choice of orbit representatives is

$$\begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \quad \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \quad \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \quad \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

not needed

Simpler example: For the group action

$$\mathrm{GL}(3)^2 \times \mathbb{Z}_2^{3\times 3} \to \mathbb{Z}_2^{3\times 3}, \qquad (U, V) \cdot A := U\,A\,V^{-1}$$

we have four orbits. A choice of orbit representatives is

$$\begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \quad \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \quad \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \quad \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

not needed	Stabilizer $H_1 \leq \mathrm{GL}(3)^2$ $\lvert H_1\rvert = 576$	Stabilizer $H_2 \leq \mathrm{GL}(3)^2$ $\lvert H_2\rvert = 96$	Stabilizer $H_3 \leq \mathrm{GL}(3)^2$ $\lvert H_3\rvert = 168$

Next, for each $i = 1, 2, 3$ consider the group action

$$\big(H_i \times \mathrm{GL}(3)\big) \times \mathbb{Z}_2^{3\times 3} \to \mathbb{Z}_2^{3\times 3}, \quad (U, V, W) \times B := V\,B\,W^{-1}.$$

Next, for each $i = 1, 2, 3$ consider the group action

$$\big(H_i \times \mathrm{GL}(3)\big) \times \mathbb{Z}_2^{3\times 3} \to \mathbb{Z}_2^{3\times 3}, \quad (U, V, W) \times B := V\,B\,W^{-1}.$$

For instance for $i = 2$ we get 6 orbits.

Next, for each $i = 1, 2, 3$ consider the group action

$$\big(H_i \times \mathrm{GL}(3)\big) \times \mathbb{Z}_2^{3\times 3} \to \mathbb{Z}_2^{3\times 3}, \quad (U, V, W) \times B := V\,B\,W^{-1}.$$

For instance for $i = 2$ we get 6 orbits. A list of representatives is

$$\begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \quad \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \quad \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix},$$
$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \quad \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}, \quad \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

Next, for each $i = 1, 2, 3$ consider the group action

$$\big(H_i \times \mathrm{GL}(3)\big) \times \mathbb{Z}_2^{3\times 3} \to \mathbb{Z}_2^{3\times 3}, \quad (U, V, W) \times B := V\, B\, W^{-1}.$$

For instance for $i = 2$ we get 6 orbits. A list of representatives is

$$\begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \quad \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \quad \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix},$$
$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \quad \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}, \quad \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

Similarly, we have 6 orbits for $i = 1$ and 4 orbits for $i = 3$.

Altogether, we have found $5+5+3$ nontrivial orbits for the combined group action

$$\begin{aligned}&\mathrm{GL}(3)^3 \times (\mathbb{Z}_2^{3\times 3})^2 \to (\mathbb{Z}_2^{3\times 3})^2,\\ &(U,V,W)\times(A,B) := (U\,A\,V^{-1},\ V\,B\,W^{-1}).\end{aligned}$$

Altogether, we have found $5+5+3$ nontrivial orbits for the combined group action

$$\mathrm{GL}(3)^3 \times (\mathbb{Z}_2^{3\times 3})^2 \to (\mathbb{Z}_2^{3\times 3})^2,$$
$$(U, V, W) \times (A, B) := (U\,A\,V^{-1},\ V\,B\,W^{-1}).$$

For each representative, determine the stabilizer $H \leq \mathrm{GL}(3)^3$ and find the orbits of the action of H on $C \in \mathbb{Z}_2^{3\times 3}$.

Altogether, we have found $5+5+3$ nontrivial orbits for the combined group action

$$\mathrm{GL}(3)^3 \times (\mathbb{Z}_2^{3\times 3})^2 \to (\mathbb{Z}_2^{3\times 3})^2,$$
$$(U,V,W) \times (A,B) := (U\,A\,V^{-1},\ V\,B\,W^{-1}).$$

For each representative, determine the stabilizer $H \leq \mathrm{GL}(3)^3$ and find the orbits of the action of H on $C \in \mathbb{Z}_2^{3\times 3}$.

Merge orbits that are equivalent under the action of S_3.

Altogether, we have found $5 + 5 + 3$ nontrivial orbits for the combined group action

$$\mathrm{GL}(3)^3 \times (\mathbb{Z}_2^{3\times 3})^2 \to (\mathbb{Z}_2^{3\times 3})^2,$$
$$(U, V, W) \times (A, B) := (U\,A\,V^{-1},\ V\,B\,W^{-1}).$$

For each representative, determine the stabilizer $H \leq \mathrm{GL}(3)^3$ and find the orbits of the action of H on $C \in \mathbb{Z}_2^{3\times 3}$.

Merge orbits that are equivalent under the action of S_3.

Finally, instead of $(2^9 - 1)^3 = 133432830$ matrix triples, we have to consider only 94 orbit representatives.

Actually there are some much more obvious and much more effective symmetries:

$$\forall\ i, j, k, l, m, n \in \{1, 2, 3\} : \sum_{q=1}^{21} \alpha_{i,j}^{(q)} \beta_{k,l}^{(q)} \gamma_{m,n}^{(q)} = \delta_{j,k} \delta_{i,m} \delta_{l,n}$$

Actually there are some much more obvious and much more effective symmetries:

$$\forall\, i, j, k, l, m, n \in \{1, 2, 3\} : \sum_{q=1}^{21} \alpha_{i,j}^{(q)} \beta_{k,l}^{(q)} \gamma_{m,n}^{(q)} = \delta_{j,k} \delta_{i,m} \delta_{l,n}$$

We can change the order of summation in 21! possible ways.

Actually there are some much more obvious and much more effective symmetries:

$$\forall\ i, j, k, l, m, n \in \{1, 2, 3\} : \sum_{q=1}^{21} \alpha_{i,j}^{(q)} \beta_{k,l}^{(q)} \gamma_{m,n}^{(q)} = \delta_{j,k} \delta_{i,m} \delta_{l,n}$$

We can change the order of summation in 21! possible ways.

We impose a lexicographic order on the summands in order to break this symmetry.

At the end of the day, we have an enriched CNF with 25078 variables and 130709 clauses.

At the end of the day, we have an enriched CNF with 25078 variables and 130709 clauses.

Unfortunately, we haven't been able to solve it within some 10 years of CPU time.

At the end of the day, we have an enriched CNF with 25078 variables and 130709 clauses.

Unfortunately, we haven't been able to solve it within some 10 years of CPU time.

In contrast, finding Strassen's original algorithm for 2×2 matrices is a matter of a few seconds only.

At the end of the day, we have an enriched CNF with 25078 variables and 130709 clauses.

Unfortunately, we haven't been able to solve it within some 10 years of CPU time.

In contrast, finding Strassen's original algorithm for 2×2 matrices is a matter of a few seconds only.

The optimal algorithms for the rectangular cases $(2 \times 2)(2 \times 3)$ and $(2 \times 3)(3 \times 3)$ are found within a few minutes.

At the end of the day, we have an enriched CNF with 25078 variables and 130709 clauses.

Unfortunately, we haven't been able to solve it within some 10 years of CPU time.

In contrast, finding Strassen's original algorithm for 2×2 matrices is a matter of a few seconds only.

The optimal algorithms for the rectangular cases $(2 \times 2)(2 \times 3)$ and $(2 \times 3)(3 \times 3)$ are found within a few minutes.

We will keep trying to find some improvement for 3×3. For the time being, we have no news on matrix multiplication.