Your name is Hsin-Po Wang; propose a cool tech talk title

Here is a possible topic for a cool, technical talk.

Title: Fast Matrix Multiplication That Does Not Slow Down

Speaker: Hsin-Po Wang

Date: (insert talk date)

Why do you pick matrix multiplication? Use itemize

- Matrix multiplication is widely used in science and engineering.
- Special cases of matrix multiplication include rotation and markov chain.
- The converse of matrix multiplication is called matrix decomposition, also widely used as it is a central part of solving linear equations.

What is fast about matrix multi? Use itemize

- Fast matrix multiplication is faster than computing the product using the schoolbook formula.
- The complexity of fast matrix multiplication is $n^{2.4}$.
- It is used when the matrices are large.

Give me an example of fast matrix multi

An example of fast matrix multi is the Strassen algorithm. The Strassen algorithm is known for multiplying 2 2 by 2 matrices using only 7 multiplications.

Give me an example of fast matrix multi

An example of fast matrix multi is the Strassen algorithm. The Strassen algorithm is known for multiplying 2 2 by 2 matrices using only 7 multiplications.

The formula for Strassen algorithm is

$$
\begin{aligned}
S^{(1)} &:= (A_{11} + A_{22})\bigstar(B_{11} + B_{22}),\\
S^{(2)} &:= (A_{21} + A_{22})\bigstar B_{11},\\
S^{(3)} &:= A_{11}\bigstar(B_{12} - B_{22}),\\
S^{(4)} &:= A_{22}\bigstar(-B_{11} + B_{21}),\\
S^{(5)} &:= (A_{11} + A_{12})\bigstar B_{22},\\
S^{(6)} &:= (-A_{11} + A_{21})\bigstar(B_{11} + B_{12}),\\
S^{(7)} &:= (A_{12} - A_{22})\bigstar(B_{21} + B_{22}).
\end{aligned}
$$

$$
\begin{aligned}
S^{(1)} &:= (A_{11} + A_{22}) \bigstar (B_{11} + B_{22}),\\
S^{(2)} &:= (A_{21} + A_{22}) \bigstar B_{11},\\
S^{(3)} &:= A_{11} \bigstar (B_{12} - B_{22}),\\
S^{(4)} &:= A_{22} \bigstar (-B_{11} + B_{21}),\\
S^{(5)} &:= (A_{11} + A_{12}) \bigstar B_{22},\\
S^{(6)} &:= (-A_{11} + A_{21}) \bigstar (B_{11} + B_{12}),\\
S^{(7)} &:= (A_{12} - A_{22}) \bigstar (B_{21} + B_{22}).
\end{aligned}
$$

$$
\begin{aligned}
C_{11} &= S^{(1)} + S^{(4)} - S^{(5)} + S^{(7)},\\
C_{12} &= S^{(3)} + S^{(5)},\\
C_{21} &= S^{(2)} + S^{(4)},\\
C_{22} &= S^{(1)} - S^{(2)} + S^{(3)} + S^{(6)},
\end{aligned}
$$

Can I use Strassen for large matrix?

Yes, you can use Strassen algorithm for large matrix.

A large matrix can be divided into 2 by 2 blocks. Applying Strassen algorithm to those blocks will give you 7 “block multiplications”. For each block multiplication, you can apply Strassen algorithm again to generate more blocks.

Can I use Strassen for large matrix?

Yes, you can use Strassen algorithm for large matrix.

A large matrix can be divided into 2 by 2 blocks. Applying Strassen algorithm to those blocks will give you 7 “block multiplications”. For each block multiplication, you can apply Strassen algorithm again to generate more blocks.

The complexity of Strassen algorithm is $n^{\log_2 7}$ according to the master theorem with fan-out factor 7 and size reduction factor 2.

Is Strassen used in real systems?

Yes, a variant of Strassen algorithm is implemented in BLAS and shown to outperform the standard method at matrix size 1000 by 1000.

Then why do you say Strassen slow?

Strassen algorithm is a fast matrix multiplication algorithm that is claimed to be faster in certain parameter range.

However, the machines that carries the computation might slow down. Since large matrices are divided into blocks and distributed over a network of computers, any computer failure will slow down the whole distributed matrix multiplication.

Introduce distributed matrix multi. Use itemize

- Distributed matrix multiplication is a parallel algorithm to compute the product of matrices using a network of computers.
- Distributed matrix multiplication can be significantly faster as it takes advantage on parallelism.
- If the matrices is too large for a single computer, distributed matrix multiplication might be necessary.

How to avoid slow down

Here is a list of previous works that avoids straggling.

Lee et al. '18 suggests using error correcting codes to protect distributed matrix multiplication.

Suppose A and B is k by k matrices. Suppose G is an $[n, k]$-MDS code represented by an n by k matrix. Lee et al. suggests computing $(GA)\bigstar B$ because the rows are linearly dependent.

How to avoid slow down

Here is a list of previous works that avoids straggling.

Lee et al. '18 suggests using error correcting codes to protect distributed matrix multiplication.

Suppose A and B is k by k matrices. Suppose G is an $[n, k]$-MDS code represented by an n by k matrix. Lee et al. suggests computing $(GA)\bigstar B$ because the rows are linearly dependent.

Bartan and Pilanci '19 suggests using polar code as G because polar code is a capacity-achieving low-complexity code.

How to avoid slow down

Here is a list of previous works that avoids straggling.

Lee et al. '18 suggests using error correcting codes to protect distributed matrix multiplication.

Suppose A and B is k by k matrices. Suppose G is an $[n, k]$-MDS code represented by an n by k matrix. Lee et al. suggests computing $(GA)\bigstar B$ because the rows are linearly dependent.

Bartan and Pilanci '19 suggests using polar code as G because polar code is a capacity-achieving low-complexity code.

Mallick et al. '20 suggests using LT code as G because LT code is a rate-less code.

Lee, Suh, and Ramchandran '17 suggests computing $(GA)\bigstar(BG^\top)$, where G is an MDS matrix.

Lee, Suh, and Ramchandran '17 suggests computing $(GA)\bigstar(BG^\top)$, where G is an MDS matrix.

Baharav et al. '18 suggests using a tensor power of $\left[\begin{smallmatrix}1&1&0\\1&0&1\end{smallmatrix}\right]$ as G.

Lee, Suh, and Ramchandran '17 suggests computing $(GA)\bigstar(BG^\top)$, where G is an MDS matrix.

Baharav et al. '18 suggests using a tensor power of $\left[\begin{smallmatrix}1&1&0\\1&0&1\end{smallmatrix}\right]$ as G.

Dutta et al. '20 suggests that the ith computer computes the tensor product $(Ac_i)\otimes(r_iB)$, where r_i are row vectors, c_i are column vectors, and any $2k-1$ c_ir_i span the space of k by k Toeplitz matrices.

Lee, Suh, and Ramchandran '17 suggests computing $(GA)\bigstar(BG^\top)$, where G is an MDS matrix.

Baharav et al. '18 suggests using a tensor power of $\left[\begin{smallmatrix}1&1&0\\1&0&1\end{smallmatrix}\right]$ as G.

Dutta et al. '20 suggests that the ith computer computes the tensor product $(Ac_i)\otimes(r_iB)$, where r_i are row vectors, c_i are column vectors, and any $2k-1$ c_ir_i span the space of k by k Toeplitz matrices.

Yu, Maddah-Ali, and Avestimehr '20 suggests that the ith computer computes the inner product $(r_iA)\cdot(Bc_i)$, where r_i are row vectors, c_i are column vectors, and any k^2 c_ir_i span the space of k by k matrices.

Why Yu's method work?

Suppose r_i are row vectors, c_i are column vectors, and any k^2 $c_i r_i$ span the space of k by k matrices. Yu, Maddah-Ali, and Avestimehr '20 suggests that the ith computer computes the inner product $(r_i A) \cdot (B c_i)$.

Yu, Maddah-Ali, and Avestimehr's idea is that because $(r_i A) \cdot (B c_i)$ is a scalar, a scalar is equal to its trace. Therefore we can add Tr and use the cyclic property

$$(r_i A) \cdot (B c_i) = \mathrm{Tr}((r_i A) \cdot (B c_i)) = \mathrm{Tr}((c_i r_i) \bigstar (AB)).$$

The ith computer is computing the inner produce of AB and $c_i r_i$. It remains to ensure that $c_i r_i$ span the space of matrices.

What might be a way to make contribution to this subject?

A possible way to make contribution to this subject is to combine fast matrix multiplication with distributed matrix multiplication.

Fast matrix multiplication makes matrix multiplication fast; distributed matrix multiplication makes matrix multiplication distributed and resilient against failure.

Show me an example of combination

The formula for Strassen algorithm is

$$
\begin{aligned}
S^{(1)} &:= (A_{11} + A_{22})\bigstar(B_{11} + B_{22}), \\
S^{(2)} &:= (A_{21} + A_{22})\bigstar B_{11}, \\
S^{(3)} &:= A_{11}\bigstar(B_{12} - B_{22}), \\
S^{(4)} &:= A_{22}\bigstar(-B_{11} + B_{21}), \\
S^{(5)} &:= (A_{11} + A_{12})\bigstar B_{22}, \\
S^{(6)} &:= (-A_{11} + A_{21})\bigstar(B_{11} + B_{12}), \\
S^{(7)} &:= (A_{12} - A_{22})\bigstar(B_{21} + B_{22}).
\end{aligned}
$$

Strassen algorithm can be combined with

$$
\begin{aligned}
S^{(8)} &:= (A_{11} + 2A_{21})\bigstar(-B_{11} + B_{12}), \\
S^{(9)} &:= (A_{12} + 2A_{22})\bigstar(-B_{21} + B_{22}).
\end{aligned}
$$

They satisfy the following equations

$$\begin{aligned}
C_{11} &= S^{(1)} && && && +S^{(4)} && -S^{(5)} && && +S^{(7)}, \\
C_{12} &= && && S^{(3)} && && +S^{(5)} && && , \\
C_{21} &= && S^{(2)} && && +S^{(4)} && && && , \\
C_{22} &= S^{(1)} && -S^{(2)} && +S^{(3)} && && && +S^{(6)} && ,
\end{aligned}$$

and a parity-check equation

$$S^{(1)} - 4S^{(2)} + 3S^{(3)} - 3S^{(4)} + 2S^{(5)} + 2S^{(6)} - S^{(7)} - S^{(8)} - S^{(9)} = 0.$$

Prove the parity-check equation

The parity-check equation

$$S^{(1)} - 4S^{(2)} + 3S^{(3)} - 3S^{(4)} + 2S^{(5)} + 2S^{(6)} - S^{(7)} - S^{(8)} - S^{(9)} = 0$$

can be proved by utilizing the associativity of matrix multiplication as illustrated by the following equations.

$$\begin{aligned}
&S^{(8)} + S^{(9)} \\
&\quad = \left(\begin{bmatrix} 1 & 2 \end{bmatrix} A\right) \bigstar \left(B \begin{bmatrix} -1 \\ 1 \end{bmatrix}\right) \\
&\quad = \begin{bmatrix} 1 & 2 \end{bmatrix} C \begin{bmatrix} -1 \\ 1 \end{bmatrix} \\
&\quad = \begin{bmatrix} 1 & 2 \end{bmatrix} \begin{bmatrix} S^{(1)} + S^{(4)} - S^{(5)} + S^{(7)} & S^{(3)} + S^{(5)} \\ S^{(2)} + S^{(4)} & S^{(1)} - S^{(2)} + S^{(3)} + S^{(6)} \end{bmatrix} \begin{bmatrix} -1 \\ 1 \end{bmatrix}
\end{aligned}$$

I want to call it "something" code.
What is a good word for "something"?
What is used to be nine but become eight?

A category that used to contain 9 items but is then reduced to 8 items is planets. In 2008, Pluto is reclassified as dwarf planet, leaving 8 planets behind.

Therefore, you can call it Pluto code.

Can you add more parity checks to Pluto code?

Strassen algorithm can be combined with

$$
\begin{aligned}
S^{(8)} &:= (A_{11} + 2A_{21})\bigstar(-B_{11} + B_{12}), \\
S^{(9)} &:= (A_{12} + 2A_{22})\bigstar(-B_{21} + B_{22}), \\
S^{(10)} &:= (3A_{11} - A_{21})\bigstar(B_{11} + 2B_{12}), \\
S^{(11)} &:= (3A_{12} - A_{22})\bigstar(B_{21} + 2B_{22}).
\end{aligned}
$$

They satisfy parity-check equations

$$
\begin{aligned}
S^{(1)} - 4S^{(2)} + 3S^{(3)} - 3S^{(4)} + 2S^{(5)} + 2S^{(6)} - S^{(7)} - S^{(8)} - S^{(9)} &= 0, \\
S^{(1)} + S^{(2)} + 4S^{(3)} + 2S^{(4)} + 3S^{(5)} - 2S^{(6)} + 3S^{(7)} - S^{(10)} - S^{(11)} &= 0.
\end{aligned}
$$

One more check

Strassen algorithm can be combined with

$$
\begin{aligned}
S^{(8)} &:= (A_{11} + 2A_{21}) \bigstar (-B_{11} + B_{12}),\\
S^{(9)} &:= (A_{12} + 2A_{22}) \bigstar (-B_{21} + B_{22}),\\
S^{(10)} &:= (3A_{11} - A_{21}) \bigstar (B_{11} + 2B_{12}),\\
S^{(11)} &:= (3A_{12} - A_{22}) \bigstar (B_{21} + 2B_{22}),\\
S^{(12)} &:= (2A_{11} - 3A_{21}) \bigstar (2B_{11} + B_{12}),\\
S^{(13)} &:= (2A_{12} - 3A_{22}) \bigstar (2B_{21} + B_{22}).
\end{aligned}
$$

They satisfy parity-check equations

$$
\begin{aligned}
S^{(1)} - 4S^{(2)} + 3S^{(3)} - 3S^{(4)} + 2S^{(5)} + 2S^{(6)} - S^{(7)} - S^{(8)} - S^{(9)} &= 0,\\
S^{(1)} + S^{(2)} + 4S^{(3)} + 2S^{(4)} + 3S^{(5)} - 2S^{(6)} + 3S^{(7)} - S^{(10)} - S^{(11)} &= 0.\\
S^{(1)} - 3S^{(2)} - S^{(3)} - 2S^{(4)} - 2S^{(5)} - 3S^{(6)} + 4S^{(7)} - S^{(12)} - S^{(13)} &= 0.
\end{aligned}
$$

Turn the parity check in to matrix form

The parity-check equations corresponds to the parity-check matrix

$$\begin{bmatrix}
1 & -4 & 3 & -3 & 2 & 2 & -1 & -1 & -1 & & & & \\
1 & 1 & 4 & 2 & 3 & -2 & 3 & & & -1 & -1 & & \\
1 & -3 & -1 & -2 & -2 & -3 & 4 & & & & & -1 & -1
\end{bmatrix}$$

Mark the triples of columns that has rank 2

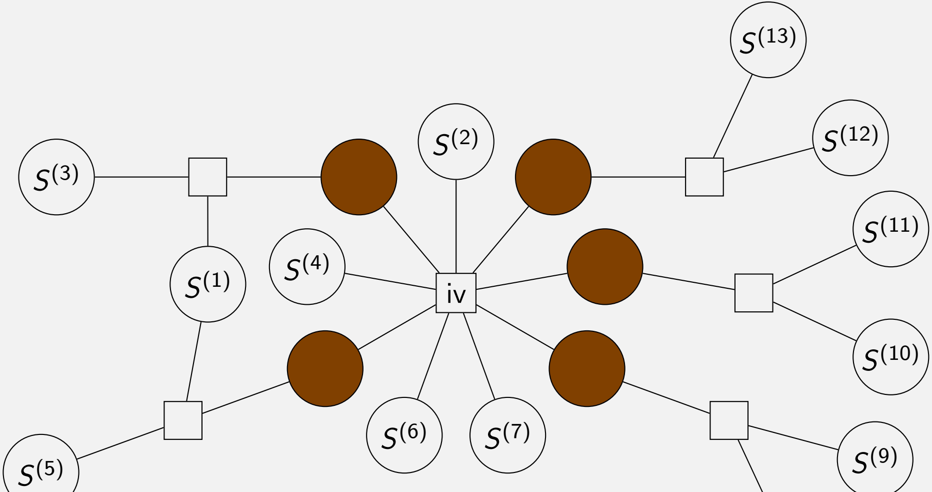
Done

$$
\begin{bmatrix}
1 & -4 & 3 & -3 & 2 & 2 & -1 & -1 & -1 & & & & \\
1 & 1 & 4 & 2 & 3 & -2 & 3 & & & -1 & -1 & & \\
1 & -3 & -1 & -2 & -2 & -3 & 4 & & & & & -1 & -1
\end{bmatrix}
$$

$$
\begin{array}{ccccccccccccc}
\cdot & \cdot & & \cdot & & & & & & & & & \\
\cdot & & \cdot & & \cdot & & & & & & & & \\
\cdot & & & & & \cdot & \cdot & & & & & &
\end{array}
$$

Turn the parity-check matrix into Tanner graph

The parity-check matrix can be visualized as this Tanner graph



Does Pluto applicable to large matrix?

Yes, since Pluto code is a parity-check code, and since parity-check code can be amplified using tensor product, Pluto code is applicable to large matrix.

Output a 7 by 7 array of dots

Here is a 7 by 7 array of dots:

```
. . . . . . .
. . . . . . .
. . . . . . .
. . . . . . .
. . . . . . .
. . . . . . .
. . . . . . .
```

Output a 9 by 9 array of x's
The upper left 7 by 7 dots are colored black
The upper right 7 by 2 dots are colored red
The bottom left 2 by 7 dots are colored blue
The bottom right 2 by 2 dots are colored cyan

Here is a 9 by 9 array of x's colored as described:

x x x x x x x x x
x x x x x x x x x
x x x x x x x x x
x x x x x x x x x
x x x x x x x x x
x x x x x x x x x
x x x x x x x x x
x x x x x x x x x
x x x x x x x x x

Since every axis is like a $[9, 8]$-MDS code,
every dimension will square the erasure probability, right?
What is the error exponent of tensored Pluto code?

Yes, since the weight enumerator of a $[9, 8]$-MDS code is divisible by x^2, asymptotically if x is small enough the block error probability can be estimated by x^2.

Since every axis is like a $[9, 8]$-MDS code,
every dimension will square the erasure probability, right?
What is the error exponent of tensored Pluto code?

Yes, since the weight enumerator of a $[9, 8]$-MDS code is divisible by x^2, asymptotically if x is small enough the block error probability can be estimated by x^2.

The error exponent is the number E such that the block error probability can be estimated by x^{n^E}, where n is the block length. Since every dimension squares the error probability and since there are $\log_2 n$ dimensions. The number of error probability getting squared is $\log_2 n$. Hence the block error probability can be estimated by $x^{2^{\log_2 n}} = x^n$. Equating $n^E = n$, we can infer that $E = 1$.

What should I say when I want to end my talk?

At the end of your presentation, you can say

- "Thank you for listening."
- "I hope this has been helpful/informative/interesting."
- "I appreciate your attention."
- "Any questions?"