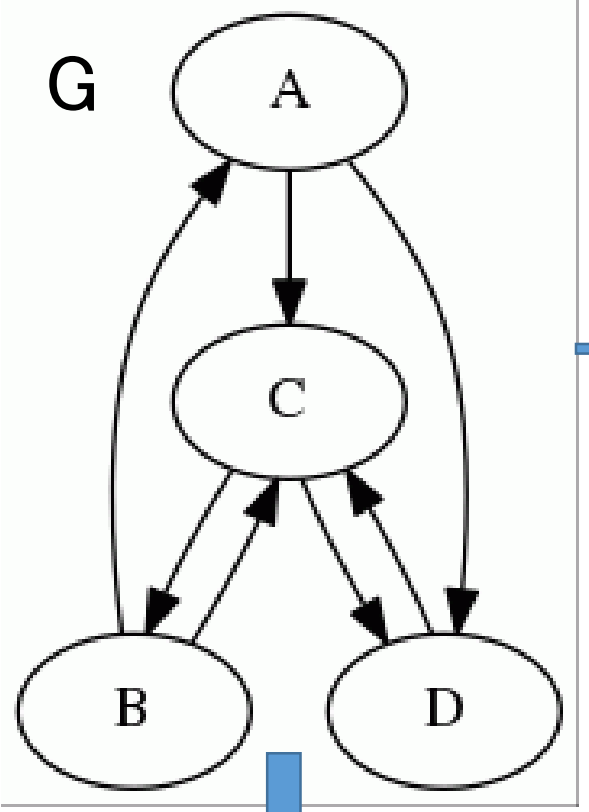


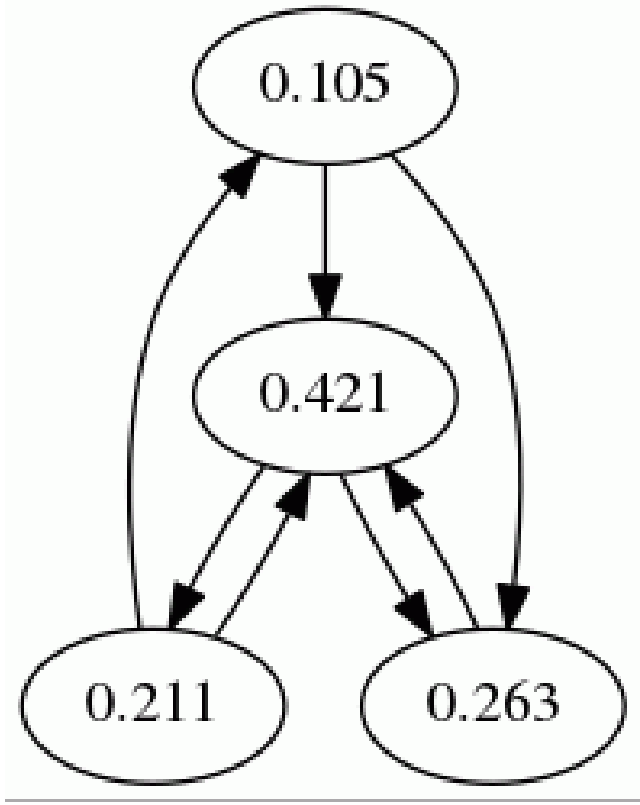
リンク構造 G が与えられたとする。頂点 X から r≧1本のリンクが$Y_1,Y_2,\cdots,Y_r$ に出ている時、(Y_i,X) 成分が 1/r である行列 M_G を考えよう（ここで i=1,2,…,r）。

M_G は、必ず固有値1を持つことが証明できる（easy）。
そこで、固有値1の固有ベクトルを求めてみよう。

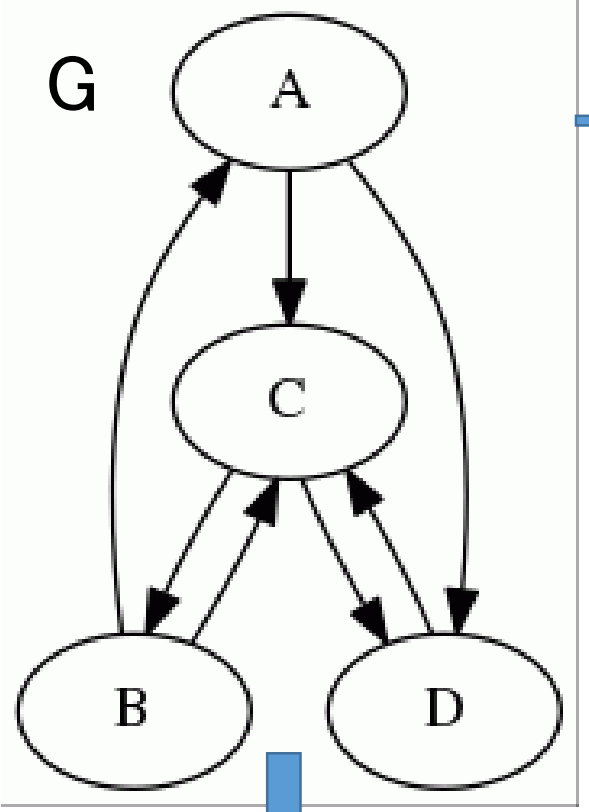
$$M_G = \begin{array}{c} \begin{matrix} A & B & C & D \end{matrix} \\ \begin{pmatrix} 0 & 1/2 & 0 & 0 \\ 0 & 0 & 1/2 & 0 \\ 1/2 & 1/2 & 0 & 1 \\ 1/2 & 0 & 1/2 & 0 \end{pmatrix} \end{array} \begin{matrix} A \\ B \\ C \\ D \end{matrix}$$

$$M_G \boldsymbol{v} = \boldsymbol{v}, \boldsymbol{v} \neq \boldsymbol{0}$$

$$\boldsymbol{v} = \frac{1}{19} \begin{pmatrix} 2 \\ 4 \\ 8 \\ 5 \end{pmatrix} t, t \neq 0$$



何故、この方法でうまくいく(or うまくいきそう)なのだろう？



方程式 $M_G v=v$ を書き直すと

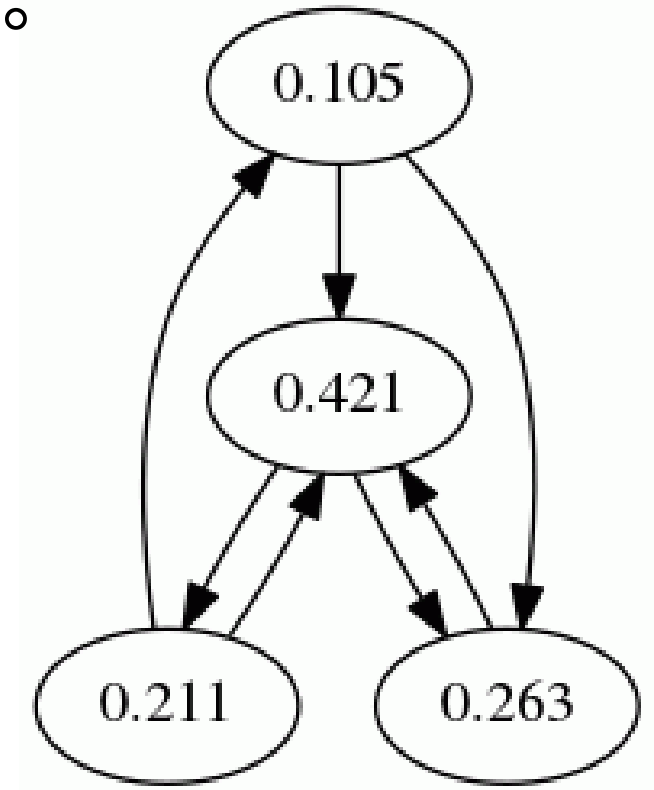
$$\begin{cases} x_A = \frac{1}{2}x_B \\ x_B = \frac{1}{2}x_C \\ x_C = \frac{1}{2}x_A + \frac{1}{2}x_B + x_D \\ x_D = \frac{1}{2}x_A + \frac{1}{2}x_C \end{cases}$$

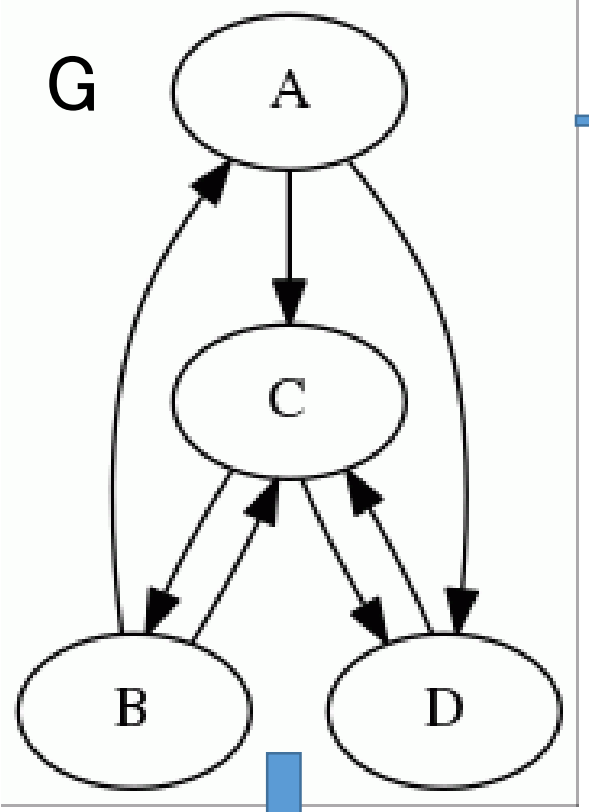
リンク構造を人気投票と思う。
すると、右辺は「投票の結果」
左辺は「人気」と解釈される。
方程式は「これらが整合的」
ということ。

$$M_G = \begin{array}{c} \begin{matrix} \text{A} & \text{B} & \text{C} & \text{D} \end{matrix} \\ \begin{pmatrix} 0 & 1/2 & 0 & 0 \\ 0 & 0 & 1/2 & 0 \\ 1/2 & 1/2 & 0 & 1 \\ 1/2 & 0 & 1/2 & 0 \end{pmatrix} \end{array} \begin{matrix} \text{A} \\ \text{B} \\ \text{C} \\ \text{D} \end{matrix}$$

$$M_G \boldsymbol{v} = \boldsymbol{v}, \boldsymbol{v} \neq \boldsymbol{0}$$

$$\boldsymbol{v} = \frac{1}{19}\begin{pmatrix} 2 \\ 4 \\ 8 \\ 5 \end{pmatrix} t, t \neq 0$$





しかし、さまざまな疑問が浮かぶ。

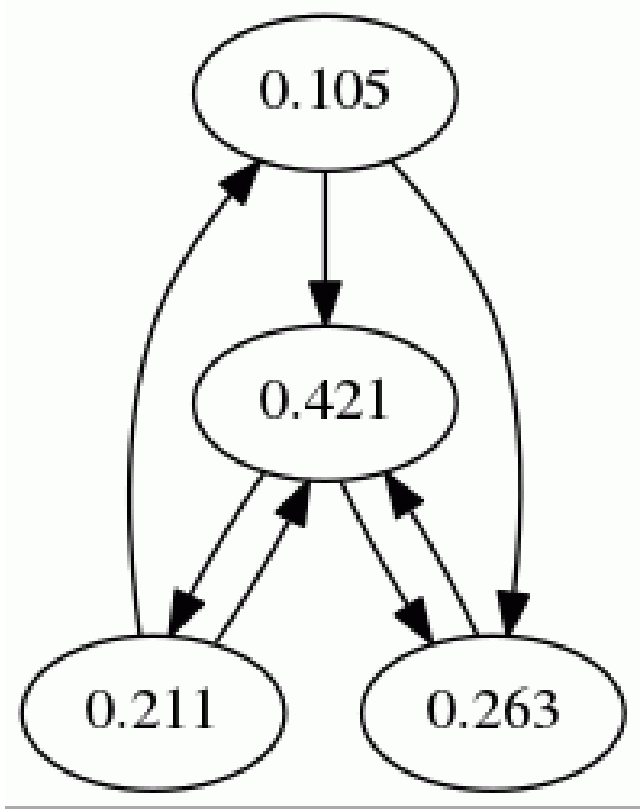
固有ベクトルは「本質的に1つ」なのだろうか？つまり固有値1の固有空間は、いつでも1次元なのか？

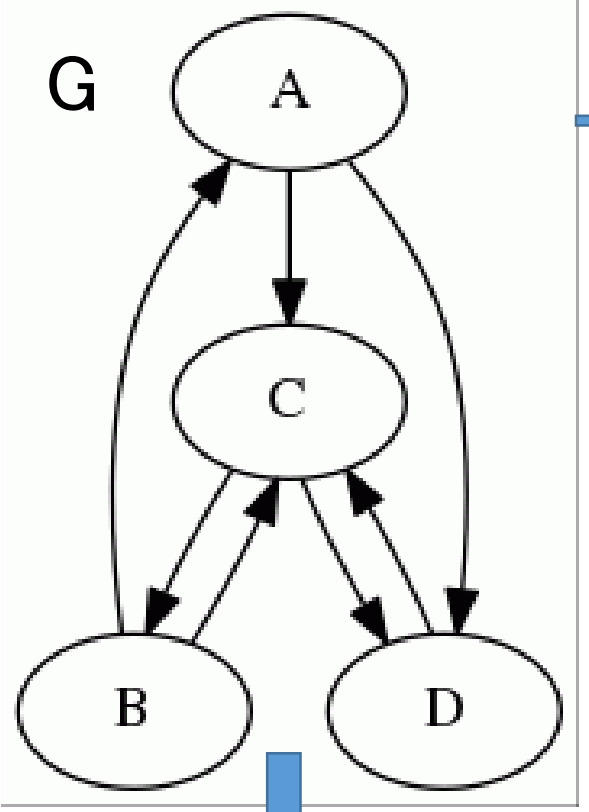
固有ベクトルは、一斉に≧0となるようにとれるか？（負の「ページの重要性」はちょっと気持ち悪い）

$$M_G = \begin{array}{c} \begin{matrix} A & B & C & D \end{matrix} \\ \begin{pmatrix} 0 & 1/2 & 0 & 0 \\ 0 & 0 & 1/2 & 0 \\ 1/2 & 1/2 & 0 & 1 \\ 1/2 & 0 & 1/2 & 0 \end{pmatrix} \end{array} \begin{matrix} A \\ B \\ C \\ D \end{matrix}$$

$$M_G \boldsymbol{v} = \boldsymbol{v}, \boldsymbol{v} \neq \boldsymbol{0}$$

$$\boldsymbol{v} = \frac{1}{19} \begin{pmatrix} 2 \\ 4 \\ 8 \\ 5 \end{pmatrix} t, t \neq 0$$





しかし、さまざまな疑問が浮かぶ。

固有ベクトルは「本質的に1つ」なのだろうか？つまり固有値1の固有空間は、いつでも1次元なのか？

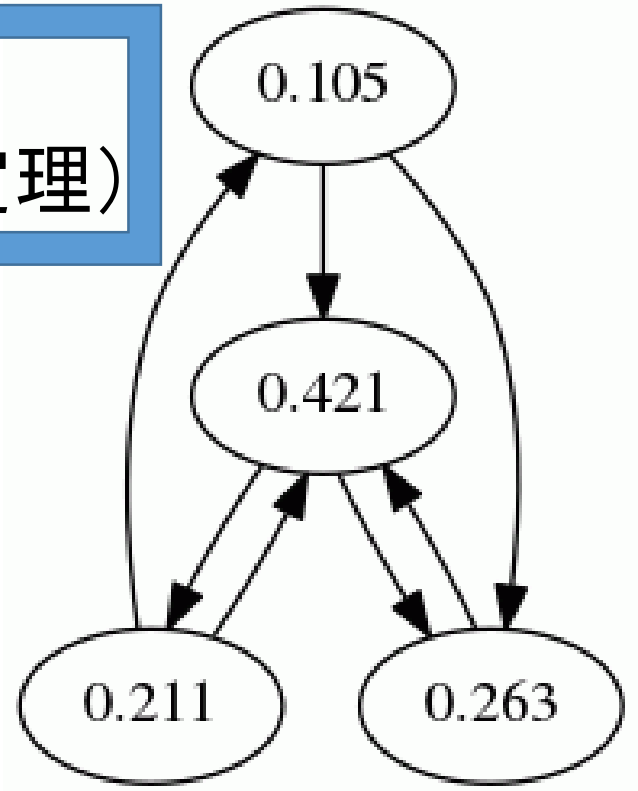
固有ベクトルは、一斉に≧0となるようにとれるか？（負の「ページの重要性」はちょっと気持ち悪い）

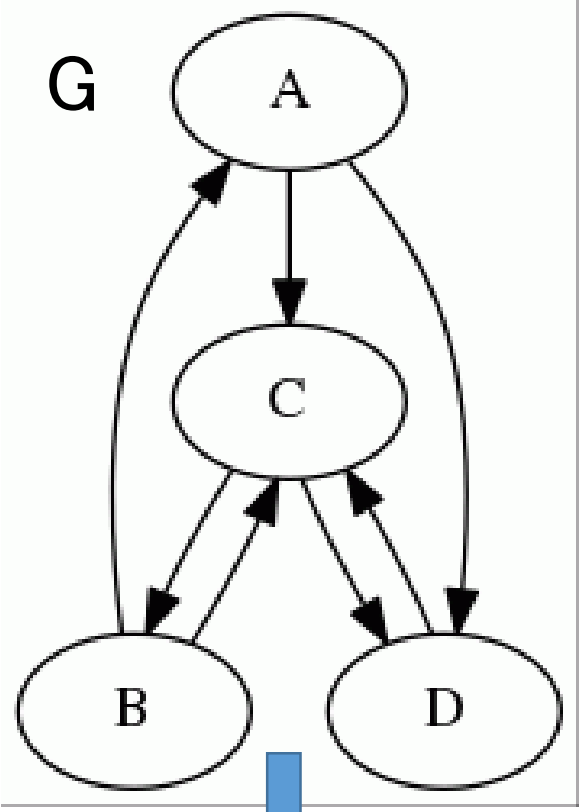
「グラフ G がつながっていれば」
大丈夫（ペロン・フロベニウスの定理）

$$M_G = \begin{array}{c} \begin{matrix} A & B & C & D \end{matrix} \\ \begin{pmatrix} 0 & 1/2 & 0 & 0 \\ 0 & 0 & 1/2 & 0 \\ 1/2 & 1/2 & 0 & 1 \\ 1/2 & 0 & 1/2 & 0 \end{pmatrix} \end{array} \begin{matrix} A \\ B \\ C \\ D \end{matrix}$$

$$M_G \boldsymbol{v} = \boldsymbol{v}, \boldsymbol{v} \neq \boldsymbol{0}$$

$$\boldsymbol{v} = \frac{1}{19} \begin{pmatrix} 2 \\ 4 \\ 8 \\ 5 \end{pmatrix} t, t \neq 0$$





ここまでは5月に説明しました。「つながり」を保証するため

$$(1-\alpha)M_G+\alpha\begin{pmatrix}1/4 & 1/4 & 1/4 & 1/4\\ 1/4 & 1/4 & 1/4 & 1/4\\ 1/4 & 1/4 & 1/4 & 1/4\\ 1/4 & 1/4 & 1/4 & 1/4\end{pmatrix}$$

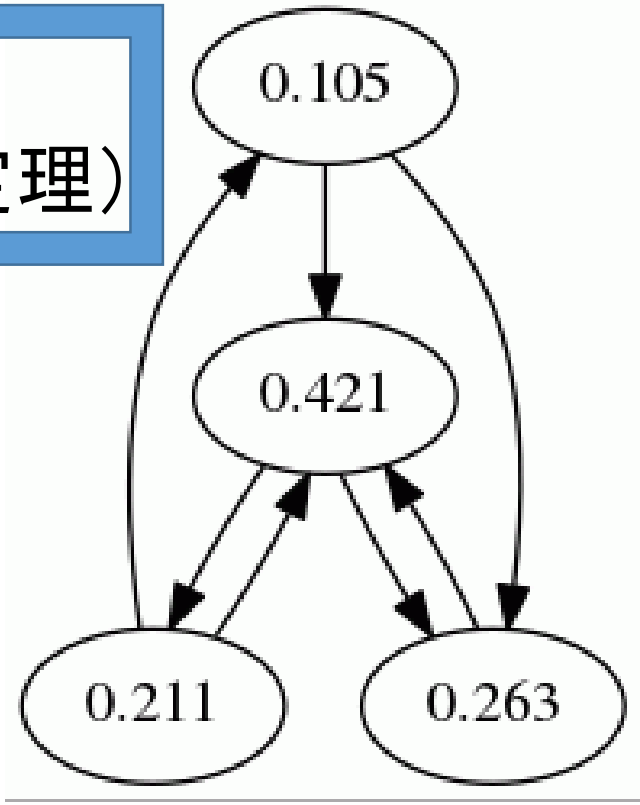
に、実際は補正するようです（0<α<1は経験で決める）。

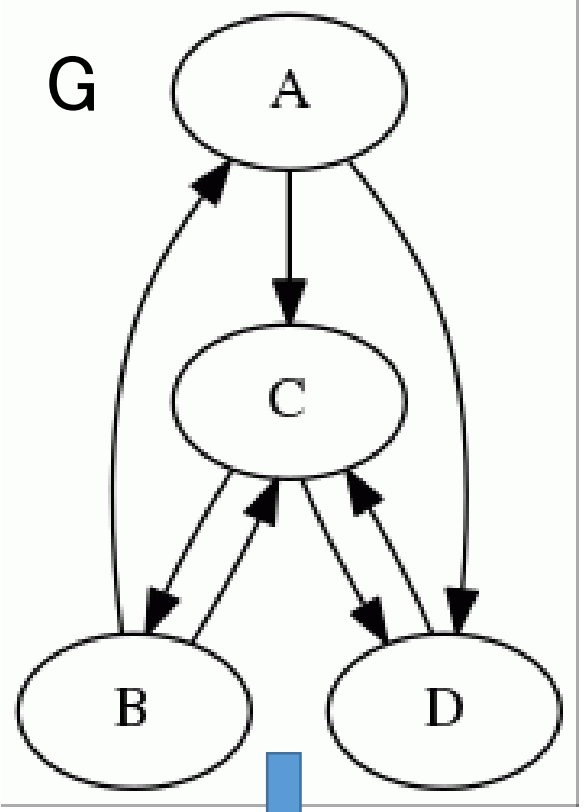
「グラフ G がつながっていれば」
大丈夫（ペロン・フロベニウスの定理）

$$M_G=\begin{array}{c}\begin{matrix}A & B & C & D\end{matrix}\\ \begin{pmatrix}0 & 1/2 & 0 & 0\\ 0 & 0 & 1/2 & 0\\ 1/2 & 1/2 & 0 & 1\\ 1/2 & 0 & 1/2 & 0\end{pmatrix}\end{array}\begin{matrix}A\\B\\C\\D\end{matrix}$$

$$M_G\boldsymbol{v}=\boldsymbol{v},\boldsymbol{v}\neq\boldsymbol{0}$$

$$\boldsymbol{v}=\frac{1}{19}\begin{pmatrix}2\\4\\8\\5\end{pmatrix}t,t\neq 0$$





ここまでは5月に説明しました。「つながり」を保証するため

$$(1-\alpha)M_G+\alpha\begin{pmatrix}1/4 & 1/4 & 1/4 & 1/4\\ 1/4 & 1/4 & 1/4 & 1/4\\ 1/4 & 1/4 & 1/4 & 1/4\\ 1/4 & 1/4 & 1/4 & 1/4\end{pmatrix}$$

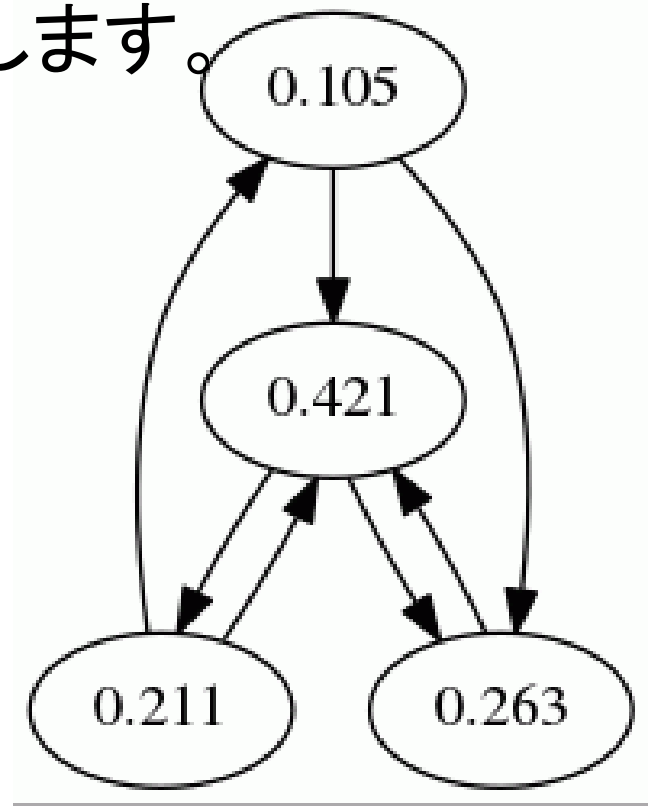
に、実際は補正するようです（0<α<1は経験で決める）。

今日は、この補正を別の視点から理解します。

$$M_G=\begin{array}{c}\begin{matrix}A & B & C & D\end{matrix}\\ \begin{pmatrix}0 & 1/2 & 0 & 0\\ 0 & 0 & 1/2 & 0\\ 1/2 & 1/2 & 0 & 1\\ 1/2 & 0 & 1/2 & 0\end{pmatrix}\end{array}\begin{matrix}A\\B\\C\\D\end{matrix}$$

$$M_G\boldsymbol{v}=\boldsymbol{v},\boldsymbol{v}\neq\boldsymbol{0}$$

$$\boldsymbol{v}=\frac{1}{19}\begin{pmatrix}2\\4\\8\\5\end{pmatrix}t,t\neq 0$$



定義：$n \times n$非負行列 $P=(p_{ij})_{1 \leqq i,j \leqq n}$ が確率行列とは、各列の和=1 となること。

例：

$$M_G = \begin{pmatrix} 0 & 1/2 & 0 & 0 \\ 0 & 0 & 1/2 & 0 \\ 1/2 & 1/2 & 0 & 1 \\ 1/2 & 0 & 1/2 & 0 \end{pmatrix} \begin{matrix} A \\ B \\ C \\ D \end{matrix}$$

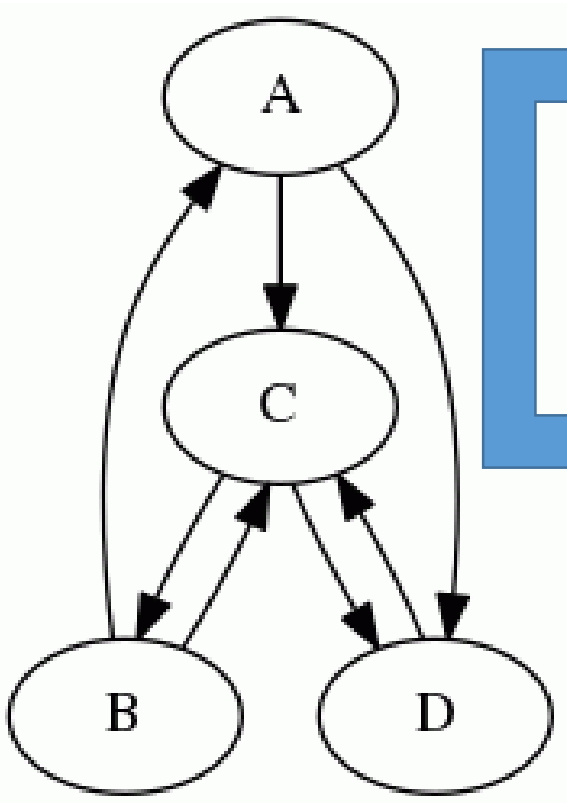
$$\begin{matrix} A & B & C & D \end{matrix}$$

確率行列は、確率の遷移のデータと思える。

例えば、m秒後に頂点Xにいる確率を$p_X(m)$とすると

$$\begin{pmatrix} p_A(m+1) \\ p_B(m+1) \\ p_C(m+1) \\ p_D(m+1) \end{pmatrix} = \begin{pmatrix} 0 & 1/2 & 0 & 0 \\ 0 & 0 & 1/2 & 0 \\ 1/2 & 1/2 & 0 & 1 \\ 1/2 & 0 & 1/2 & 0 \end{pmatrix} \begin{pmatrix} p_A(m) \\ p_B(m) \\ p_C(m) \\ p_D(m) \end{pmatrix}$$

G



定理：P は固有値1をもつが、これがただ一つの絶対値最大固有値で、かつ固有値1の固有空間が1次元のとき P^n の各列は、「固有値1の固有ベクトル」or 0 に収束する。

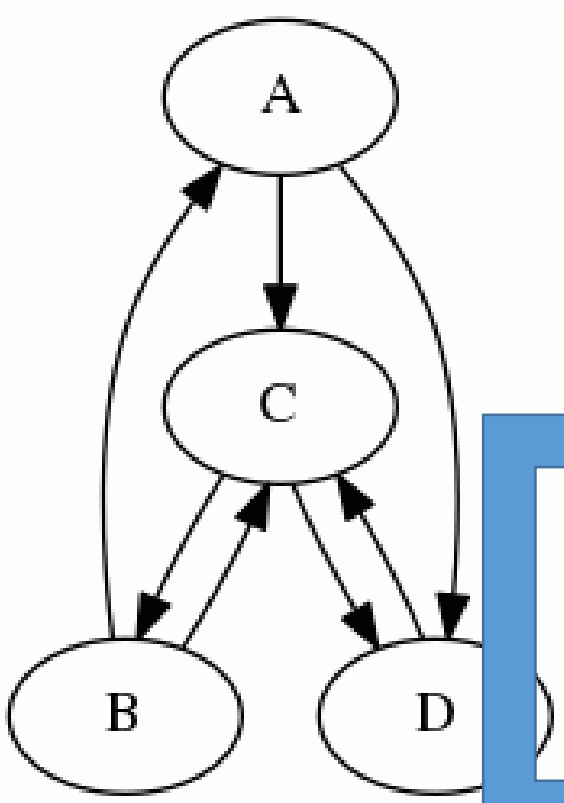
証明は5月にはできませんでしたが、今だと可能なのでせっかくなのでやってみることにします（10分くらい?）。

ページランクの方程式 $M_G v = v$ (i.e., vは固有値1の固有ベクトル)は、5月に「人気投票の整合性」として説明したが、「ランダムウォークの定常状態」でもあることがわかった！補正は「ランダムジャンプ」を付加したと思える。

$$(1-\alpha)\begin{pmatrix} 0 & 1/2 & 0 & 0 \\ 0 & 0 & 1/2 & 0 \\ 1/2 & 1/2 & 0 & 1 \\ 1/2 & 0 & 1/2 & 0 \end{pmatrix} + \alpha \begin{pmatrix} 1/4 & 1/4 & 1/4 & 1/4 \\ 1/4 & 1/4 & 1/4 & 1/4 \\ 1/4 & 1/4 & 1/4 & 1/4 \\ 1/4 & 1/4 & 1/4 & 1/4 \end{pmatrix}$$

ランダムにリンクをたどる　リンクと無関係に飛ぶ（ベーシックインカム）

G



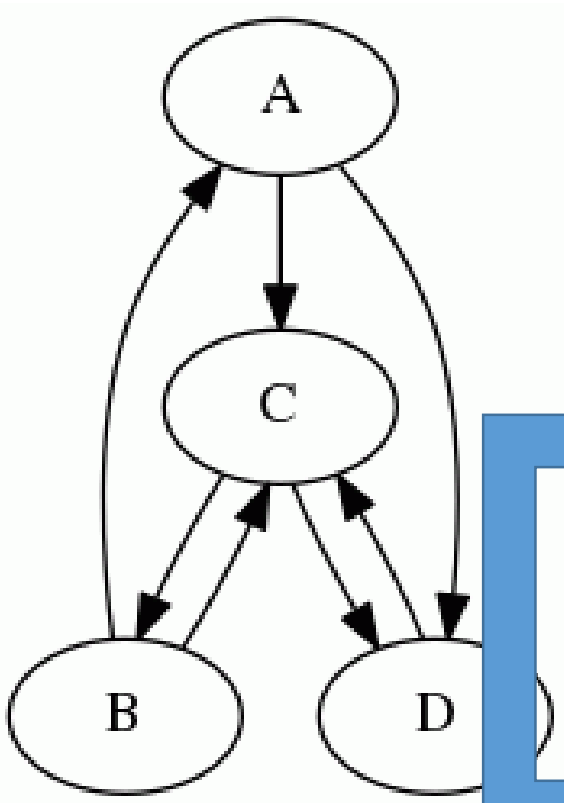
定理：P は固有値1をもつが、これがただ一つの絶対値最大固有値で、かつ固有値1の固有空間が1次元のとき P^m の各列は、「固有値1の固有ベクトル」or 0 に収束する。

ページランクの方程式 $M_G v = v$ (i.e., vは固有値1の固有ベクトル)は、5月に「人気投票の整合性」として説明したが、「ランダムウォークの定常状態」でもあることがわかった！補正は「ランダムジャンプ」を付加したと思える。

$$(1-\alpha)\begin{pmatrix} 0 & 1/2 & 0 & 0 \\ 0 & 0 & 1/2 & 0 \\ 1/2 & 1/2 & 0 & 1 \\ 1/2 & 0 & 1/2 & 0 \end{pmatrix} + \alpha \begin{pmatrix} 1/4 & 1/4 & 1/4 & 1/4 \\ 1/4 & 1/4 & 1/4 & 1/4 \\ 1/4 & 1/4 & 1/4 & 1/4 \\ 1/4 & 1/4 & 1/4 & 1/4 \end{pmatrix}$$

固有値は1(重複度1)と0(重複度3)

G



また補正には「第2固有値の絶対値を小さくすることで収束を早くする」という効果もある。

定理: P は固有値1をもつが、これがただ一つの絶対値最大固有値で、かつ固有値1の固有空間が1次元のとき P^m の各列は、「固有値1の固有ベクトル」 or 0 に収束する。

教訓：線形代数は重要。とくに行列の掛け算は重要！

⟹ 大きな行列を、高速に掛け算したい。

教訓：線形代数は重要。とくに行列の掛け算は重要！

⇒ 大きな行列を、高速に掛け算したい。

理論でがんばる編：もしも加減法より乗法が「重たい」数体系では…

定理(Strassen,1969)：*2x2* 行列の掛け算は、通常8回のスカラ乗算が必要だが、実は7回で済ませる方法がある（ただし、加減法の回数は増える）

教訓：線形代数は重要。とくに行列の掛け算は重要！

⇒ 大きな行列を、高速に掛け算したい。

理論でがんばる編：もしも加減法より乗法が「重たい」数体系では…

定理(Strassen,1969)：*2x2* 行列の掛け算は、通常8回のスカラ乗算が必要だが、実は7回で済ませる方法がある（ただし、加減法の回数は増える）

$n \times n$ 行列の掛け算を普通に行うと、n^3 回の乗算が必要になる。Strassenを繰り返し用いることで、”だいたい”この 3 を $\log_2 7 = 2.807$ にできる。

教訓：線形代数は重要。とくに行列の掛け算は重要！

⇒ 大きな行列を、高速に掛け算したい。

理論でがんばる編：もしも加減法より乗法が「重たい」数体系では…

定理(Strassen,1969)：*2x2* 行列の掛け算は、通常8回のスカラ乗算が必要だが、実は7回で済ませる方法がある（ただし、加減法の回数は増える）

$n \times n$ 行列の掛け算を普通に行うと、n^3 回の乗算が必要になる。Strassenを繰り返し用いることで、”だいたい”この 3 を $\log_2 7 = 2.807$ にできる。

これは現実にも使われている。FFLAS-FFPACKという有限体での行列演算に特化したC++のライブラリがあるが、ビルド時に—enable-optimization をつけると、Strassenが通常の行列乗算に勝る閾値を調べて、組み込む。

教訓：線形代数は重要。とくに行列の掛け算は重要！

⇒ 大きな行列を、高速に掛け算したい。

理論でがんばる編：もしも加減法より乗法が「重たい」数体系では…

定理(Strassen,1969)：*2x2* 行列の掛け算は、通常8回のスカラ乗算が必要だが、実は7回で済ませる方法がある（ただし、加減法の回数は増える）

$n \times n$ 行列の掛け算を普通に行うと、n^3 回の乗算が必要になる。Strassenを繰り返し用いることで、”だいたい”この 3 を $\log_2 7=2.807$ にできる。

これは現実にも使われている。FFLAS-FFPACKという有限体での行列演算に特化したC++のライブラリがあるが、ビルド時に--enable-optimization をつけると、Strassenが通常の行列乗算に勝る閾値を調べて、組み込む。

この $\log_2 7=2.807$ は、究極には $\forall \varepsilon > 0,\ 2+\varepsilon$ だろうと予想されている。

F.LeGall（arXiv:1401.7714）においてε=0.3728…とできることは示されている。

教訓：線形代数は重要。とくに行列の掛け算は重要！

⇒ 大きな行列を、高速に掛け算したい。

実装でがんばる編：あまりにもよくある要求なので、APIが決められています。

⇒ BLAS (Basic Linear Algebra Subprograms,1979)

教訓：線形代数は重要。とくに行列の掛け算は重要！

⇒ 大きな行列を、高速に掛け算したい。

実装でがんばる編：あまりにもよくある要求なので、APIが決められています。

⇒ BLAS (Basic Linear Algebra Subprograms,1979)

有名な実装に、後藤BLAS→OpenBLAS や IMK（Intel Math Kernel）がある。

個人的に高級言語で書いた行列演算実装をこれらに変えるだけで、10倍は速くなります（何故なのか理解するには、メモリの階層構造の理解が必要）。

教訓：線形代数は重要。とくに行列の掛け算は重要！

⇒ 大きな行列を、高速に掛け算したい。

実装でがんばる編：あまりにもよくある要求なので、APIが決められています。

⇒ BLAS (Basic Linear Algebra Subprograms,1979)

有名な実装に、後藤BLAS→OpenBLAS や IMK（Intel Math Kernel）がある。

個人的に高級言語で書いた行列演算実装をこれらに変えるだけで、10倍は速くなります（何故なのか理解するには、メモリの階層構造の理解が必要）。

BLASで倍精度行列積のAPIは dgemm である。$n \times n$行列の掛け算を実行し、t秒かかったとすると、そのシステムのFLOPS（Floating-point Operations Per Seconds)は $(n^2(n-1)+n^3)/t$ FLOPSであると見積もれます。

（c.f.「LAPACK/BLAS入門（森北出版）」）。詳しく知りませんが、Top500でも線形代数演算（LINPACKなど）を通じて、スパコンの性能を競っているようです

教訓：線形代数は重要。とくに行列の掛け算は重要！

⟹ 大きな行列を、高速に掛け算したい。

教訓：線形代数は重要。とくに行列の掛け算は重要！

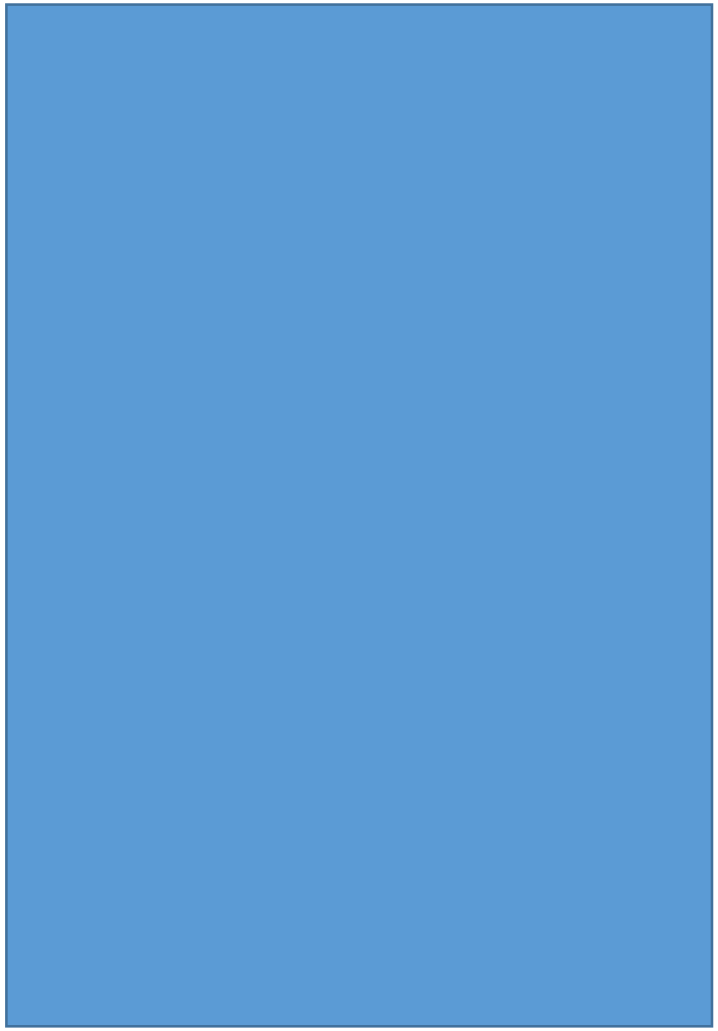
⇒ 大きな行列を、高速に掛け算したい。

⇒ FLOPSの大きいスパコンをつくる！こんな本があります：

教訓：線形代数は重要。とくに行列の掛け算は重要！

⇒ 大きな行列を、高速に掛け算したい。

⇒ FLOPSの大きいスパコンをつくる！こんな本があります：

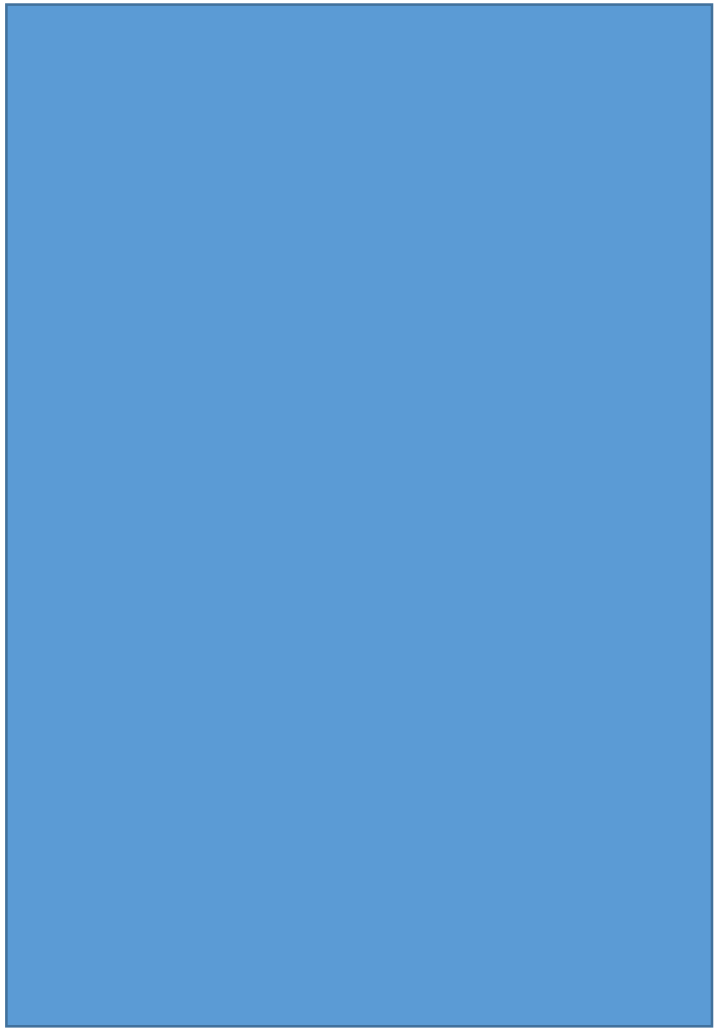


うさんくさい（失礼！）。でも魅力的。

教訓：線形代数は重要。とくに行列の掛け算は重要！

⇒ 大きな行列を、高速に掛け算したい。

⇒ FLOPSの大きいスパコンをつくる！こんな本があります：



うさんくさい（失礼！）。でも魅力的。

先週、↓なニュースがありました：

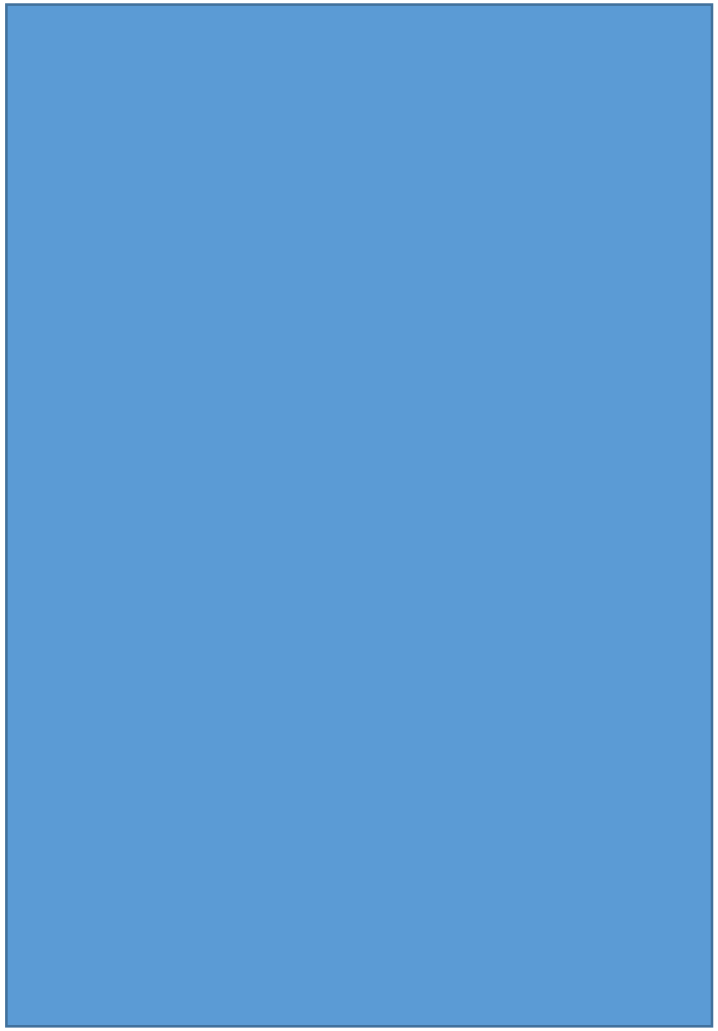
Rank	System	Cores	Rmax (TFlop/s)	Rpeak (TFlop/s)	Power (kW)
1	**Sunway TaihuLight** - Sunway MPP, Sunway SW26010 260C 1.45GHz, Sunway , NRCPC National Supercomputing Center in Wuxi China	10,649,600	93,014.6	125,435.9	15,371
2	**Tianhe-2 (MilkyWay-2)** - TH-IVB-FEP Cluster, Intel Xeon E5-2692 12C 2.200GHz, TH Express-2, Intel Xeon Phi 31S1P , NUDT National Super Computer Center in Guangzhou China	3,120,000	33,862.7	54,902.4	17,808
3	**Piz Daint** - Cray XC50, Xeon E5-2690v3 12C 2.6GHz, Aries interconnect , NVIDIA Tesla P100 , Cray Inc. Swiss National Supercomputing Centre (CSCS) Switzerland	361,760	19,590.0	25,326.3	2,272
4	**Gyoukou** - ZettaScaler-2.2 HPC system, Xeon D-1571 16C 1.3GHz, Infiniband EDR, PEZY-SC2 700Mhz , ExaScaler Japan Agency for Marine-Earth Science and Technology Japan	19,860,000	19,135.8	28,192.0	1,350
5	**Titan** - Cray XK7, Opteron 6274 16C 2.200GHz, Cray Gemini interconnect, NVIDIA K20x , Cray Inc. DOE/SC/Oak Ridge National Laboratory United States	560,640	17,590.0	27,112.5	8,209
6	**Sequoia** - BlueGene/Q, Power BQC 16C 1.60 GHz, Custom , IBM DOE/NNSA/LLNL United States	1,572,864	17,173.2	20,132.7	7,890
7	**Trinity** - Cray XC40, Intel Xeon Phi 7250 68C 1.4GHz, Aries interconnect , Cray Inc. DOE/NNSA/LANL/SNL United States	979,968	14,137.3	43,902.6	3,844

Rank	TOP500 Rank	System	Cores	Rmax (TFlop/s)	Power (kW)	Power Efficiency (GFlops/watts)
1	259	**Shoubu system B** - ZettaScaler-2.2, Xeon D-1571 16C 1.3GHz, Infiniband EDR, PEZY-SC2 , PEZY Computing / Exascaler Inc. Advanced Center for Computing and Communication, RIKEN Japan	794,400	842.0	50	17.009
2	307	**Suiren2** - ZettaScaler-2.2, Xeon D-1571 16C 1.3GHz, Infiniband EDR, PEZY-SC2 , PEZY Computing / Exascaler Inc. High Energy Accelerator Research Organization /KEK Japan	762,624	788.2	47	16.759
3	276	**Sakura** - ZettaScaler-2.2, Xeon E5-2618Lv3 8C 2.3GHz, Infiniband EDR, PEZY-SC2 , PEZY Computing / Exascaler Inc. PEZY Computing K.K. Japan	794,400	824.7	50	16.657
4	149	**DGX SaturnV Volta** - NVIDIA DGX-1 Volta36, Xeon E5-2698v4 20C 2.2GHz, Infiniband EDR, NVIDIA Tesla V100 , Nvidia NVIDIA Corporation United States	22,440	1,070.0	97	15.113
5	4	**Gyoukou** - ZettaScaler-2.2 HPC system, Xeon D-1571 16C 1.3GHz, Infiniband EDR, PEZY-SC2 700Mhz , ExaScaler Japan Agency for Marine-Earth Science and Technology Japan	19,860,000	19,135.8	1,350	14.173
6	13	**TSUBAME3.0** - SGI ICE XA, IP139-SXM2, Xeon	135,828	8,125.0	792	13.704

教訓：線形代数は重要。とくに行列の掛け算は重要！

⇒ 大きな行列を、高速に掛け算したい。

⇒ FLOPSの大きいスパコンをつくる！こんな本があります：



うさんくさい（失礼！）。でも魅力的。

先週、↓なニュースがありました：

PEZYは、peta＝10^{15}、exa＝10^{18}、

zetta＝10^{21}、yotta＝10^{24}からとったそうです。

スパコンは基本、行列演算のためにつくる

と理解して問題ないと思います。

今日は行列の応用についてふれました。

来週は、微積と線形代数の先、特にリクエストがあった内容にふれようと思っています。