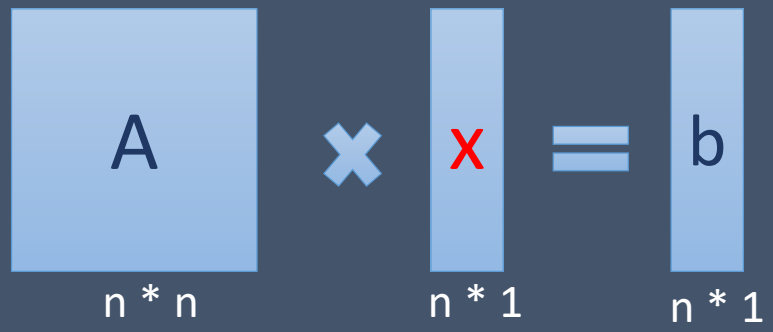
HPL实验报告

HPL - High Performance Linpack

- Linpack是国际计算机浮点性能基准测试
  - HPL即高性能Linpack，是Linpack的一种
- HPL的特点
  - 求解稠密线性代数方程组的阶数没有限制
  - 求解问题的规模可以改变
  - 可进行除基本算法之外的任何优化

To Solve

线性矩阵求解



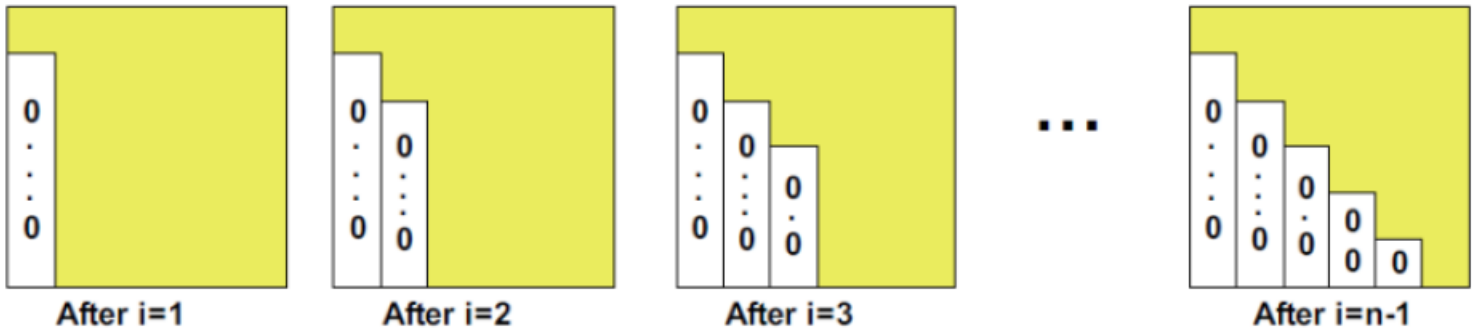
线性方程求解 – 高斯消元

- **初等变换**

  A → 上三角矩阵

  A(j, i) – c * A(i, i) = 0

  c为适当的系数



(James Demmel: Application of Parallel Computers, Lecture 14)

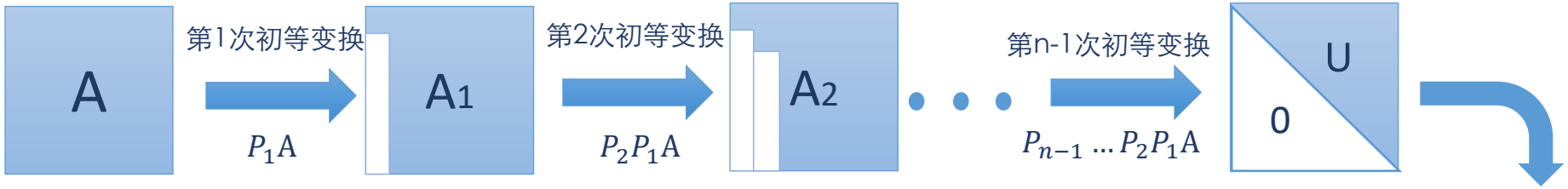
- 问题 若A(i, i) 为0，无法计算结果

$$A = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

行交换 (Pivot)

每次计算之前，设A(k, i)为第i列最大的值，交换第i、k列，使A(i, i)为最大值

线性方程求解－LU分解



$$P_{n-1}\dots P_2P_1A = \mathrm{U}$$

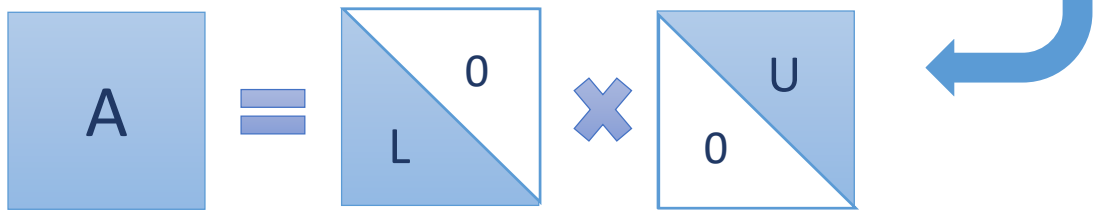
$$A = \underbrace{P_{n-1}^{-1}\dots P_2^{-1}P_1^{-1}}_{L}U$$

$$A = LU$$

代入$Ax = b$,

得,

$$\boldsymbol{x = U^{-1}L^{-1}b}$$



Before HPL

- **分块LU分解**

A_{00}	A_{01}	A_{02}
A_{10}	A_{11}	A_{12}
A_{20}	A_{21}	A_{22}

=

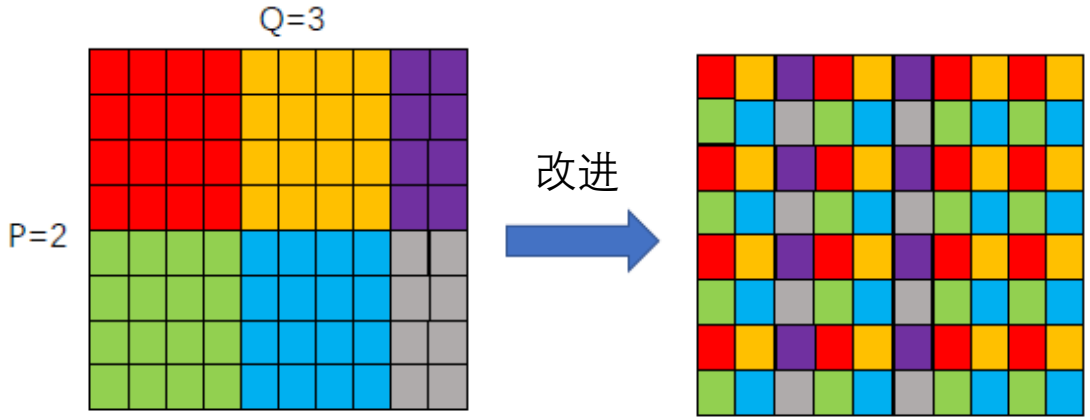
L_{00}	0	0
L_{10}	L_{11}	0
L_{20}	L_{21}	L_{22}

U_{00}	U_{01}	U_{02}
0	U_{11}	U_{12}
0	0	U_{22}

$$A_{00} = L_{00}U_{00} \qquad A_{01} = L_{00}U_{01}$$
$$A_{10} = L_{10}U_{00} \qquad A_{11} = L_{10}U_{01} + L_{11}U_{11}$$
$$A_{21} = L_{20}U_{00} \qquad A_{21} = L_{20}U_{01} + L_{21}U_{11}$$

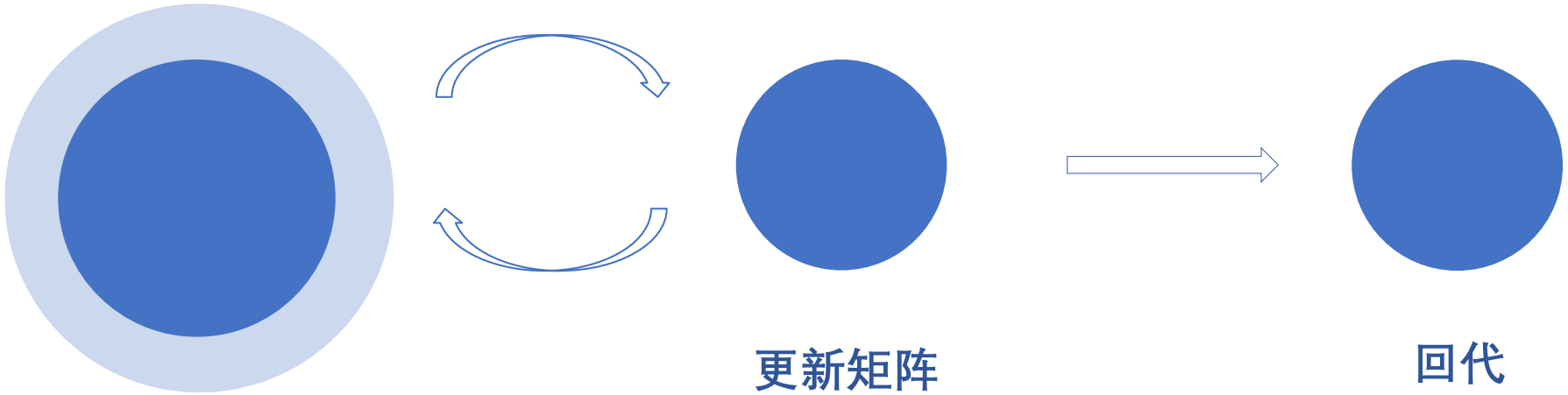
$$A_{02} = L_{00}U_{02}$$
$$A_{12} = L_{10}U_{02} + L_{11}U_{12}$$
$$A_{22} = L_{20}U_{02} + L_{21}U_{12} + L_{22}U_{22}$$

- **数据划分**



- 相同颜色的网格属于同一进程。
- 分散进程，提高效率。

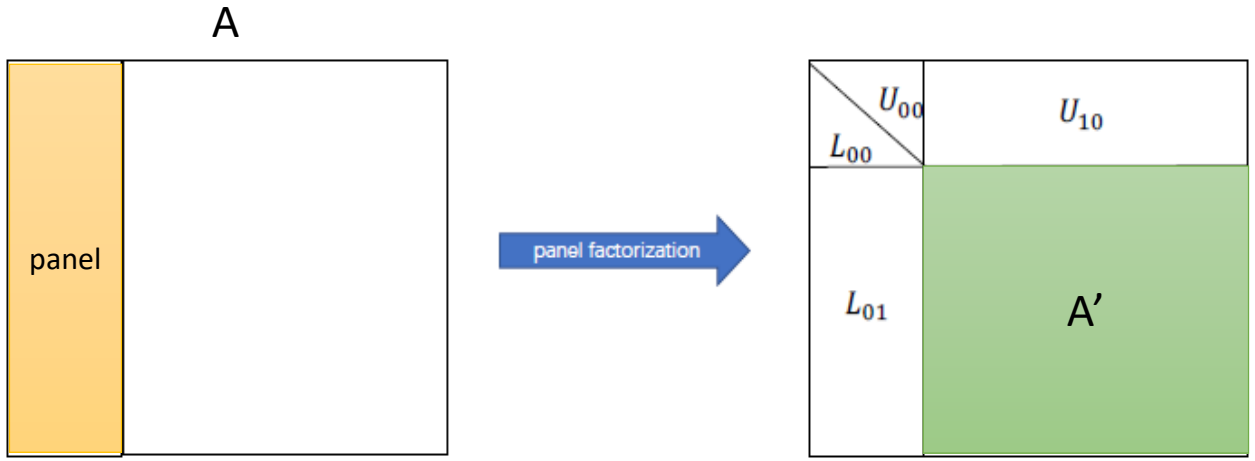
HPL算法



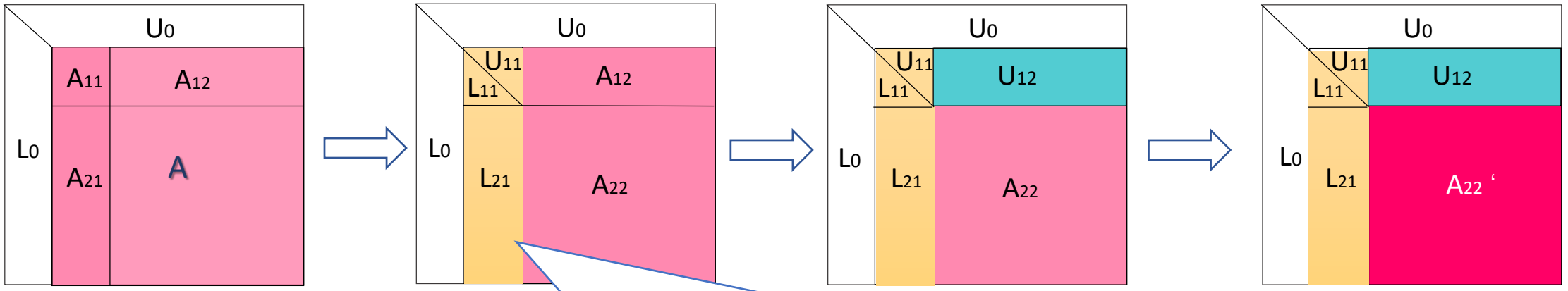
Panel 分解

Panel 分解算法

- Right-Looking Algorithm
- Left-Looking Algorithm
- Crout Algorithm



Right-Looking Algorithm



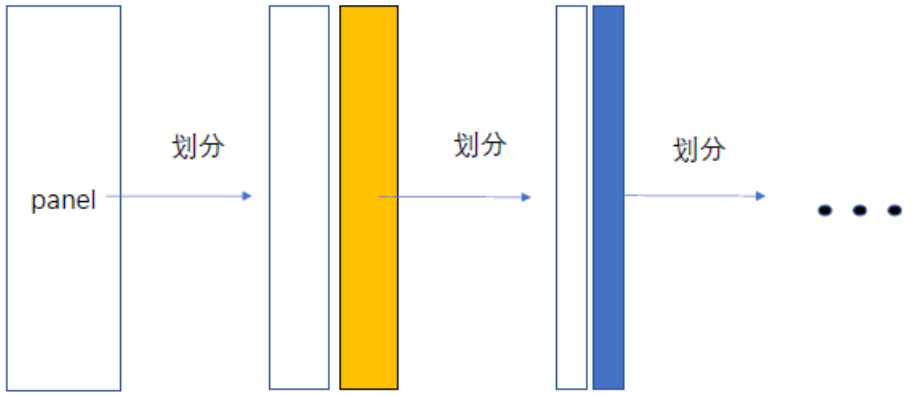
调用SWAP算法，选取
将 A_{11} LU分解为L_{11}和
求解U_{11}：

$$A_{11} = L_{10}U_{01} +$$
$$A_{21} = L_{20}U_{01} +$$

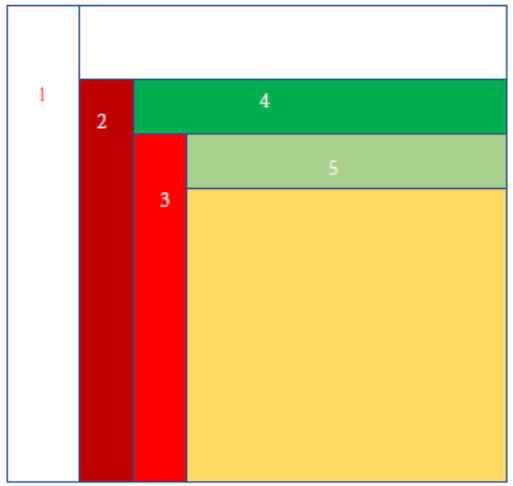
红色为已知量

$$\begin{bmatrix} A_{11} \\ A_{21} \end{bmatrix} = \begin{bmatrix} L_{11} \\ L_{21} \end{bmatrix}$$

- **内层Panel分解**
  - 递归过程



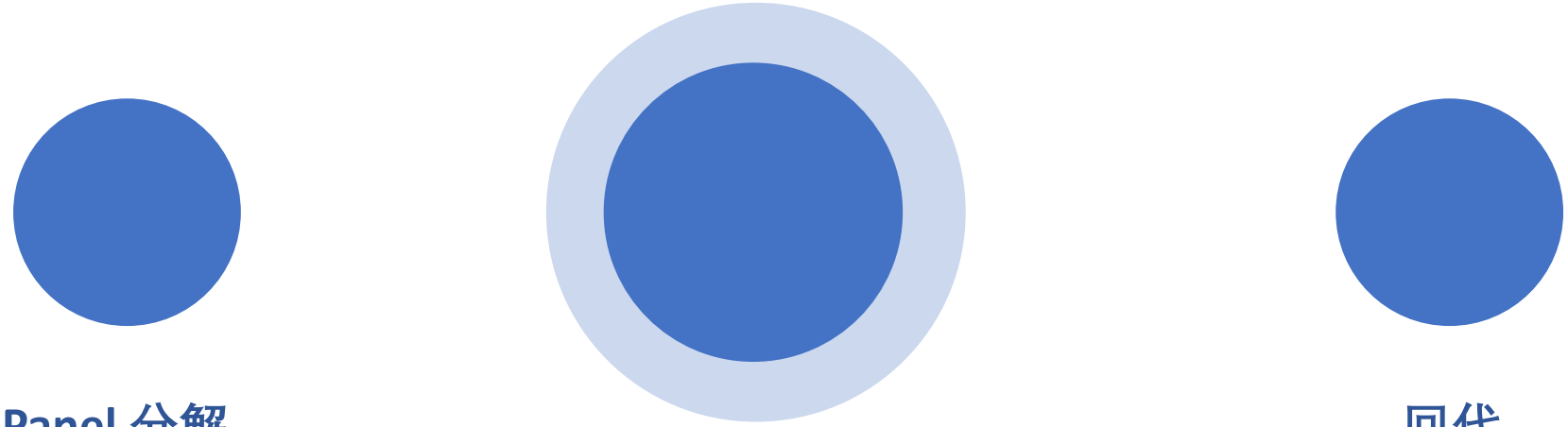
Panel 分解 – Look Head



在本轮进行panel分解的同时，更新下一个panel
下一轮可直接进行panel分解，节省时间

Look Head深度为可调参数DEPTH，一般为1或2

HPL算法



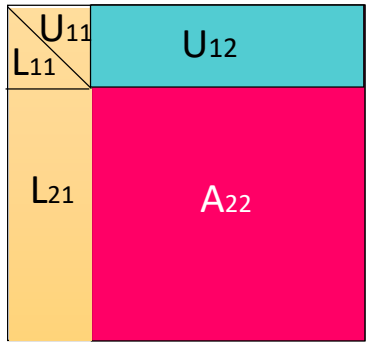
Panel 分解

更新矩阵

回代

更新矩阵

把U行交换的结果应用到矩阵其他元素



1. **Swapping**
   行交换算法：Binary-exchange、 Long、 Mix
2. **Trsm**
   通过下三角矩阵L_{11}更新U_{12}
3. **gemm**
   将U广播到其余每一行的进程

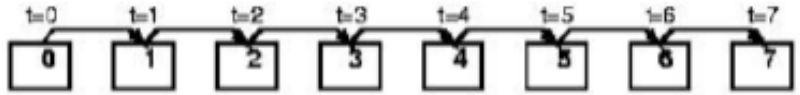
Panel Broadcast

1. Increasing-ring
2. Increasing-ring(modified)
3. Increasing-2-ring
4. Increasing-2-ring(modified)
5. Long(bandwidth reducing)
6. Long(bandwidth reducing modified)

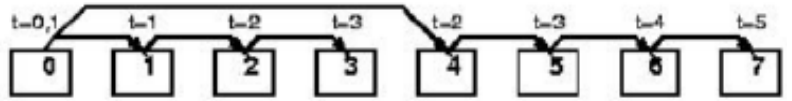
每个Panel进行LU分解后，需将这个panel的信息传给其余列。

Panel Broadcast

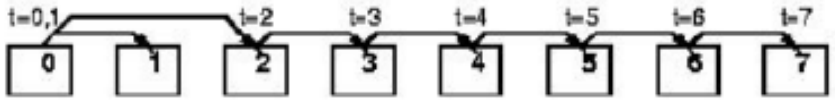
1-ring



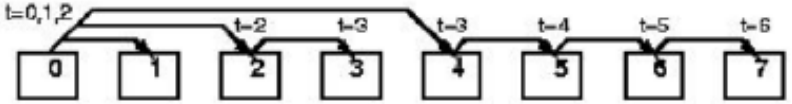
1-ring modified



2-ring



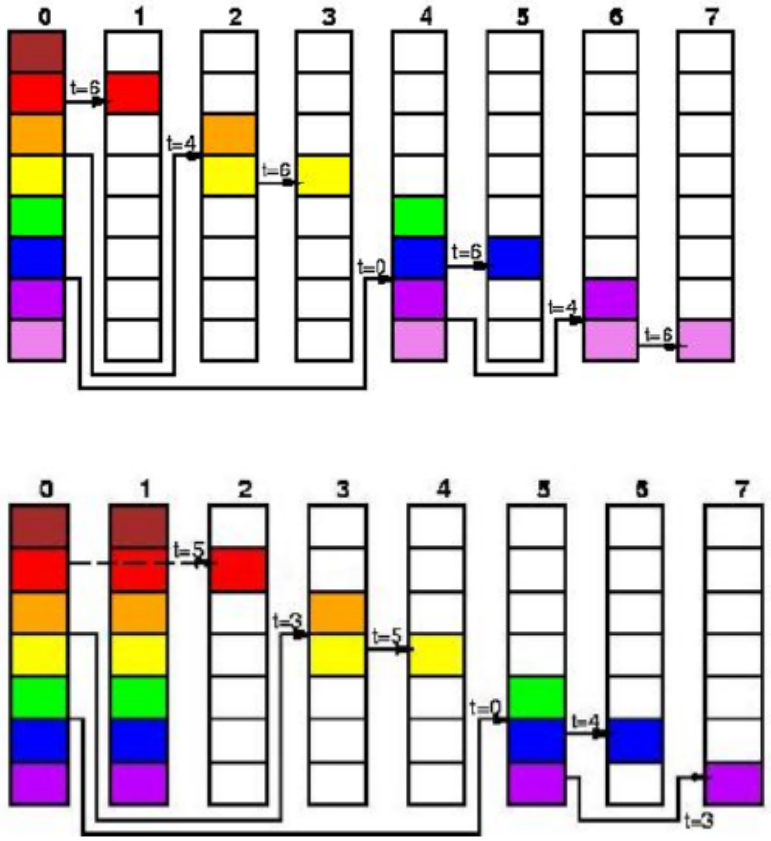
2-ring modified

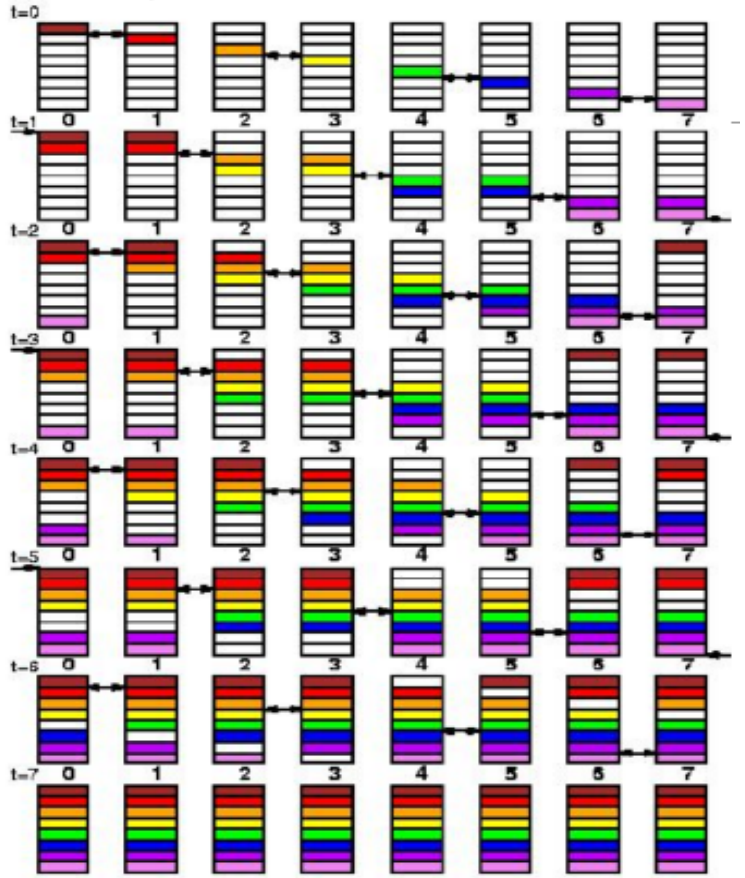


Panel Broadcast

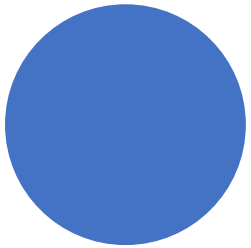
Long

Long modified

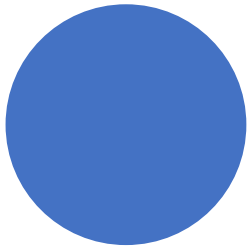




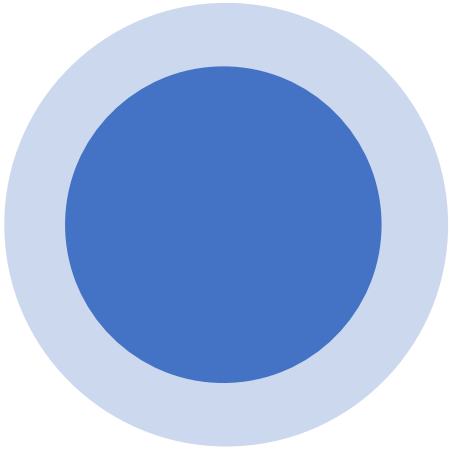
HPL算法



Panel 分解



更新矩阵



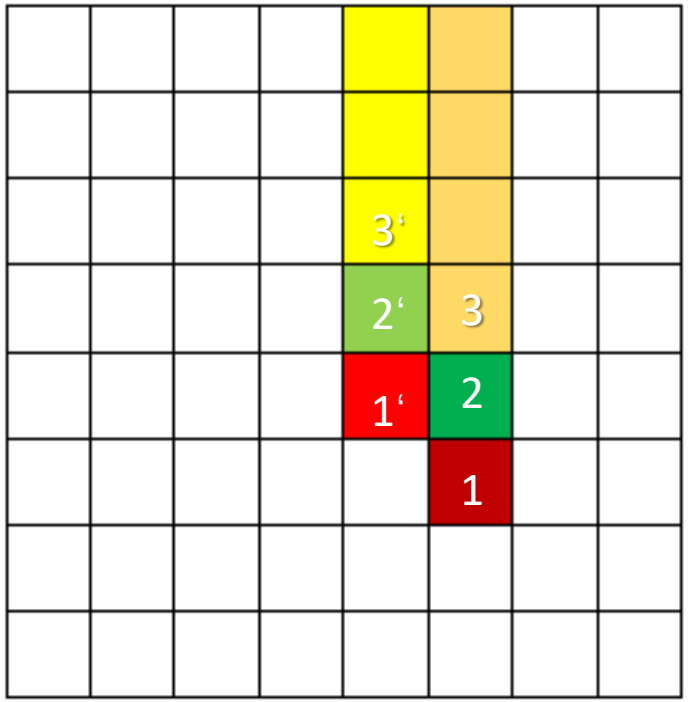
回代

Backward Substitution

1 调用dstrv，更新对角线上的矩阵块

2 调用dgemv，更新下一个对角矩阵块

3 更新这个panel，传递到下一个panel



Thanks