SC16@Salt Lake
HPCG BoF@Room 335E

HPCG Performance Improvement on the K computer ～short introduction～

Kiyoshi Kumahata[1), Kazuo Minami[1)]
Akira Hosoi[2)], Ikuo Miyoshi[2)]
1) RIKEN AICS
2) FUJITSU LIMITED

Our previous code until SC15

Our previous tuned code marked 0.461 PFLOPS for SC15

SC15 score

Rank	Computer	Country	HPL PFLOPS	HPCG PFLOPS	Ratio to HPL %
1	Tianhe-2	China	33.86	0.580	1.7%
2	K computer	Japan	10.51	0.461	4.4%
3	Titan	USA	17.59	0.322	1.8%
4	Trinity	USA	8.10	0.183	2.3%
5	Mira	USA	8.59	0.167	1.9%

http://www.hpcg-benchmark.org/custom/index.html?lid=155&slid=282

Bandwidth on the K computer@Compute Node

	Theoretical	STREAM	SPMV	SYMGS
GB/s	64	Practical limit 46	48	44

It is impossible to improve the bandwidth.

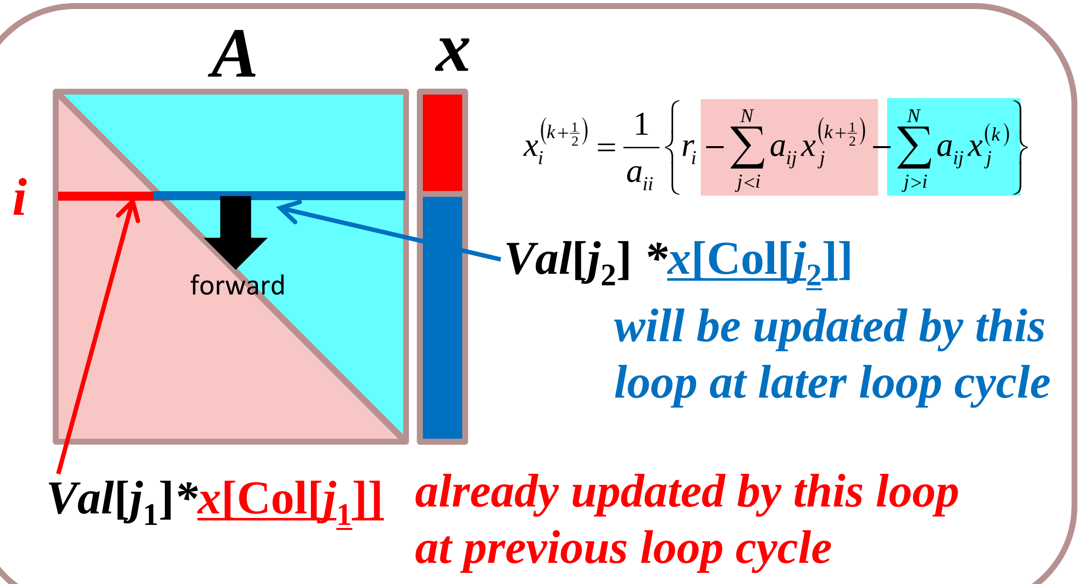
To get more score, we have to use another way, especially hot kernel SYMGS

Symmetric Gauss-Seidel

- Structure of Original Symmetric Gauss-Seidel

Forward Loop

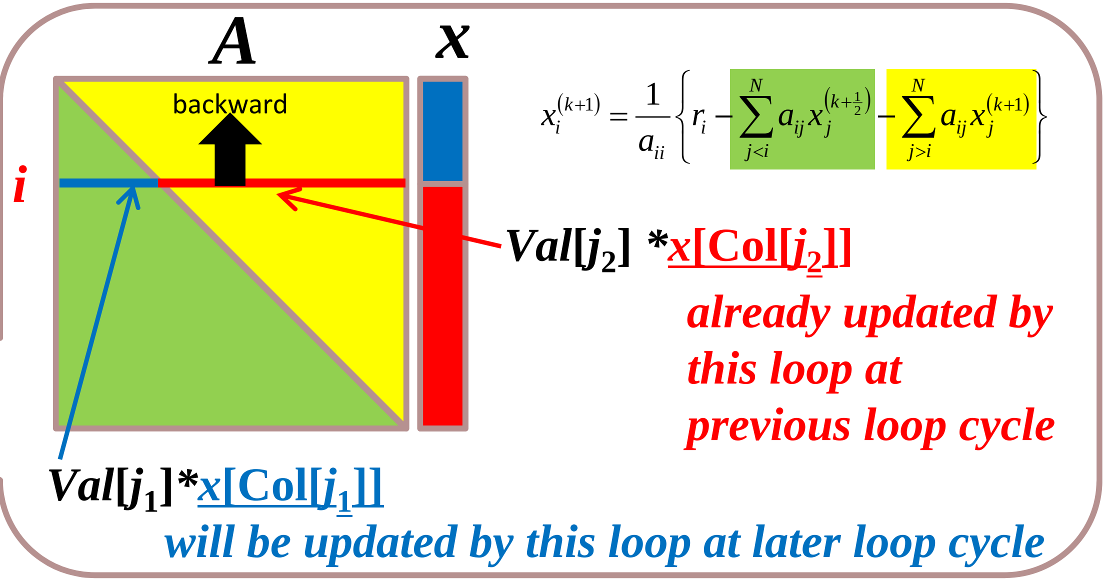
```
for (int i = 0; i < nrow; i++) {
            …
  double sum = r[i];
  for (int j = 0; j < nzInRow[i]; j++)
    sum -= Val[j]*x[Col[j]];
   sum += x[i]*Diag;
   x[i] = sum/Diag;
}
```



$$x_i^{(k+\frac{1}{2})} = \frac{1}{a_{ii}}\left\{ r_i - \sum_{j<i}^{N} a_{ij} x_j^{(k+\frac{1}{2})} - \sum_{j>i}^{N} a_{ij} x_j^{(k)} \right\}$$

Backward Loop

```
for (int i = nrow-1; i  >= 0; i--) {
            …
  double sum = r[i];
  for (int j = 0; j < nzInRow[i]; j++)
    sum -= Val[j]*x[Col[j]];
   sum += x[i]*Diag;
   x[i] = sum/Diag;
}
```



$$x_i^{(k+1)} = \frac{1}{a_{ii}}\left\{ r_i - \sum_{j<i}^{N} a_{ij} x_j^{(k+\frac{1}{2})} - \sum_{j>i}^{N} a_{ij} x_j^{(k+1)} \right\}$$

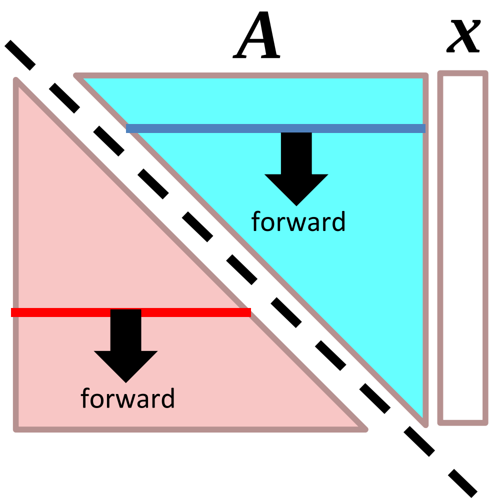
Symmetric Gauss-Seidel

- *Forward* loop can be split into two loops: *Loop1 & Loop2*

Loop2

```
for (int i = 0; i < nrow; i++) {
            …
  double sum = y[i];
  for (int j = 0; j < nzLInRow[i]; j++)
    sum -= Val[j]*x[Col[j]];
  sum += x[i]*Diag;     updated by Loop2
  x[i] = sum/Diag;      previously
}
```

$$x_i^{(k+\frac{1}{2})} = \frac{1}{a_{ii}}\left\{ y_i - \sum_{j<i}^{N} a_{ij} x_j^{(k+\frac{1}{2})} \right\}$$



Loop1

```
#omp parallel for
for (int i = 0; i < nrow; i++) {
        …
  double sum = r[i];
  for (int j = nzLInRow[i];
               j < nzInRow[i]; j++)
    sum -= Val[j]*x[Col[j]];
  y[i] = sum;        defined before Loop1
}                    and not updated
```

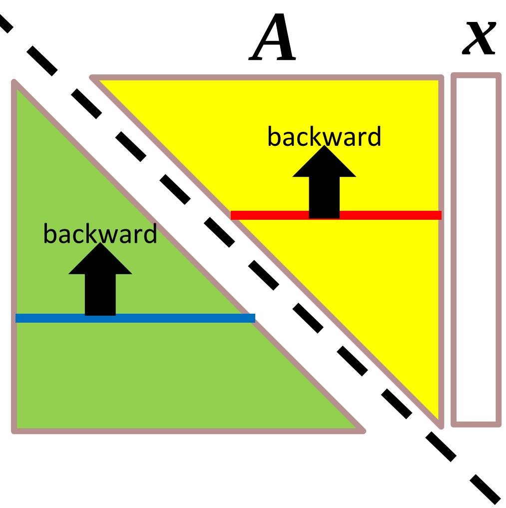
$$y_i = r_i - \sum_{j>i}^{N} a_{ij} x_j^{(k)}$$

- *Backward* loop can be split into two loops: *Loop3 & Loop4*

Loop3

```
#omp parallel for
for (int i = nrow-1; i >= 0; i--) {
            …
  double sum = r[i];
  for (int j = 0; j < nzLInRow[i]; j++)
    sum -= Val[j]*x[Col[j]];
  y[i] = sum;        updated by Loop2
}                    and not updated
```

$$y_i = r_i - \sum_{j<i}^{N} a_{ij} x_j^{(k+\frac{1}{2})}$$



Loop4

```
for (int i = nrow-1; i >= 0: i--) {
        …
  double sum = y[i];
  for (int j = nzLInRow[i];
              j < nzInRow[i];  j++)
    sum -= Val[j]*x[Col[j]];
  sum += x[i]*Diag;   updated by Loop4
  x[i] = sum/Diag;    previously
}
```

$$x_i^{(k+1)} = \frac{1}{a_{ii}}\left\{ y_i - \sum_{j>i}^{N} a_{ij} x_j^{(k+1)} \right\}$$

- *Execution Order*: *Loop1 → Loop2 → Loop3 → Loop4*

Symmetric Gauss-Seidel

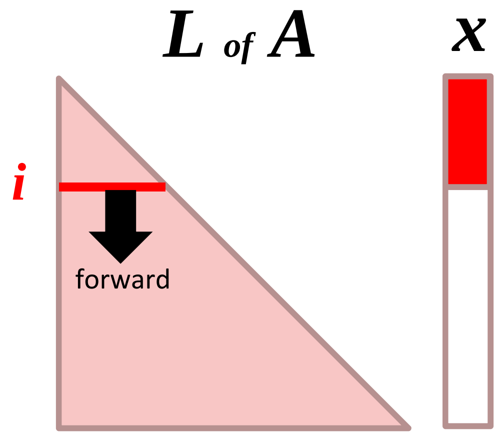
- *Loop3* can be reversed
- Row *i* of *Loop3* can be calculated *only if* from row 0 to *i* of *Loop2* are calculated → more chance to use cache effectively

Loop direction reversing does not change arithmetic order!!

Loop2

```
for (int i = 0; i < nrow; i++) {
              …
  double sum = y[i];
  for (int j = 0; j < nzLInRow[i]; j++)
     sum -= Val[j]*x[Col[j]];
   sum += x[i]*Diag;
   x[i] = sum/Diag;
}
```

defind in Loop2

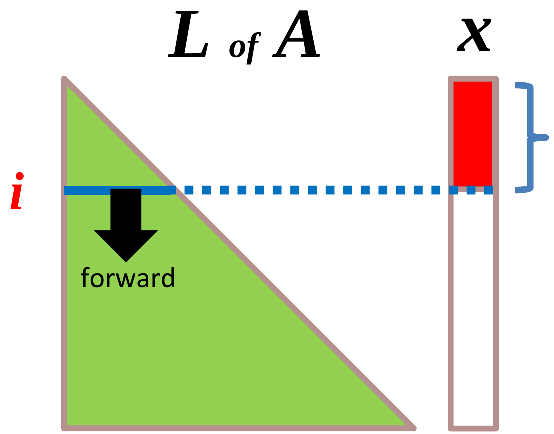


$$x_i^{(k+\frac{1}{2})} = \frac{1}{a_{ii}} \left\{ y_i - \sum_{j<i}^{N} a_{ij} x_j^{(k+\frac{1}{2})} \right\}$$

Loop3

```
for (int i = 0; i < nrow; i++) {
              …
  double sum = r[i];
  for (int j = 0; j < nzLInRow[i]; j++)
     sum -= Val[j]*x[Col[j]];
   y[i] = sum;
}
```

defined in Loop2



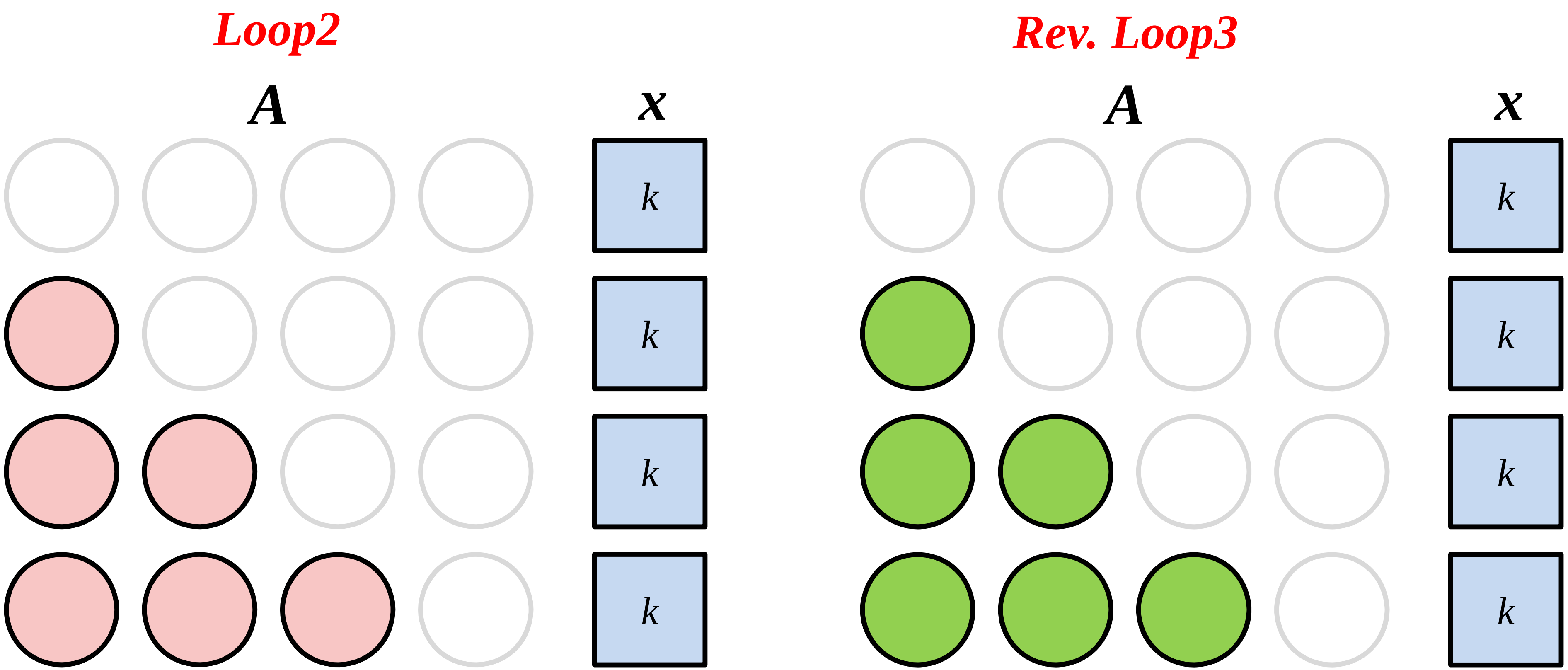
updated row 0 to i of Loop2

$$y_i = r_i - \sum_{j<i}^{N} a_{ij} x_j^{(k+\frac{1}{2})}$$

Symmetric Gauss-Seidel

Marching of updating of *Loop2* and *Loop3*
(sample by 4X4 problem)

initial



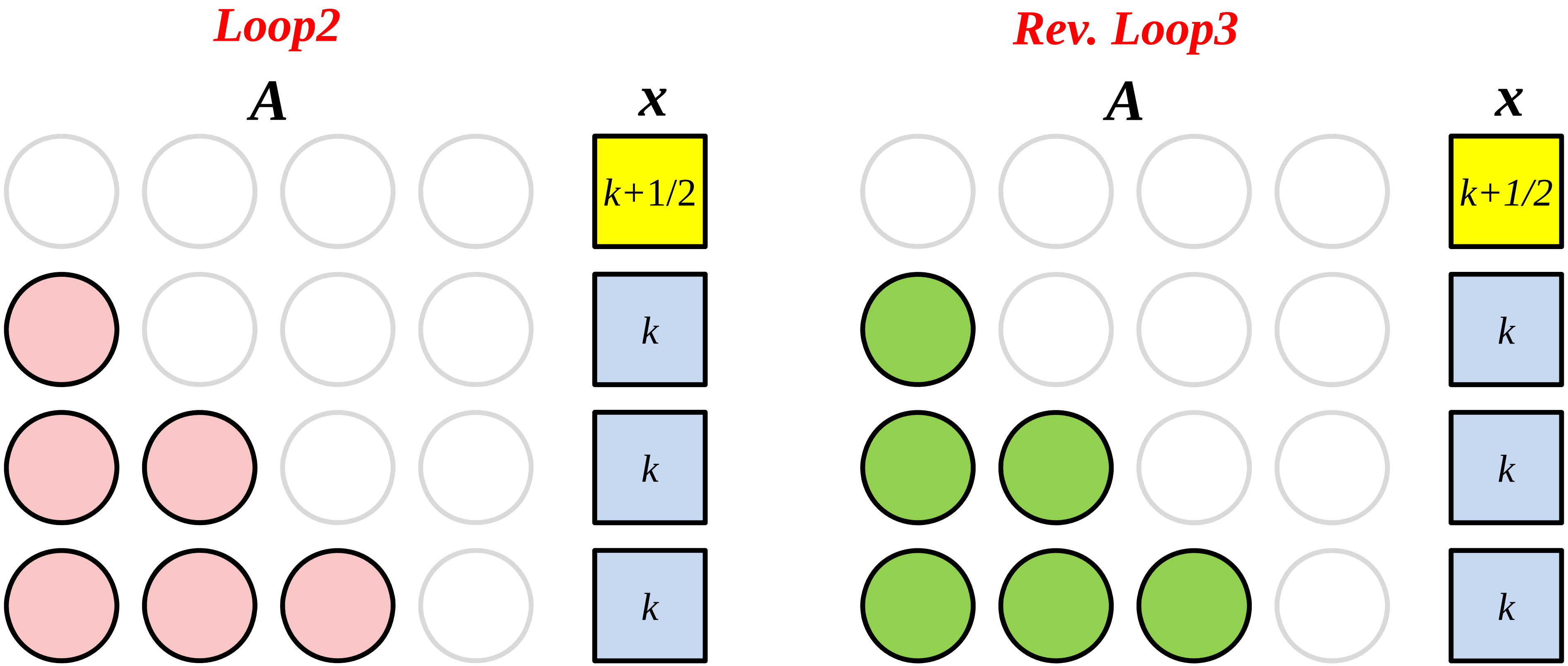
$$x_i^{\left(k+\frac{1}{2}\right)} = \frac{1}{a_{ii}}\left\{ y_i - \sum_{j<i}^{N} a_{ij} x_j^{\left(k+\frac{1}{2}\right)} \right\}$$

$$y_i = r_i - \sum_{j<i}^{N} a_{ij} x_j^{\left(k+\frac{1}{2}\right)}$$

Symmetric Gauss-Seidel

Marching of updating of *Loop2* and *Loop3*
(sample by 4X4 problem)

i=1



$$x_i^{\left(k+\frac{1}{2}\right)} = \frac{1}{a_{ii}}\left\{ y_i - \sum_{j<i}^{N} a_{ij} x_j^{\left(k+\frac{1}{2}\right)} \right\}$$

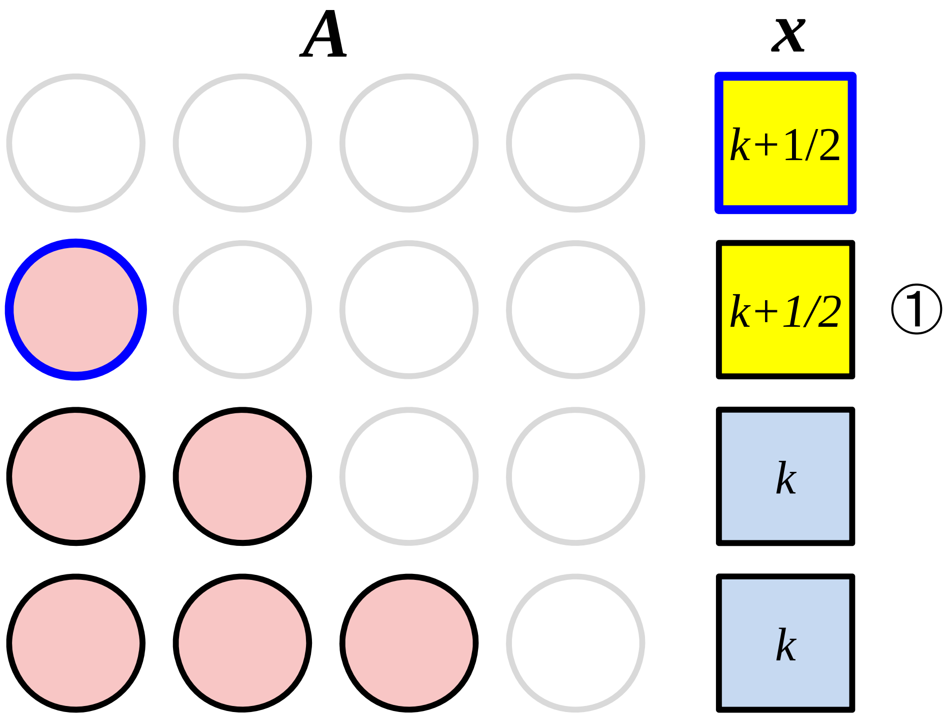
$$y_i = r_i - \sum_{j<i}^{N} a_{ij} x_j^{\left(k+\frac{1}{2}\right)}$$

Symmetric Gauss-Seidel

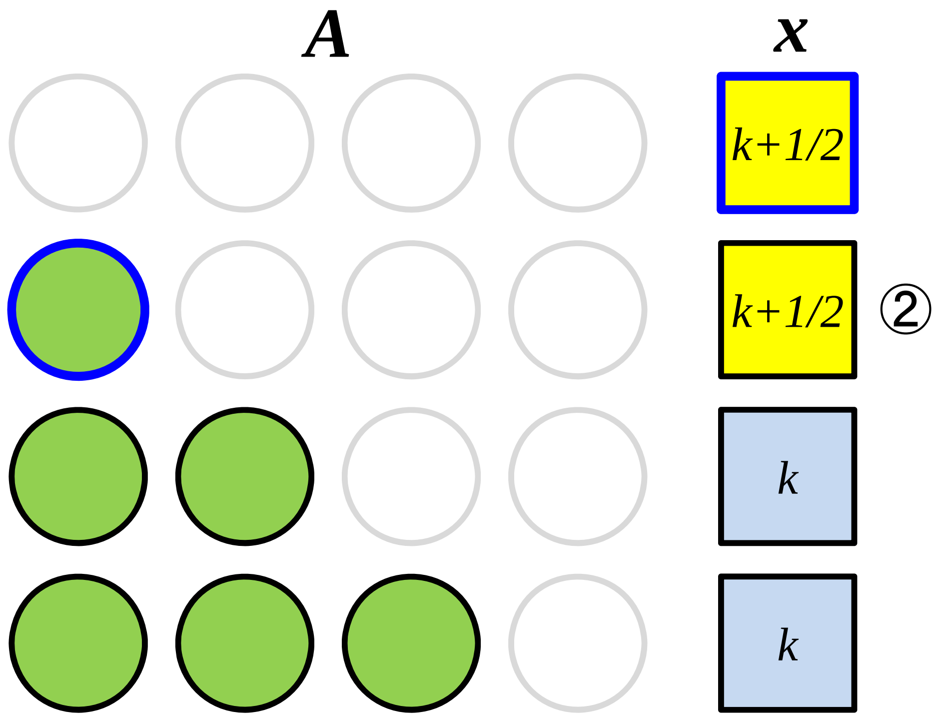
Marching of updating of *Loop2* and *Loop3*
(sample by 4X4 problem)

i=2

Loop2



Rev. Loop3



$$x_i^{\left(k+\frac{1}{2}\right)} = \frac{1}{a_{ii}} \left\{ y_i - \sum_{j<i}^{N} a_{ij} x_j^{\left(k+\frac{1}{2}\right)} \right\}$$

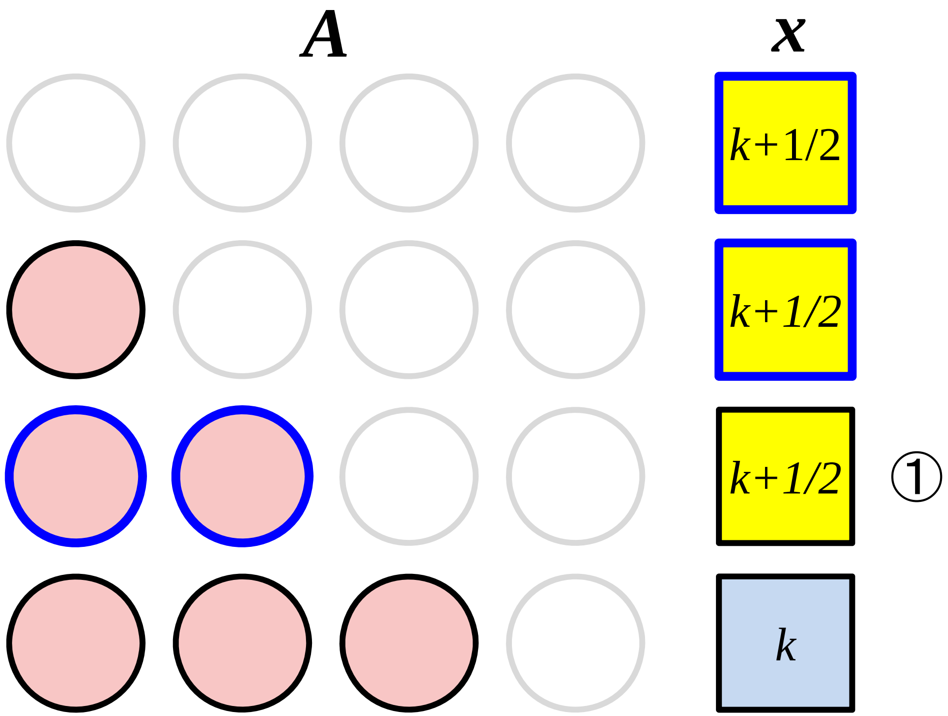
$$y_i = r_i - \sum_{j<i}^{N} a_{ij} x_j^{\left(k+\frac{1}{2}\right)}$$

Symmetric Gauss-Seidel

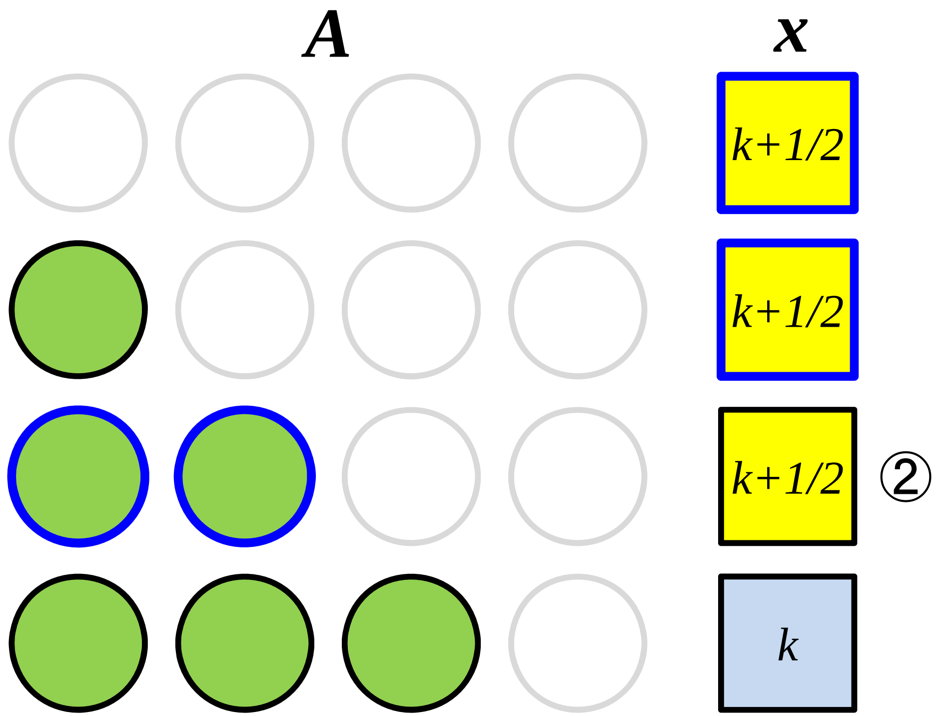
Marching of updating of *Loop2* and *Loop3*
(sample by 4X4 problem)

i=3

Loop2



Rev. Loop3



$$x_i^{\left(k+\frac{1}{2}\right)} = \frac{1}{a_{ii}}\left\{ y_i - \sum_{j<i}^{N} a_{ij} x_j^{\left(k+\frac{1}{2}\right)} \right\}$$

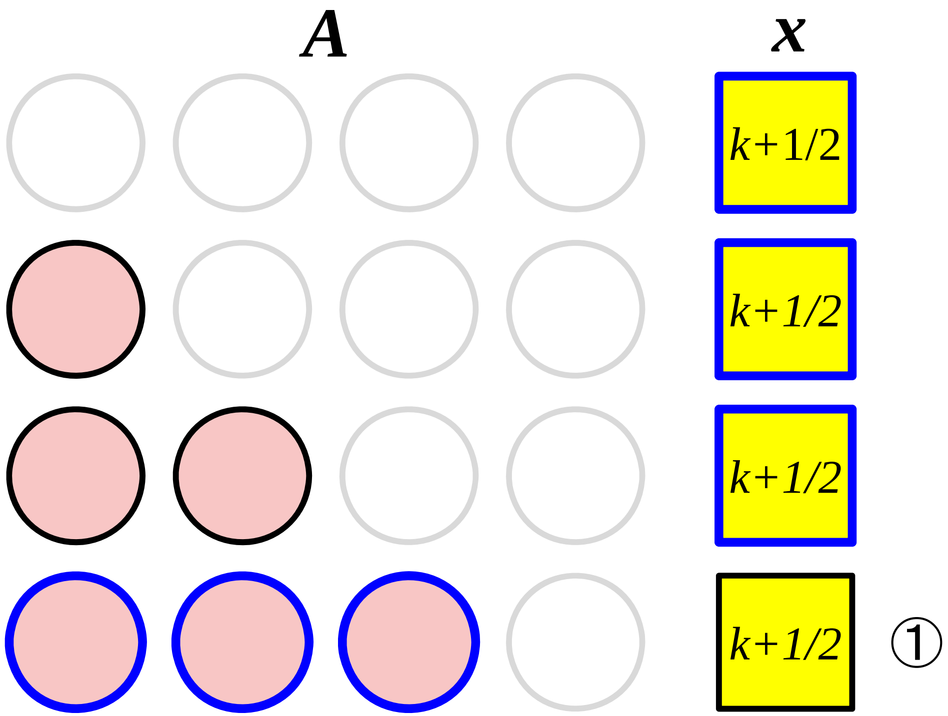
$$y_i = r_i - \sum_{j<i}^{N} a_{ij} x_j^{\left(k+\frac{1}{2}\right)}$$

Symmetric Gauss-Seidel

Marching of updating of *Loop2* and *Loop3*
(sample by 4X4 problem)

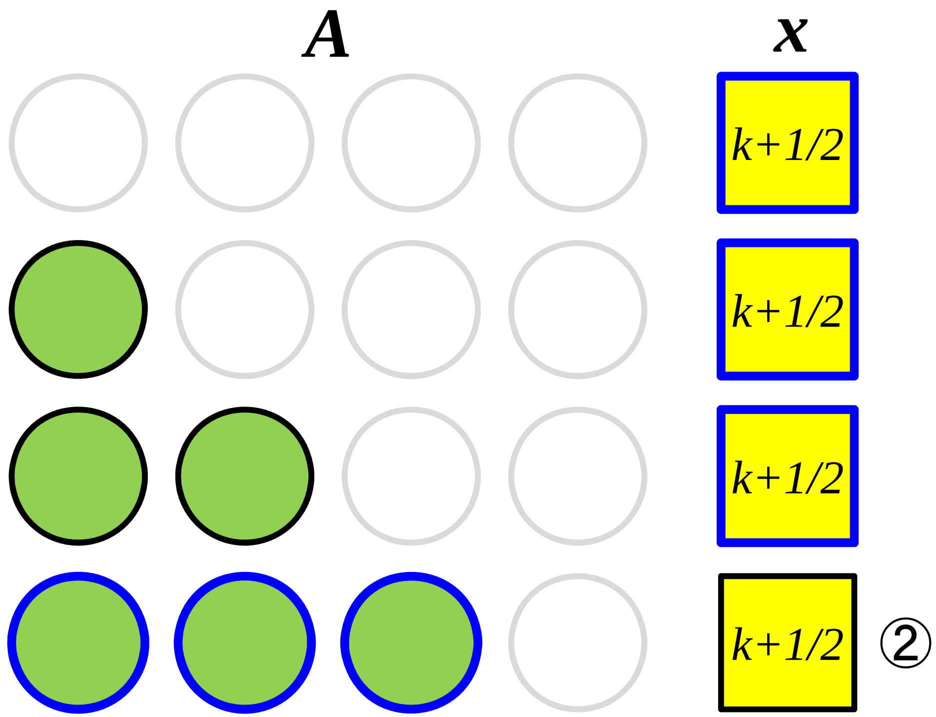
i=4

Loop2



$$x_i^{\left(k+\frac{1}{2}\right)} = \frac{1}{a_{ii}}\left\{ y_i - \sum_{j<i}^{N} a_{ij} x_j^{\left(k+\frac{1}{2}\right)} \right\}$$

Rev. Loop3



$$y_i = r_i - \sum_{j<i}^{N} a_{ij} x_j^{\left(k+\frac{1}{2}\right)}$$

History of score improvement

BoF@	Tune	PFLOPS	RANK
SC15	Previous DOI: 10.1177/1094342015607950	0.461	2
ISC2016	Cache utilization by splitting SYMGS loops (just implement)	0.554	2
SC16	Cache utilization by splitting SYMGS loops (more adjustment)	0.602	1