H26年度
並列コンピュータの高速化技法入門
参考資料

2015年　1月21日

大阪大学サイバーメディアセンター
日本電気株式会社

本資料は，東北大学サイバーサイエンスセンターとＮＥＣの
共同により作成され，大阪大学サイバーメディアセンターの
環境で実行確認を行い，修正を加えたものです．

キャッシュブロッキング

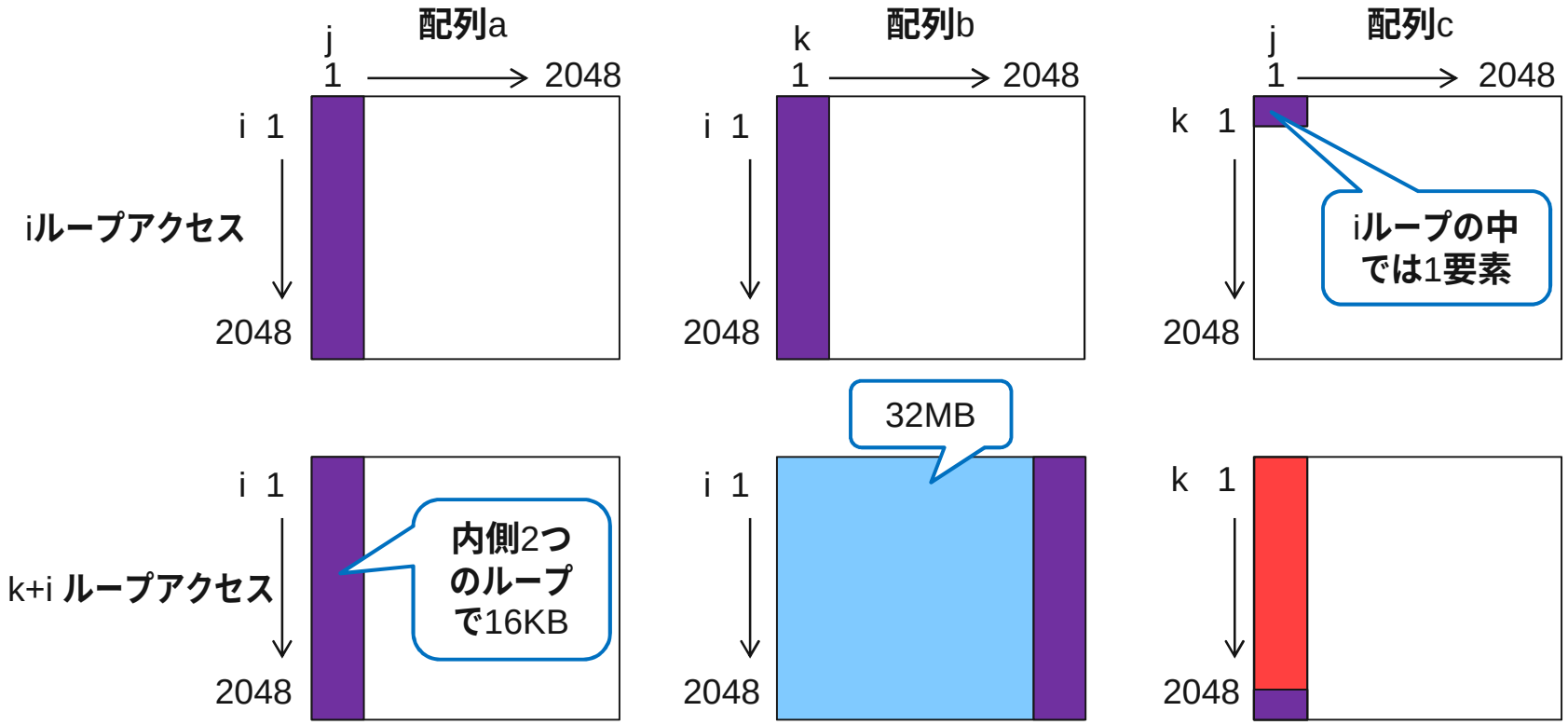
メモリアクセスの確認

- iループとkループ時の配列アクセス

倍精度の配列
2048*8/1024=16K
2048*2048*8/1024/1024=32MB

行列積コード

```
do j=1,n
   do k=1,n
      do i=1,n
         a(i,j)=a(i,j)+b(i,k)*c(k,j)
```



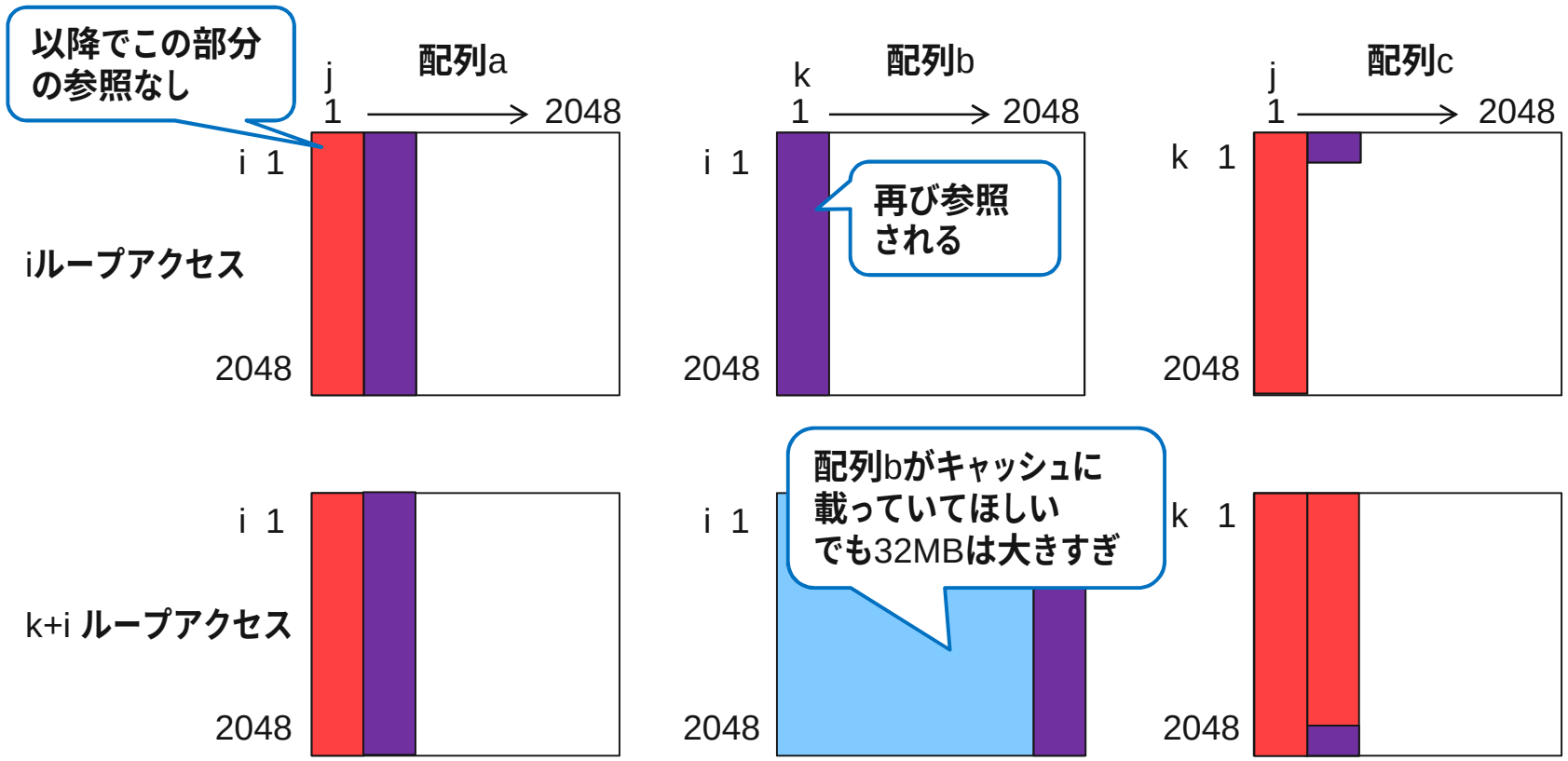
キャッシュブロッキング

メモリアクセスの確認(続き)

- 外側jが1つ進んだ時の配列アクセス

行列積コード

```
do j=1,n
   do k=1,n
      do i=1,n
        a(i,j)=a(i,j)+b(i,k)*c(k,j)
```



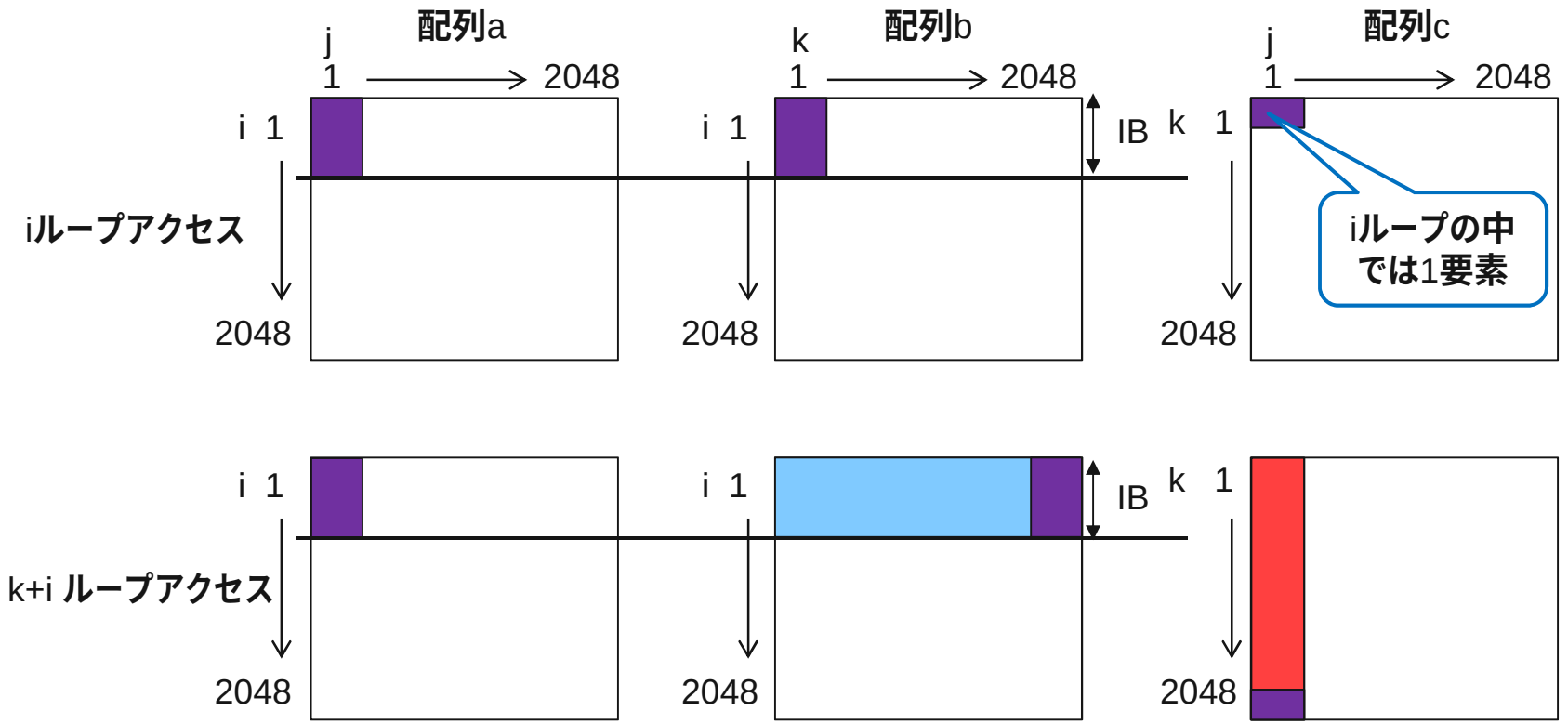
キャッシュブロッキング

iのブロッキングの場合

- iをブロッキングした時の iループとkループ配列アクセス
- IBがブロッキングサイズ

行列積コード

```
do ii=1,n,IB
do j=1,n
   do k=1,n
      do i=ii,min(n,ii+IB-1)
         a(i,j)=a(i,j)+b(i,k)*c(k,j)
```



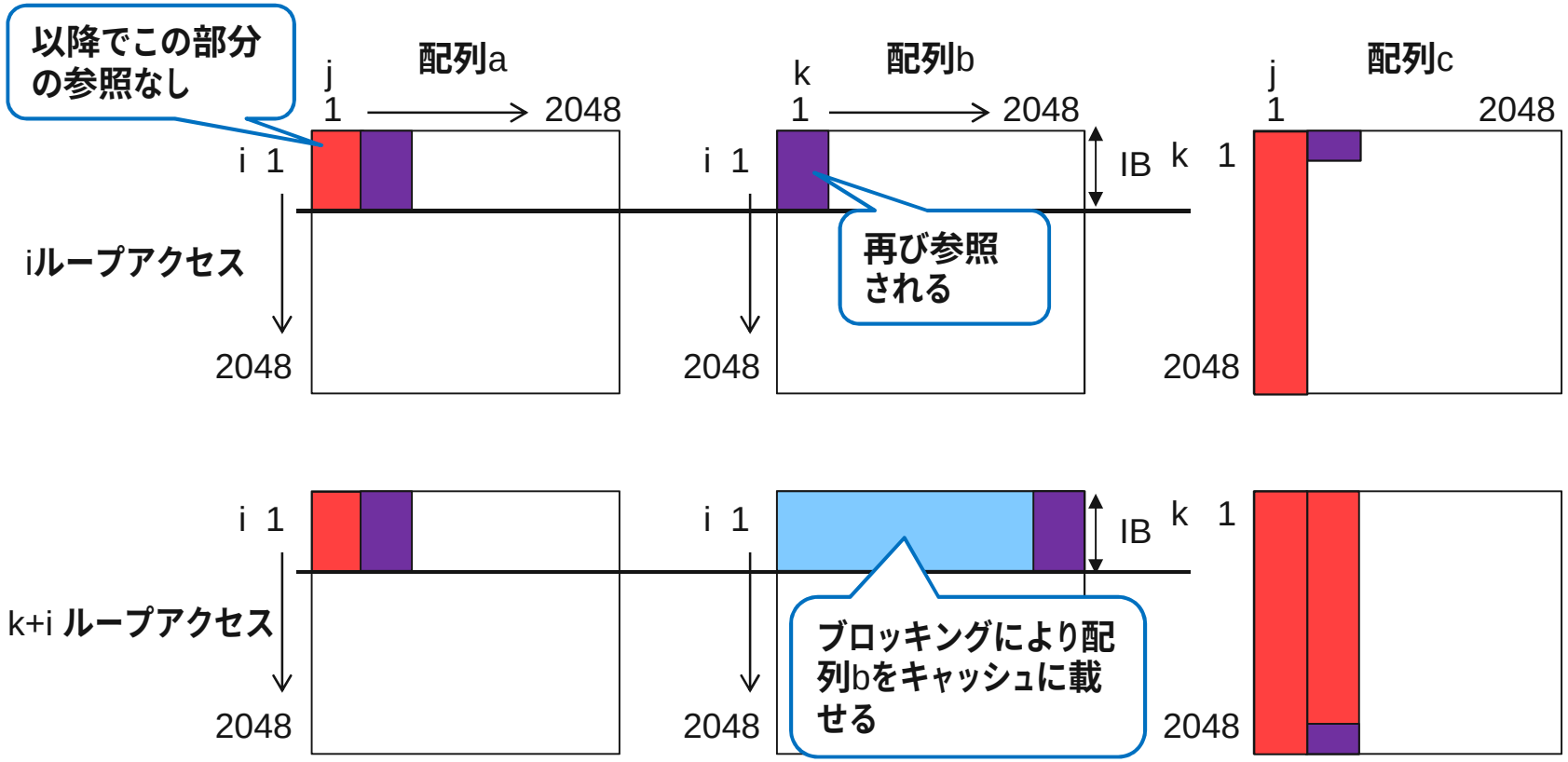
キャッシュブロッキング

iのブロッキングの場合(続き)

- iをブロッキングし、外側jが1つ進んだ時の配列アクセス
- 配列bに着目し、2048*IBのサイズがキャッシュに収まればjループの繰り返しの間はキャッシュヒット

行列積コード

```
do ii=1,n,IB
do j=1,n
   do k=1,n
      do i=ii,min(n,ii+IB-1)
        a(i,j)=a(i,j)+b(i,k)*c(k,j)
```



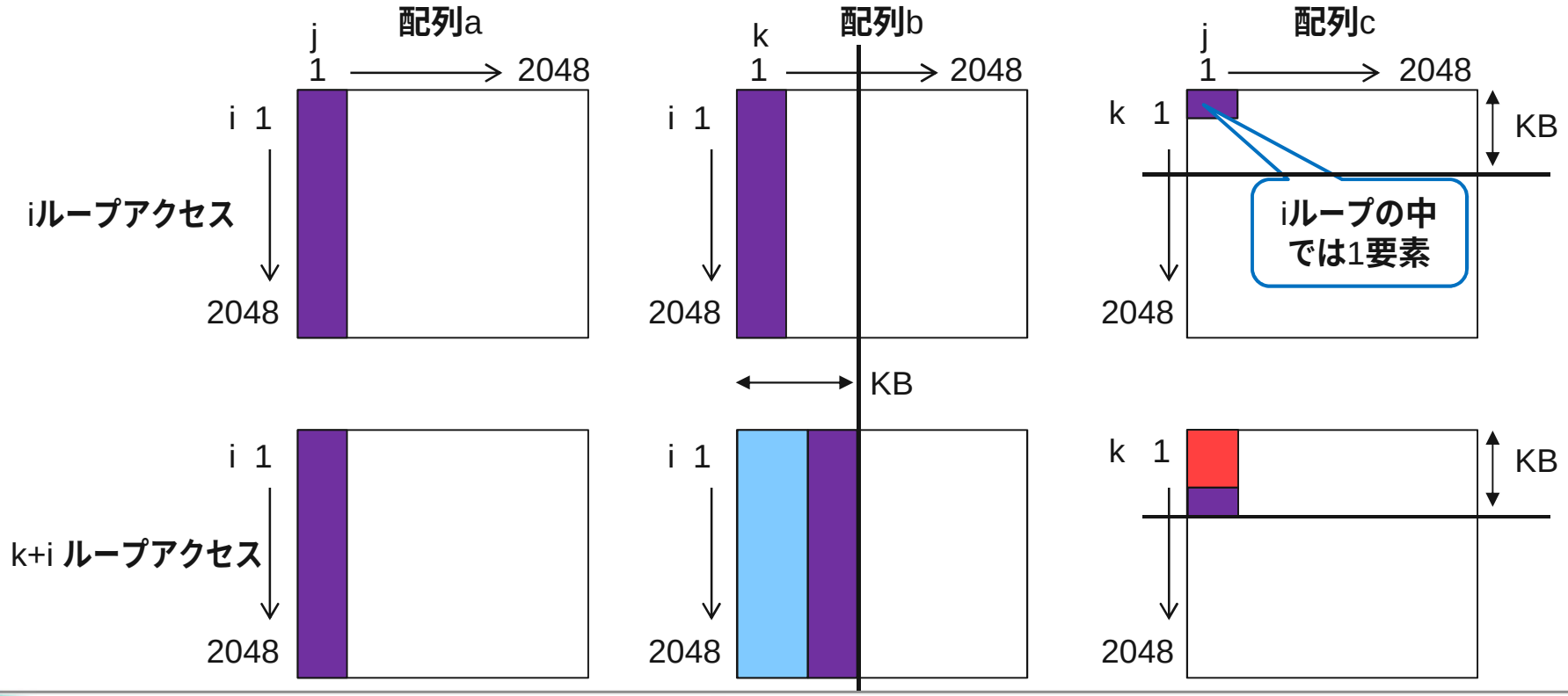
キャッシュブロッキング

kのブロッキングの場合

- kをブロッキングした時の iループとkループ配列アクセス
- KBがブロッキングサイズ

行列積コード

```
do kk=1,n,KB
do j=1,n
   do k=kk,min(n,kk+KB-1)
      do i=1,n
         a(i,j)=a(i,j)+b(i,k)*c(k,j)
```



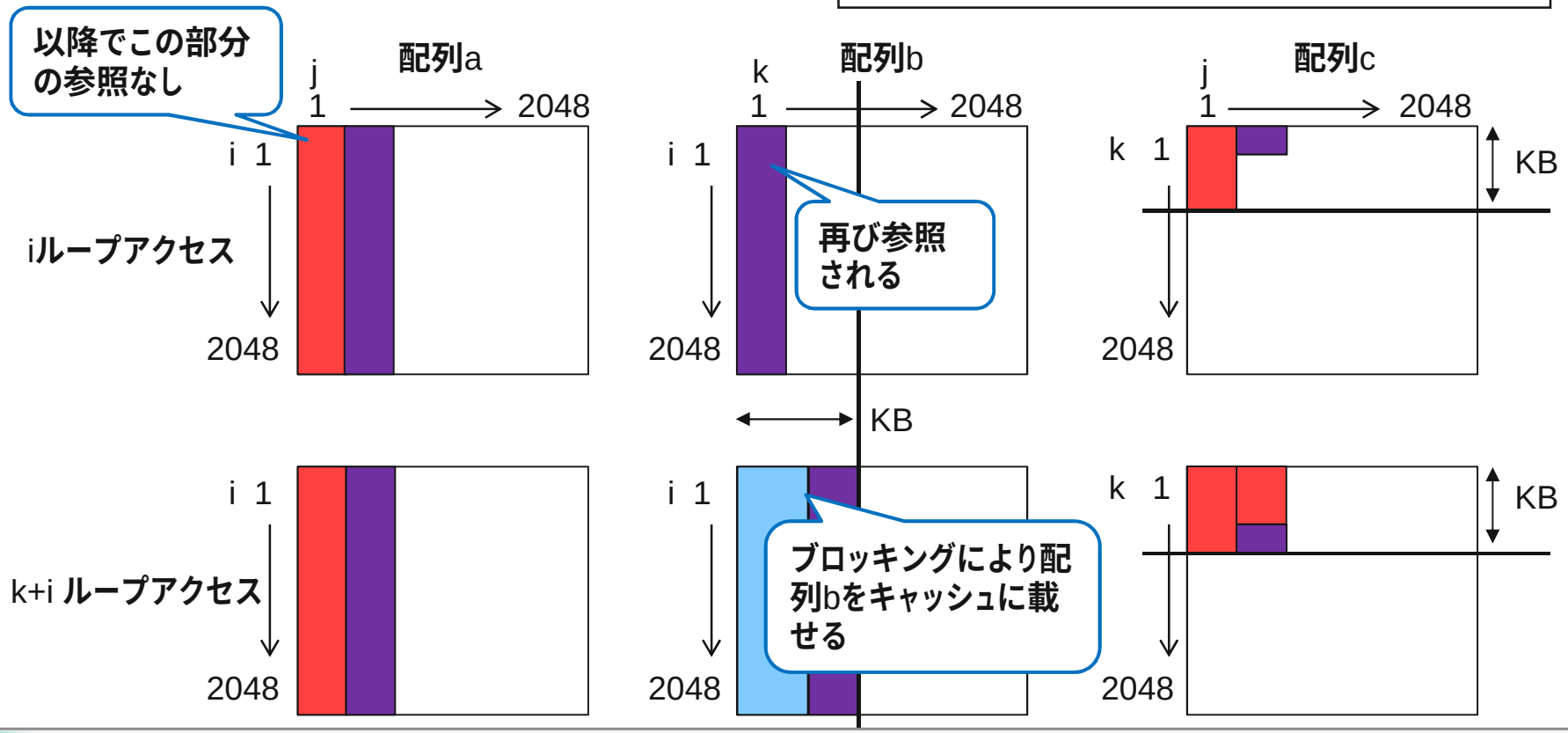
キャッシュブロッキング

kのブロッキングの場合(続き)

- kをブロッキングし
  外側jが1つ進んだ時の配列アクセス
- 配列bに着目し、2048*KBのサイズが
  キャッシュに収まればjループの繰り返しの
  間はキャッシュヒット

行列積コード

```
do kk=1,n,KB
do j=1,n
   do k=kk,min(n,kk+KB-1)
      do i=1,n
        a(i,j)=a(i,j)+b(i,k)*c(k,j)
```



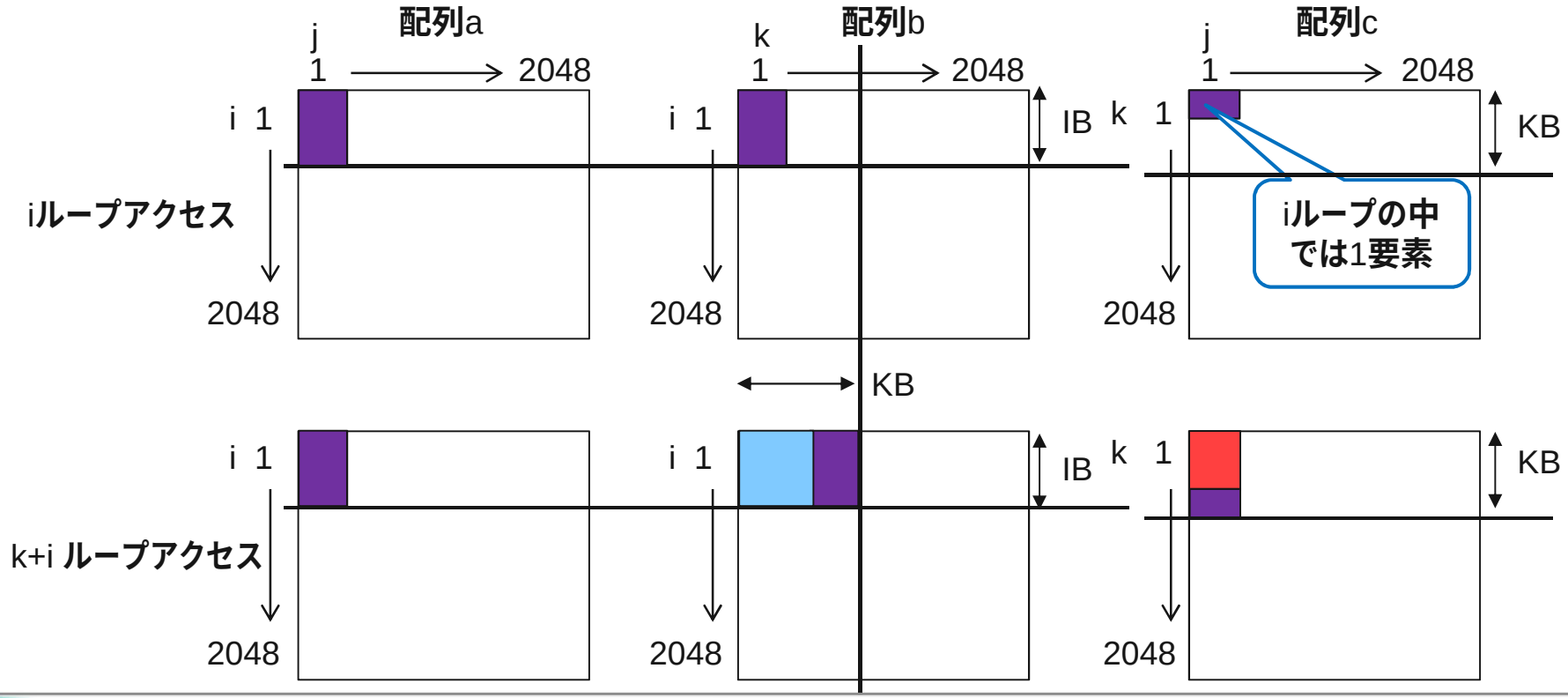
キャッシュブロッキング

iとkのブロッキングの場合

- iとkをブロッキングした時の iループとkループ配列アクセス
- IB、KBがブロッキングサイズ

行列積コード

```
do kk=1,n,KB
do ii=1,n,IB
do j=1,n
   do k=kk,min(n,kk+KB-1)
      do i=ii,min(n,ii+IB-1)
         a(i,j)=a(i,j)+b(i,k)*c(k,j)
```



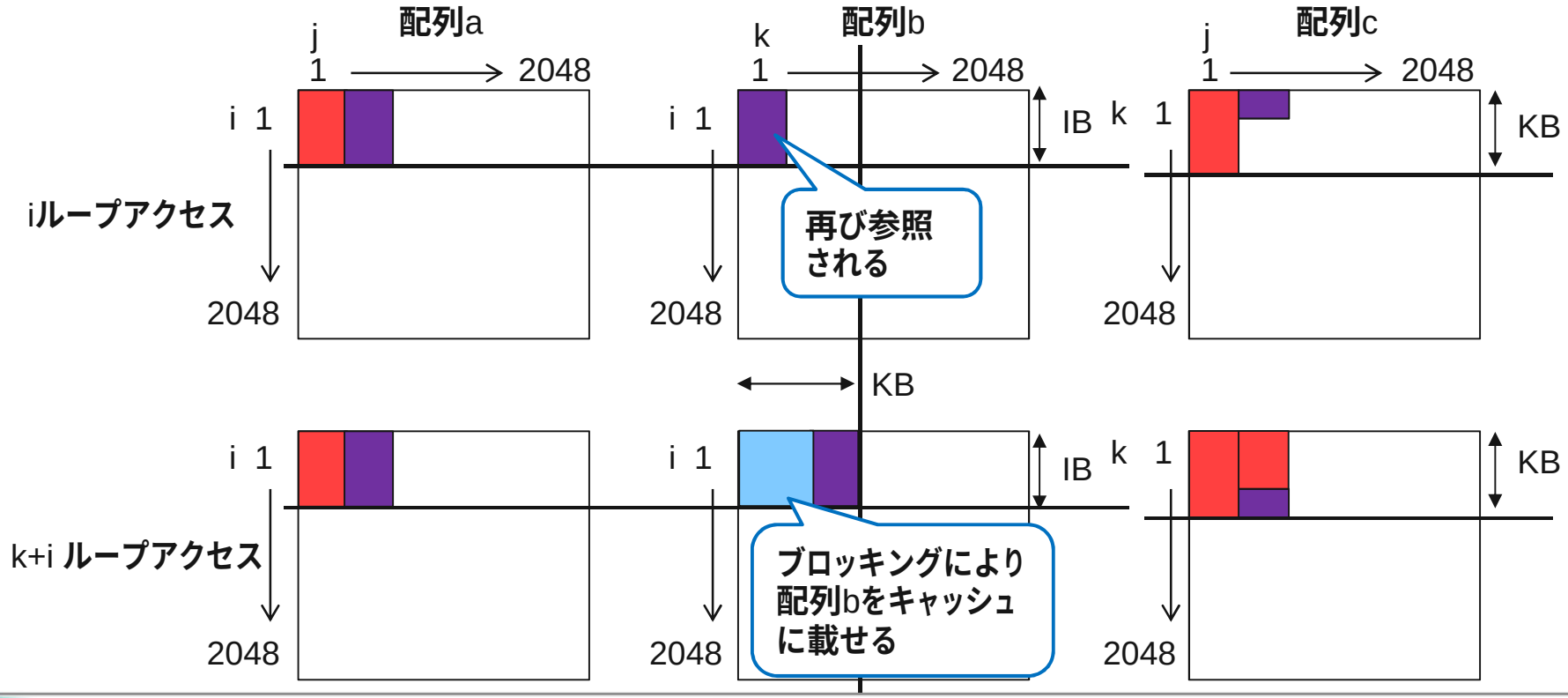
キャッシュブロッキング

iとkのブロッキングの場合(続き)

- iとkをブロッキングし
  外側jが1つ進んだ時の配列アクセス
- 配列bに着目し、IB*KBのサイズが
  キャッシュに収まればjループの繰り返しの
  間はキャッシュヒット

行列積コード

```
do kk=1,n,KB
do ii=1,n,IB
do j=1,n
   do k=kk,min(n,kk+KB-1)
      do i=ii,min(n,ii+IB-1)
        a(i,j)=a(i,j)+b(i,k)*c(k,j)
```



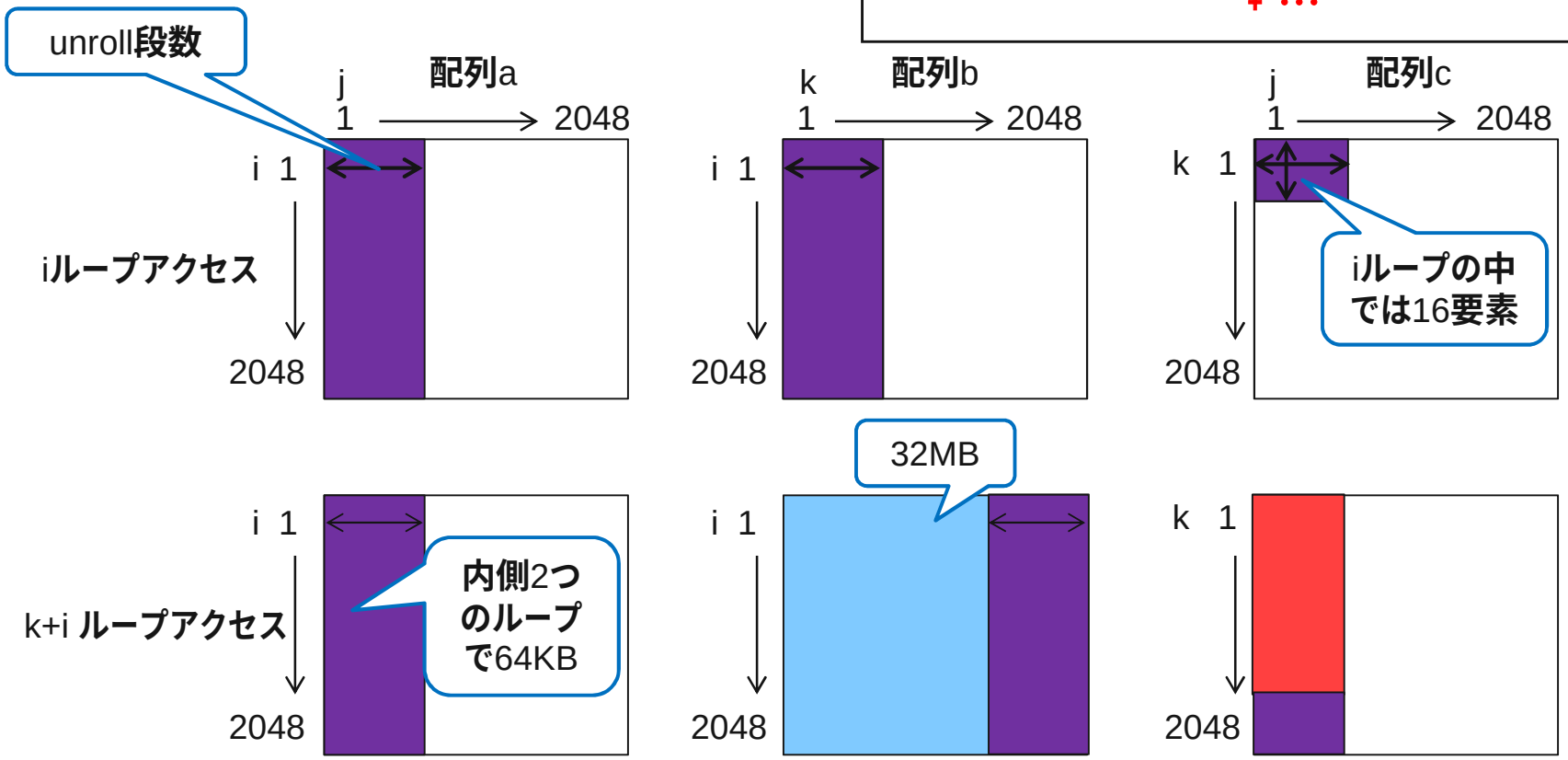
キャッシュブロッキング

unrollありでのメモリアクセスの確認

- iループとkループ時の配列アクセス
  外側2ループがそれぞれ4段unroll
- 以降、jループを進めた時のアクセスと
  ブロッキングの考え方は、前記と同じ

行列積コード

```
do j=1,n,4
   do k=1,n,4
      do i=1,n
        a(i,j)=a(i,j)+b(i,k)*c(k,j)
                     +b(i,k+1)*c(k+1,j)
                     + ・・・
        a(i,j+1)=a(i,j+1)+b(i,k)*c(k,j+1)
                         + ・・・
```

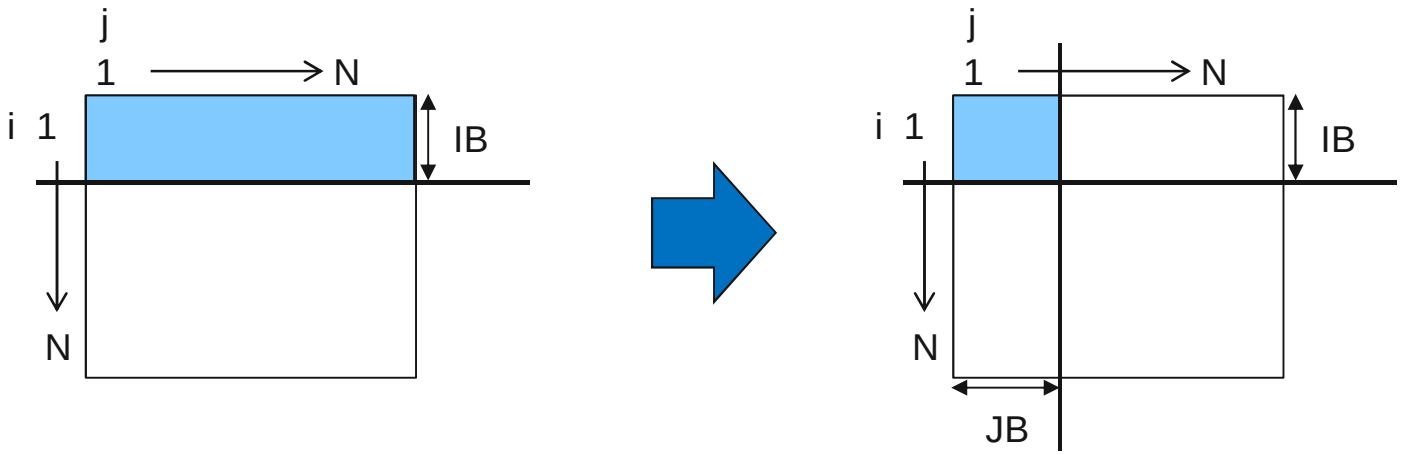


キャッシュブロッキング

注意点

- ブロッキングサイズはL2キャッシュサイズをターゲットとする
- ブロッキングサイズと最内ループ長に注意する

→ 最内ループ長が短くなりすぎない、かつ、キャッシュに載るサイズを考える
必要なら複数のブロッキングを組み合わせる



Nが大きい場合、キャッシュに載せるために
IBを小さくすると、最内がiとなるループの
ループ長が短くなる

IBとJBのブロッキングを組み合わせ、
IBを小さくしすぎないようにする