Multiplication and Truncation to `rkmatrix`

Task: compute $\mathcal{T}_{\mathcal{H}\to k}(r + a \cdot b)$

Multiplication and Truncation to `rkmatrix`

Task: compute $\mathcal{T}_{\mathcal{H}\to k}(r + a \cdot b)$

```
void addprod2_rkmatrix(prkmatrix r, psupermatrix a, psupermatrix b){
   ...
   brows = a->block_rows;
   bmid = a->block_cols;
   bcols = b->block_cols;
```

Multiplication and Truncation to `rkmatrix`

Task: compute $\mathcal{T}_{\mathcal{H}\to k}(r + a \cdot b)$

```
void addprod2_rkmatrix(prkmatrix r, psupermatrix a, psupermatrix b){
   ...
   brows = a->block_rows;
   bmid = a->block_cols;
   bcols = b->block_cols;
   if(a->s!=0x0 && b->s!=0x0){
      /* allocate rk_no,rk_mo,rk_r */
      no = 0;
      for(i=0,mo=0; i<brows; i++){
         for(j=0; j<bcols; j++){
            rk_no[i+j*brows] = no;
            rk_mo[i+j*brows] = mo;









      addparts2_rkmatrix(r,brows*bcols,rk_no,rk_mo,rk_r);
   }else{
      ...  /* no hierarchical conversion necessary */
   }}
```

Multiplication and Truncation to rkmatrix

Task: compute $\mathcal{T}_{\mathcal{H}\to k}(r + a \cdot b)$

```
void addprod2_rkmatrix(prkmatrix r, psupermatrix a, psupermatrix b){
   ...
   brows = a->block_rows;
   bmid = a->block_cols;
   bcols = b->block_cols;
   if(a->s!=0x0 && b->s!=0x0){
      /* allocate rk_no,rk_mo,rk_r */
      no = 0;
      for(i=0,mo=0; i<brows; i++){
         for(j=0; j<bcols; j++){
            rk_no[i+j*brows] = no;
            rk_mo[i+j*brows] = mo;
             rk_r[i+j*brows] = new_rkmatrix(r->k,a->s[i]->rows,b->s[j*bmid]->cols);
            for(k=0; k<bmid; k++)
               addprod2_rkmatrix(rk_r[i+j*brows],a->s[i+k*brows],b->s[k+j*bmid]);
            mo += b->s[j*brows]->cols;
         }
         no += a->s[i]->rows;}
      addparts2_rkmatrix(r,brows*bcols,rk_no,rk_mo,rk_r);
   }else{
      ...  /* no hierarchical conversion necessary */
   }}
```

Multiplication and Truncation to `supermatrix`

Task: compute $\mathcal{T}_k(c + a \cdot b)$

Multiplication and Truncation to `supermatrix`

Task: compute $\mathcal{T}_k(c + a \cdot b)$

```
void
muladd_supermatrix(psupermatrix c, psupermatrix a, psupermatrix b){
    ...
    brows = a->block_rows;
    bmid = a->block_cols;
    bcols = b->block_cols;
```

Multiplication and Truncation to supermatrix

Task: compute $\mathcal{T}_k(c + a \cdot b)$

```
void
muladd_supermatrix(psupermatrix c, psupermatrix a, psupermatrix b){
   ...
   brows = a->block_rows;
   bmid = a->block_cols;
   bcols = b->block_cols;
    if(a->s!=0x0 && b->s!=0x0 && c->s!=0x0){
      for(i=0; i<brows; i++)
         for(j=0; j<bcols; j++)
            for(k=0; k<bmid; k++)
               muladd_supermatrix(c->s[i+j*brows],
                                  a->s[i+k*brows],b->s[k+j*bmid]);
   }else{




    }}
```

Multiplication and Truncation to `supermatrix`

Task: compute $\mathcal{T}_k(c + a \cdot b)$

```
void
muladd_supermatrix(psupermatrix c, psupermatrix a, psupermatrix b){
   ...
   brows = a->block_rows;
   bmid = a->block_cols;
   bcols = b->block_cols;
    if(a->s!=0x0 && b->s!=0x0 && c->s!=0x0){
      for(i=0; i<brows; i++)
         for(j=0; j<bcols; j++)
            for(k=0; k<bmid; k++)
               muladd_supermatrix(c->s[i+j*brows],
                                  a->s[i+k*brows],b->s[k+j*bmid]);
   }else{
       if(a->s!=0x0 && b->s!=0x0 && c->r!=0x0){
         addprod2_rkmatrix(c->r,a,b);
      }else{

       }
    }}
```

Multiplication and Truncation to supermatrix

Task: compute $\mathcal{T}_k(c + a \cdot b)$

```
void
muladd_supermatrix(psupermatrix c, psupermatrix a, psupermatrix b){
   ...
   brows = a->block_rows;
   bmid = a->block_cols;
   bcols = b->block_cols;
    if(a->s!=0x0 && b->s!=0x0 && c->s!=0x0){
      for(i=0; i<brows; i++)
         for(j=0; j<bcols; j++)
            for(k=0; k<bmid; k++)
               muladd_supermatrix(c->s[i+j*brows],
                                  a->s[i+k*brows],b->s[k+j*bmid]);
   }else{
       if(a->s!=0x0 && b->s!=0x0 && c->r!=0x0){
         addprod2_rkmatrix(c->r,a,b);
      }else{
          /* nothing hierarchical:  fullmatrix/rkmatrix */
       }
    }}
```

MULTIPLICATION