```
//Original CUDA code
for(k=0;k<B2_dimension;k++){
  pA2_begin=i_blockStarts[block];
  pA2_end=i_blockStarts[block+1];
  fposA=block*256;
  i_pos=taco_binarySearchBefore(
  A2_pos,pA2_begin,pA2_end,fposA);
  i=i_pos;
  fposA=block*256+fpos1;
  if(fposA>=A2_pos[A1_dimension])
    break;
  f=A2_crd[fposA];
  kB=f*B2_dimension+k;
  while(fposA==A2_pos[i_pos+1]){
    i_pos=i_pos+1;
    i=i_pos;
  }
  kC=i*C2_dimension+k;
  float val=0.0;
  val=A_vals[fposA]*B_vals[kB];
  atomicAdd(&C_vals[kC],val);
}
```

```
//Modified CUDA code
for(k=0;k<B2_dimension;k++){
  pA2_begin=i_blockStarts[block];
  pA2_end=i_blockStarts[block+1];
  fposA=block*256+fpos1;
  i_pos=taco_binarySearchBefore(
  A2_pos,pA2_begin,pA2_end,fposA);
  i=i_pos;
  float val=0.0;
  if(fposA>=A2_pos[A1_dimension])
    val=0;
  else{
    f=A2_crd[fposA];
    kB=f*B2_dimension+k;
    while(fposA==A2_pos[i_pos+1]){
      i_pos=i_pos+1;
      i=i_pos;
    }
    val=A_vals[fposA]*B_vals[kB];
  }
  kC=i*C2_dimension+k;
  segReduceWarp<float,32>(C_vals,
  kC,val);
}
```